Politechnika Warszawska

Wydział Matematyki i Nauk Informacyjnych

Sieci neuronowe

Rozpoznawanie i klasyfikacja pisanych cyfr przy użyciu modeli matematycznych - raport

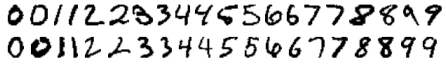
Autorzy:

Anna Zawadzka
Piotr Waszkiewicz

21 stycznia 2017

1 Opis problemu badawczego

Problem badawczy przedstawiony na stronie *https://www.kaggle.com/c/digit-recognizer* polega na rozpoznawaniu i klasyfikacji ręcznie pisanych cyfr poprzez przetwarzanie i analizę obrazów przedstawiających odpowiednie symbole. Zbiory danych zostały zaczerpnięte z publicznej bazy danych MNIST[1].



2 Cel badań

Projekt zakładał realizację zadania poprzez zbadanie różnych metod, ze szczególnym uwzględnieniem różnych modeli sieci neuronowych. Zbadane zostały dwa rodzaje takich sieci - Backpropagation oraz SoftMax. Wykorzystane zostały również jedne z najpopularniejszych obecnie klasyfikatorów: maszyny wektorów podpierających (SVM)[3], Lasy Losowe[2], kNN[4].

Celem badań było porównanie jakości klasyfikacji dla różnych modeli klasyfikatorów i wskazanie najskuteczniejszego pod względem czasu uczenia, wydajności i jakości udzielanych odpowiedzi. Oprócz tego badania miały na celu rozszerzenie istniejącego wektora cech o nowe, unikalne wartości które, jak przypuszczano, polepszyłyby jakość klasyfikacji. W trakcie obliczeń podjęta została próba odrzucenia tych cech które przeszkadzają lub pogarszają działanie modeli.

3 Opis danych

Zbiory danych treningowych oraz testowych pochodzą z publicznej bazy danych MNIST[1]. Każdy element ze zbioru treningowego jest opisany 785 wartościami. Pierwsza liczba określa zakodowaną cyfrę (wartość z przedziału [0, 9]), kolejne 784 wartości są z przedziału [0, 255] i opisują kolory pikseli zeskanowanej cyfry w skali szarości dla obrazka o wymiarach 28x28 pikseli. Zbiór testowy w przeciwieństwie do treningowego nie zawiera informacji o reprezentowanej klasie. Zbiór treningowy i testowy zawierają odpowiednio 42,000 i 28,000 elementów.

Z powodu braku skojarzonych odpowiedzi dla danych ze zbioru testowego podjęta została decyzja o utworzeniu zbioru walidacyjnego na podstawie danych treningowych. Ostatecznie do szkolenia klasyfikatorów użyto 29,400 elementów a do testowania skuteczności 12,600 (czyli podział zbioru treningowego był w proporcji 7:3).

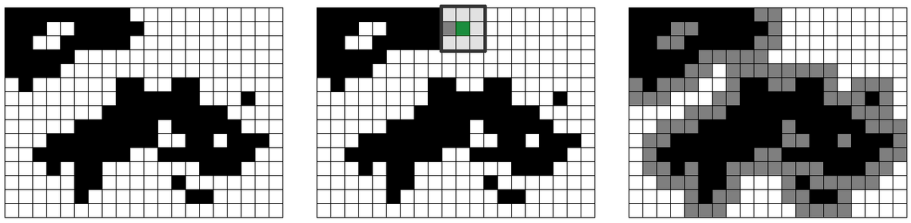
4 Operacje graficzne

Jednym z założeń dotyczącym zbioru danych było podjęcie próby zmiany wektora cech poprzez dodanie do niego nowych wartości. Zaproponowane zostały cztery nowe cechy którymi były: liczba punktów startowych cyfry, liczba punktów przecięcia cyfry, wektor przecięć cyfry (opisany dokładniej w podrozdziale 4.6) a także informacja o liczbie czarnych pikseli. Wszystkie wymienione wyżej cechy zostały wyliczone na podstawie szkieletu litery znajdującej się na obrazku, będącego wynikiem operacji zwanej szkieletyzacją.

