testerzy.pl

Uczenie sztucznych sieci neuronowych a testowanie

Uczenie sieci neuronowej to proces zbliżony do testowania, a testy wykonywane są dynamicznie. Oczywiście jeśli sieć neuronową potraktujemy jak oprogramowanie.

1. Charakterystyka sztucznych sieci neuronowych

Sieć neuronowa to ogólna nazwa struktur matematycznych i ich programowych lub sprzętowych modeli, realizujących obliczenia lub przetwarzanie sygnałów poprzez rzędy elementów, zwanych sztucznymi neuronami, wykonujących pewną podstawową operację na swoim wejściu. Oryginalną inspiracją takiej struktury była budowa naturalnych neuronów, łączących je synaps, oraz układów nerwowych, w szczególności mózgu.

Czasami nazwą **sztuczne sieci neuronowe** określa się interdyscyplinarną dziedzinę wiedzy zajmującą się konstrukcją, trenowaniem i badaniem możliwości tego rodzaju sieci.

Współcześnie nie ma wątpliwości, że sztuczne sieci neuronowe nie stanowią dobrego modelu mózgu, choć różne ich postacie wykazują cechy charakterystyczne dla biologicznych układów neuronowych: zdolność do uogólniania wiedzy, uaktualniania kosztem wcześniej poznanych wzorców, dawanie mylnych odpowiedzi po przepełnieniu· Mimo uproszczonej budowy sztuczne sieci neuronowe stosuje się czasami do modelowania schorzeń mózgu.

Sztuczne sieci neuronowe znajdują zastosowanie w rozpoznawaniu i klasyfikacji wzorców (przydzielaniu wzorcom kategorii), predykcji szeregów czasowych, analizie danych statystycznych, odszumiania i kompresji obrazu i dźwięku oraz w zagadnieniach sterowania i automatyzacji.

Ogólnie wyróżnia się dwa typy architektury sztucznych sieci neuronowych:

a) sieci jednokierunkowe, czyli sieci o jednym kierunku przepływu sygnałów; szczególnym przypadkiem architektury jednokierunkowej jest sieć warstwowa, reprezentująca zdecydowanie najpopularniejszą topologię;

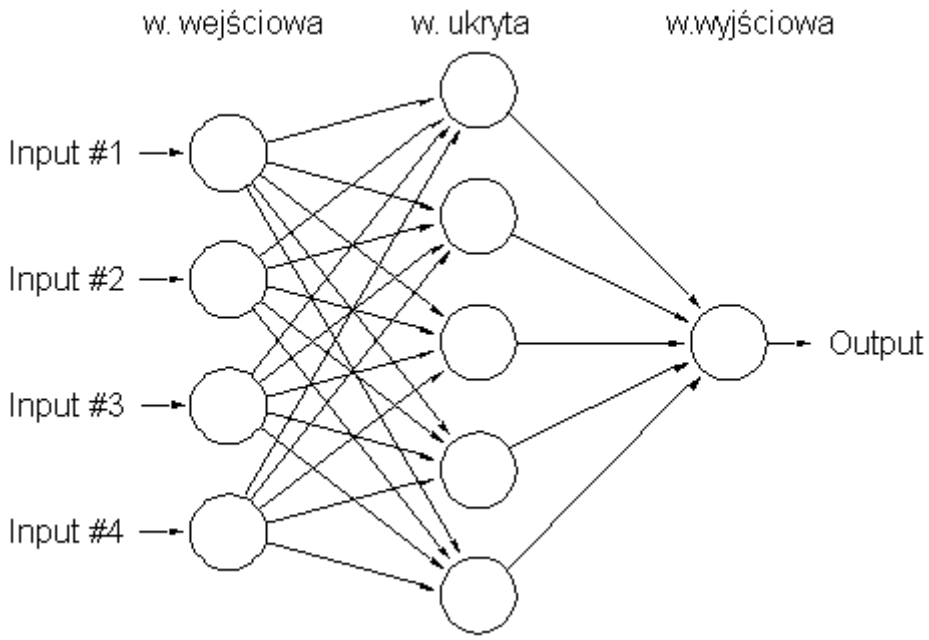
b) inne, np. sieci rekurencyjne (feedback, bidirectional) tj. sieci ze sprzężeniami zwrotnymi (sieć Hopfielda) albo sieci uczenia się przez współzawodnictwo (Kohonena).