Rysunek 1: Wczytana cyfra 7 (po lewej) po dokonaniu operacji dylatacji (po środku) a następnie szkieletyzacji (z prawej)

4.1 Dylatacja

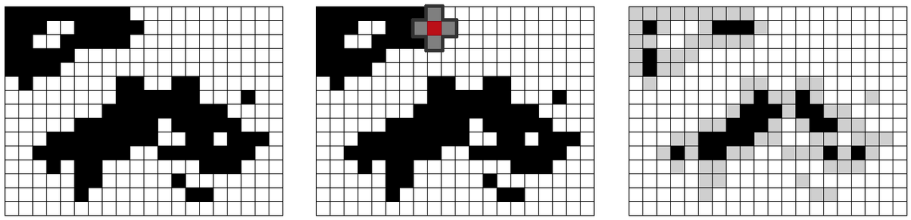
Dylatacja to operacja morfologiczna która służy do zamykania małych otworów oraz zatok we wnętrzu symbolu. Obiekty zwiększają swoją objętość i jeśli dwa lub więcej obiektów położonych jest blisko siebie, zrastają się w jedną, większą całość. Dla każdego piksela na obrazie binarnym sprawdzana jest liczba sąsiadów o wartości 1. Jeżeli wynik jest różny od zera ten piksel również zostaje "zapalony" (jego wartość ustawiona na 1).



Rysunek 2: Przykład działania algorytmu dylatacji

4.2 Erozja

Erozja jest operacją dualną do dylatacji. Jej działanie polega na obcinaniu brzegów obiektu na obrazie, powoduje zniknięcie wąskich gałęzi i małych obiektów, likwidację szumu, rozszerzenie się „dziur" w niespójnym obszarze. Dla każdego piksela jeżeli choć jeden piksel z jego sąsiedztwa ma wartość równą 0, punkt ten również zostaje "wygaszony" (jego wartość ustawiona zostaje na 0).



Rysunek 3: Przykład działania algorytmu erozji

4.3 Szkieletyzacja

Szkieletyzacja, zwana również operacją ścieniania, służy do odchudzania graficznego symbolu tak aby jako wynik otrzymać ten sam symbol narysowany linią o grubości jednego piksela. Obecnie istnieje wiele różnych algorytmów szkieletyzacji różniących się podejściem do zagadnienia, stosowalnością (bywają symbole lepiej ścieniane przez jeden algorytm podczas gdy inne mogą być przez niego odchudzane niepoprawnie) jak i złożonością. Do najbardziej znanych zaliczyć można algorytm K3M, algorytm Zhang-Suen'a, algorytm Guo-Hall'a a także algorytm KMM.

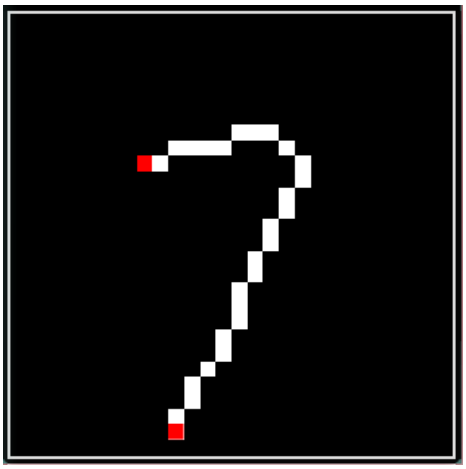
4.4 Znajdowanie punktów startowych cyfry [1]

Zazwyczaj pisząc symbole rozpoczynamy i kończymy tę czynność w pewnych szczególnych miejscach. Są to najczęściej niepołączone zakończenia linii które zmuszają do oderwania pióra. Punkty te nazywane są punktami startowymi (chociaż równie dobrze mogłyby nazywać się punktami końcowymi) a ich liczba pomaga w identyfikacji narysowanego symbolu.

4.5 Znajdowanie przecięć w cyfrze [2]

Punktami przecięcia w narysowanym symbolu nazywane są te miejsca w których następuje rozwidlenie ścieżek. Przykładem może być daszek litery T która swój punkt przecięcia posiada w miejscu złączenia daszka i nogi litery. Stosując algorytm opisany w artykule zaimplementowana została funkcjonalność liczenia punktów przecięć w cyfrach.

[1] Na podstawie artykułu: https://arxiv.org/pdf/1202.3884.pdf
[2] Na podstawie artykułu: https://arxiv.org/pdf/1202.3884.pdf



Rysunek 4: Wczytana cyfra 7 z zaznaczonymi na czerwono znalezionymi punktami startowymi

4.6 Wektor przecięć cyfry

Wektor przecięć cyfry zawiera sześć elementów. Przechowuje on wartości informujące o liczbie przecięć narysowanej cyfry z liniami prostymi przechodzącymi przez obraz w określonych miejscach, to znaczy w 30% 50% i 70% jego szerokości i wysokości.

5 Opis wykorzystanych klasyfikatorów

5.1 Backpropagation

Algorytm wstecznej propagacji błędu jest algorytmem uczenia dla wielowarstwowych sieci neuronowych. Architektura sieci jest zadana z góry, przed przystąpieniem procesu uczenia ustalona jest liczba warstw ukrytych, neuronów w każdej warstwie, neuronów wejściowych i wyjściowych. Główną zaletą algorytmu jest brak konieczności ustalania wag połączeń między neuronami, które są dobierane za pomocą metody obliczeniowej zwanej wsteczną propagacją błędu. Jest to metoda umożliwiająca modyfikację wag w sieci o architekturze warstwowej, we wszystkich jej warstwach.

Ogólny schemat procesu trenowania sieci wygląda następująco:

1. Ustalenie topologii sieci
2. Zainicjowanie wag w sposób losowy
3. Obliczenie odpowiedzi sieci dla danego wektora uczącego (warstwa po warstwie)

4. Każdy neuron wyjściowy oblicza swój błąd, oparty na różnicy pomiędzy obliczoną odpowiedzią oraz poprawną odpowiedzią

5. Błędy propagowane są do wcześniejszych warstw

6. Każdy neuron (również w warstwach ukrytych) modyfikuje wagi na podstawie wartości błędu i wielkości przetwarzanych w tym kroku sygnałów

7. Następuje powtórzenie kroków począwszy od punktu 3. dla kolejnych wektorów uczących. Gdy wszystkie wektory zostaną użyte, losowo zmienia się ich kolejność i wykorzystuje się je powtórnie

8. Algorytm zatrzymuje się, gdy średni błąd na danych treningowych osiągnie zadaną wartość (zmaleje wystarczająco)

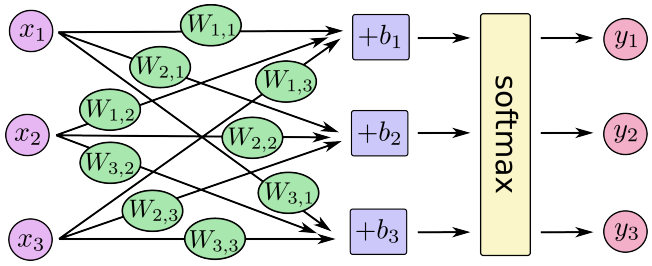
Podczas obliczeń utworzona została sieć typu Backpropagation zawierająca trzy warstwy ukryte. Warstwa wejściowa zawierała liczbę neuronów zgodnych z podawanym wejściem, kolejna warstwa liczyła $\frac{wejcie}{2}$ neuronów, następna $\frac{wejcie}{4}$ a ostatnia 10 neuronów (po jednym dla każdej klasy).