2. Ogólna postać sieci neuronowej

Sieć neuronowa to zbiór połączonych ze sobą jednostek wejściwo-wyjściowych. Z każdym połączeniem skojarzona jest waga, która może zostać zmieniona w trakcie uczenia. Dowolna sztuczna sieć neuronowa może być zdefiniowana poprzez określenie:

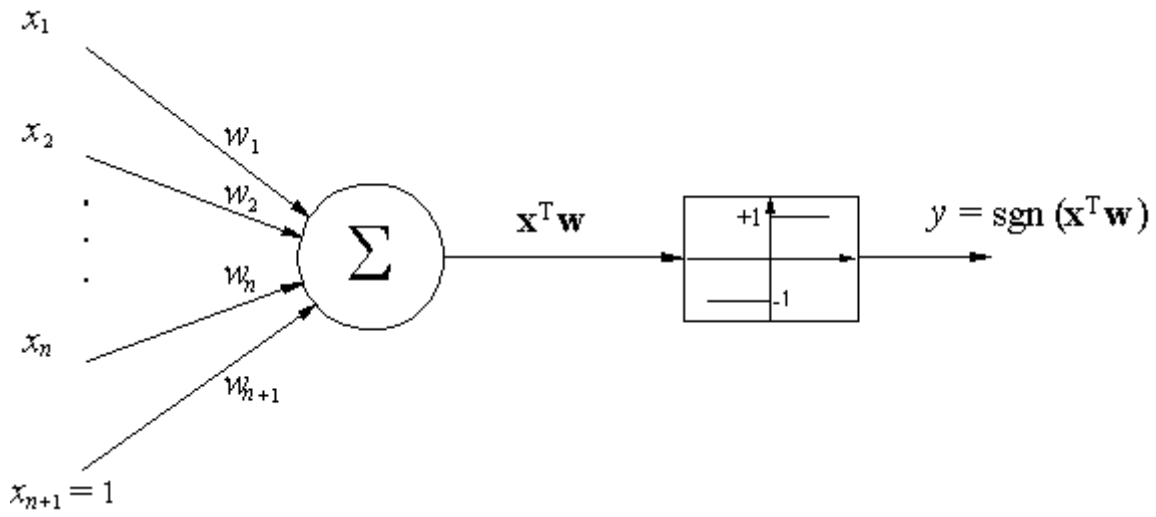
a) modelu sztucznego neuronu,
b) topologii,
c) reguły uczenia sieci.

Ogólna postać sieci neuronowej (w przypadku sieci wielowarstwowej) może wyglądać np. tak:



Na sieć można spojrzeć również jak na **czarną skrzynkę**, która dostaje sygnały na wejścia i zwraca konkretny wynik na wyjściu, i takie podejście będzie rozważane w dalszej części artykułu.

Sztuczny neuron (perceptron) reprezentowany jest na ogół tak:



Znak Sigmy to miejsce, w którym współrzędne wejściowe są mnożone przez wektor wag w postaci iloczynu skalarnego. Następnie na podstawie wartości tego iloczynu sztuczny neuron zostaje aktywowany albo nie.

2. Proces uczenia się sztucznych siei neuronowych

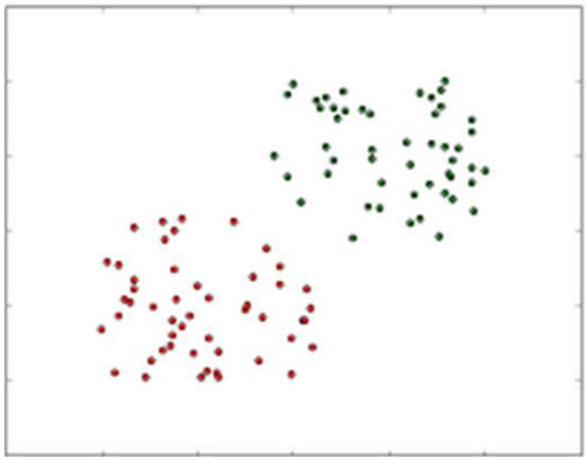
Wyróżnia się dwa podstawowe sposoby uczenia sieci:

a) uczenie nadzorowane (ang. *supervised learning*),
b) uczenie nienadzorowane (ang. *unsupervised learning*).

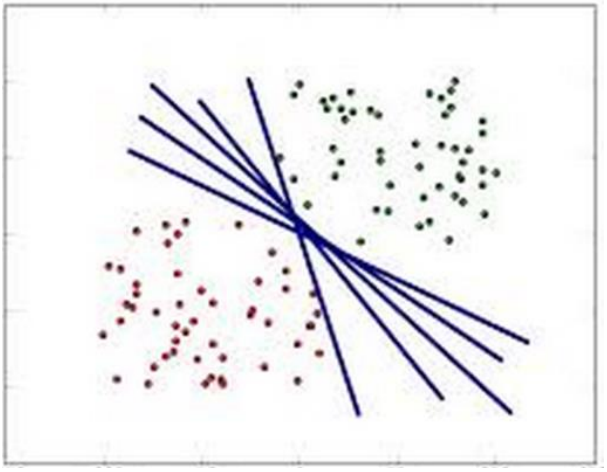
Uczenie nadzorowane – dany jest zbiór przykładów uczących składający się z par wejście-wyjście (x_j, z_j), gdzie z_j jest pożądaną odpowiedzią sieci na sygnały wejściowe x_j, $(j = 1, .. m)$. Zadaniem sieci jest nauczyć się możliwie jak najdokładniej funkcji przybliżającej powiązanie wejścia z wyjściem. Uczenie nienadzorowane polega na tym, że nie ma wartości z_j. Te wartości musi sieć znaleźć sobie sama. Elementy x_j – ***dane uczące*** - to wektory mogące zawierać dowolną skończoną ilość współrzędnych. Natomiast wartości , z_j to na ogół wartości skalarne (rzeczywiste). Natomiast dane, na których sprawdza się dokładność nauczenia się sieci, to **dane testowe.**