5.2 SoftMax

Softmax jest uogólnieniem regresji logistycznej do przypadku, gdy rozważa się wiele klas. W regresji logistycznej zakłada się, że etykiety są binarne : $y(i) \in \{0, 1\}$. Metoda Softmax pozwala obsługiwać $y(i) \in \{1, ..., K\}$ gdzie K jest liczbą klas. W przygotowanym rozwiązaniu metoda SoftMax posłużyła jako funkcja aktywacji neuronów. Ogólny wzór obliczeń realizowanych przez sieć można przedstawić jako

$$y = softmax(W * x + b)$$

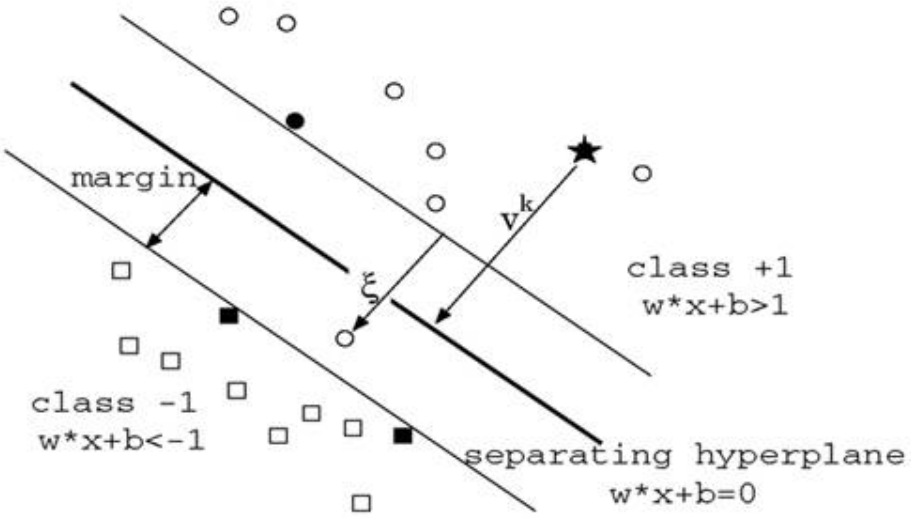
gdzie y to wyjście (odpowiedź udzielona przez sieć), W to wagi połączeń między neuronami, x to poszczególne wejścia sieci a b to bias. Wizualizacja sieci została przedstawiona na Rysunku 5.2.



Rysunek 5: Wizualizacja uproszczonej sieci typu SoftMax

5.3 SVM

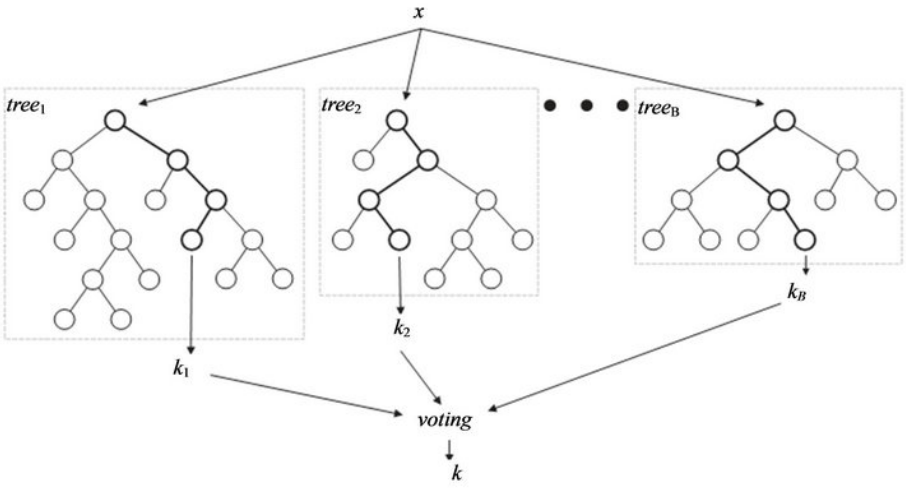
Metoda maszyny wektorów podpierających (Support Vector Machines) jest jedną z metod uczenia nadzorowanego. Klasyfikator ten znajduje hiperpłaszczyznę rozdzielającą dane treningowe na dwie klasy w ten sposób, że maksymalizuje wartość marginesu geometrycznego dla wszystkich punktów treningowych. Marginesem geometrycznym hiperpłaszczyzny H jest jej odległość do najbliższych punktów. Punkty położone najbliżej hiperpłaszczyzny nazywane są wektorami wspierającymi (ang. support vectors).



Rysunek 6: Schemat działania maszyny wektorów podpierających

5.4 Lasy losowe

Metoda lasów losowych opiera się na drzewach decyzyjnych, zwanych również drzewami klasyfikacyjnymi. Opierają się one o drzewiastą strukturę, w której węzły wewnętrzne zawierają testy na wartościach atrybutów. Z każdego węzła wewnętrznego wychodzi tyle gałęzi, ile jest możliwych wyników testu w tym węźle. Klasyfikacja zaczyna się w korzeniu drzewa, a kończy po osiągnięciu jednej z klas terminalnych, czyli liścia.
Lasy losowe (ang. random forests) będące uogólnieniem idei drzew decyzyjnych zalicza się do procedur agregujących (ang. ensemble method), w których rolę modelu klasyfikacyjnego pełni grupa modeli składowych. Działanie lasów losowych polega na klasyfikacji za pomocą grupy drzew decyzyjnych. Końcowa decyzja jest podejmowana w wyniku głosowania większościowego nad klasami wskazanymi przez poszczególne drzewa decyzyjne.



Rysunek 7: Schemat działania lasów losowych

5.5 kNN

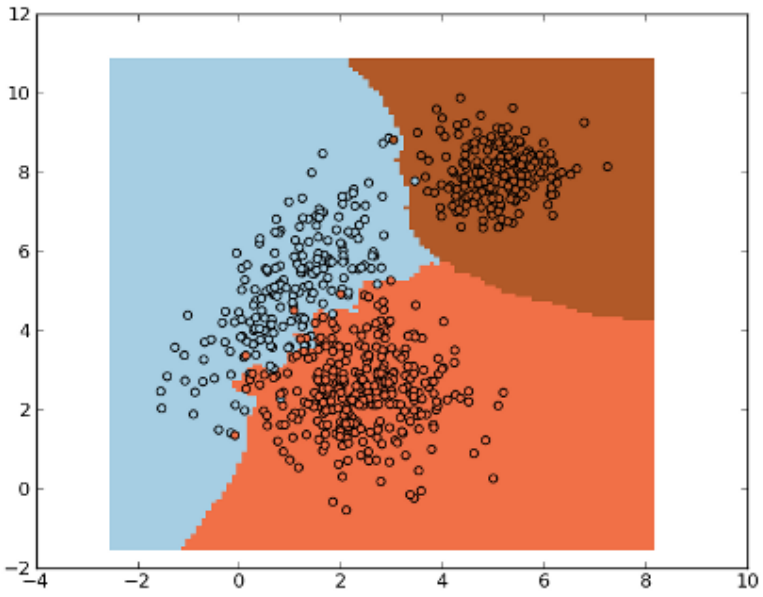
Algorytm k najbliższych sąsiadów (ang. k Nearest Neighbours) należy do grupy algorytmów opartych o analizę przypadku. Algorytmy te prezentują swoją wiedzę o świecie w postaci zbioru przypadków lub doświadczeń. Idea klasyfikacji polega na metodach wyszukiwania tych zgromadzonych przypadków, które mogą być zastosowane do klasyfikacji nowych sytuacji. Klasyfikacja nowych przypadków zgodnie z algorytmem kNN jest realizowana na bieżąco, tzn. wtedy gdy pojawia się potrzeba klasyfikacji nowego przypadku, metoda ta nie buduje klasyfikatora.
Obiekty są analizowane w ten sposób , że oblicza się odległości bądź podobieństwa między nimi. Istnieją różne miary podobieństwa czy odległości. Powinny być one wybierane konkretnie dla typu danych analizowanych, ponieważ inne są miary typowo dla danych binarnych, inne dla danych nominalnych a inne dla danych numerycznych.
Dany jest zbiór uczący zawierający obserwacje z których każda ma przypisany wektor zmiennych objaśniających oraz wartość zmiennej objaśnianej Y oraz dana jest obserwacja C z przypisanym wektorem zmiennych objaśniających dla której chcemy prognozować wartość zmiennej objaśnianej Y.
Dla tak zdefiniowanego zadania przebieg algorytmu wygląda następująco:

1. Porównanie wartości zmiennych objaśniających dla obserwacji C z wartościami tych zmiennych dla każdej obserwacji w zbiorze uczącym

2. Wybór k (ustalona z góry liczba) najbliższych do C obserwacji ze zbioru uczącego

3. Uśrednienie wartości zmiennej objaśnianej dla wybranych obserwacji, w

wyniku czego uzyskiwana jest prognoza



Rysunek 8: Schemat działania metody k najbliższych sąsiadów

6 Opis wyników

Wyniki zaprezentowane w tym rozdziale zostały uzyskane dla dwóch sposobów otrzymywania wektorów cech. Pierwszy z nich zakładał wykorzystanie obrazów dostępnych w ramach zbiorów testowych i traktowanie ich jako jedynego wejścia dla klasyfikatorów. Drugi sposób rozszerzał wektor o zbiór wyekstrahowanych cech, które zostały opisane w rozdziale 4. Dla obu podejść dokonano pełnych testów dla każdego z przygotowanych klasyfikatorów. Poniżej zamieszczone zostały tabelki zawierające wyniki. Każda kolumna oznaczona cyfrą oznacza wyniki dla obiektów z danych testowych reprezentujących tę cyfrę. Wartości w komórkach oznaczają jakość klasyfikacji dla poszczególnych klasyfikatorów (ich skuteczność). Wartości w kolumnie oznaczonej jako *error* przestawiają ogólny procent błędu dla każdego z klasyfikatorów. Wiersz *summary* zawiera wartości oznaczające skuteczność rozpoznawania każdej z cyfr przez klasyfikatory łącznie.

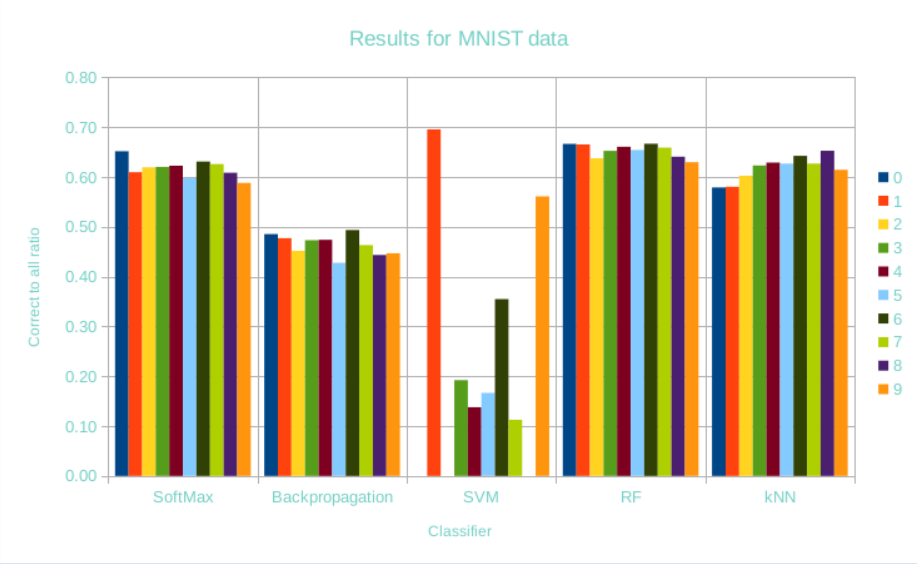
Tabela 1: Wyniki dla danych z podstawowym wektorem cech

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**error**
SoftMax	0.86	0.83	0.84	0.83	0.81	0.80	0.85	0.84	0.80	0.81	0.17
Backpropagation	0.83	0.85	0.80	0.82	0.80	0.79	0.83	0.82	0.79	0.79	0.19
SVM	0.20	0.92	0.00	0.52	0.45	0.00	0.68	0.12	0.00	0.81	0.80
RF	0.89	0.91	0.86	0.88	0.88	0.88	0.90	0.89	0.87	0.87	0.12
kNN	0.88	0.86	0.89	0.88	0.89	0.86	0.89	0.87	0.90	0.87	0.12
summary	0.73	0.87	0.68	0.79	0.77	0.67	0.83	0.71	0.67	0.83	

Tabela 2: Wyniki dla danych z rozszerzonym wektorem cech

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**error**
SoftMax	0.65	0.61	0.62	0.62	0.62	0.60	0.63	0.63	0.61	0.59	0.38
Backpropagation	0.49	0.48	0.45	0.47	0.47	0.43	0.49	0.46	0.44	0.45	0.54
SVM	0.00	0.70	0.00	0.19	0.14	0.17	0.35	0.11	0.00	0.56	0.78
RF	0.67	0.67	0.64	0.65	0.66	0.65	0.67	0.66	0.64	0.63	0.35
kNN	0.58	0.58	0.60	0.62	0.63	0.63	0.64	0.63	0.65	0.61	0.38
summary	0.48	0.61	0.46	0.51	0.50	0.49	0.56	0.50	0.47	0.57	

Rysunek 9: Wyniki dla danych z podstawowym wektorem cech przedstawione na wykresie



Rysunek 10: Wyniki dla danych z rozszerzonym wektorem cech przedstawione na wykresie

7 Podsumowanie

Wbrew oczekiwaniom rozszerzenie wektora cech wcale nie poprawiło jakości klasyfikacji. Wręcz przeciwnie, wyniki otrzymane dla dodatkowych danych są o wiele gorsze niż te dla samego obrazu. Podczas inspekcji zauważono, że skany cyfr dostarczane jako dane testowe nie poddają się łatwej obróbce graficznej. Próby ustalenia wspólnej maski dla operacji dylatacji i erozji nie przyniosły oczekiwanych rezultatów. Niezależnie od przyjętych wartości zawsze wykonanie którejś z czynności morfologicznej powodowało utratę danych na pewnych obrazach. To z kolei przyczyniło się do uzyskania miernych wyników.

Wydaje się rozsądnym podjęcie próby poszukiwania takich cech które mogłyby być wyliczane na podstawie zeskanowanych dokumentów bez potrzeby ich modyfikacji. Zgodnie z opisami podejść do zagadnienia klasyfikacji ze strony http://yann.lecun.com/exdb/mnist/ większość rozwiązań nie zawierała operacji na obrazach. Więcej uwagi poświęcono samym klasyfikatorom, ich parametrom i ewentualnym modyfikacjom. Biorąc pod uwagę otrzymane wyniki uważamy, że jest to słuszne podejście.

Literatura

[1] LeCun, Y., Cortes, C., and Burges, C., *The MNIST database of handwritten digits*, in: http://yann.lecun.com/exdb/mnist.

[2] Breiman, L., *Random Forests*. Machine Learning 45 (1), 2001

[3] Cortes, C., Vapnik, V., *Support-vector networks*. Machine Learning 20 (3), 1995.

[4] Altman N. S., *An introduction to kernel and nearest-neighbor nonparametric regression*. The American Statistician 46 (3), 1992.

[5] Scholkopf, B., Williamson, R., Smola, A., Shawe-Taylort, J., Platt, J., *Support Vector Method for Novelty Detection*, Advances in Neural Information Processing Systems 12, 1992.

[6] Wang, Y., Casasent, D., *A Support Vector Hierarchical Method for multi-class classification and rejection*, Proc. of Int. Joint Conf. on Neural Networks, 2009.

[7] Dinesh Dileep *A feature extraction technique based on character geometry for character recognition*, https://arxiv.org/pdf/1202.3884.pdf

[8] http://scikit-learn.org/stable/modules/grid_search.html