Algorytmów uczenia sieci neuronowych jest bardzo wiele. Na ten temat można pisać obszerne książki. Ale ciekawy jest fakt, że sieć neuronowa nie wymaga programowania, tylko wymaga „dopasowania" parametrów bądź swoich cech do zestawu danych uczących w taki sposób, aby jej dokładność była jak największa (na ogół) w przypadku gdy już została nauczona i nie powinna się mylić. Tak więc uczenie sztucznej sieci neuronowej polega na **testowaniu konfiguracji parametrów sieci i jej cech w kontekście konkretnych danych uczących** dotąd, aż osoba testująca uzyska jak najlepszy, postawiony przez siebie cel (np. dokładność).

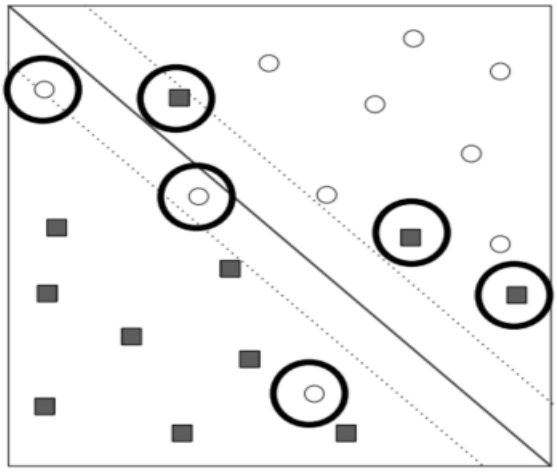
W przypadku omawianego wyżej perceptronu uczenie polega na tym, aby dobrać taki wektor wag, który po przemnożeniu przez daną testową odpowiednio będzie aktywował lub nie będzie aktywował wyjścia tego perceptronu. Graficznie wygląda to tak, że mamy dwa rodzaje klas i chcemy, żeby sieć (perceptron) nauczyła się rozpoznawać przynależność czerwonych lub zielonych punktów do swojej klasy:



Wobec tego sieć zostaje poddana uczeniu i znajdzie wektor wag taki, że będzie możliwa do przedstawienia linia (hiperpłaszczyzna w większych wymiarach), która oddzieli te klasy:



Wektor wag jest wtedy prostopadły do takiej płaszczyzny separującej. Jednak nie zawsze dane muszą być takie, że łatwo je oddzielić, np. tak jak poniżej.



W takim przypadku stosuje się inne metody, które rozwiązują tego typu problemy. Stąd też mnogość rodzajów sieci neuronowych.

Teoretycznie nauczona sieć neuronowa powinna zadziałać podobnie do znanych wzorców, jak np. lekarz podaje dane medyczne chorego na raka pacjenta na wejście, a sieć zwraca wynik, że pacjent jest rzeczywiście chory. Jeśli sieć na 100 prób pomyli się przykładowo 30 razy, to znaczy, że nie jest najlepiej wyuczona. Najlepiej, gdyby nie myliła się w ogóle.

Sieci neuronowe mogą być stosowane, z dużym prawdopodobieństwem odniesienia sukcesu, wszędzie tam, gdzie pojawiają się problemy związane z tworzeniem modeli matematycznych pozwalających odwzorowywać złożone zależności pomiędzy pewnymi sygnałami wejściowymi a wybranymi sygnałami wyjściowymi. Nie powinno się stosować sieci neuronowych w takich przypadkach, w których należy otrzymać jasny i pewny wynik.

3. Cechy wspólne sztucznych sieci neuronowych

Każda sieć ma swoje pewne własności, którymi różni się od pozostałych. Są jednak cechy, które posiada każda sieć:

a) liczba danych uczących,
b) liczba danych testowych,
c) liczba współrzędnych branych pod uwagę dla danych uczących (ponieważ nie wszystkie sygnały wejściowe muszą być przydatne w przypadku uczenia sieci),
d) dane uczące mogą zostać znormalizowane lub poddane „obróbce” metodami statystyki.

Rozważyć można pewien przykład na podstawie wyżej wymienionych podpunktów: nie znając własności sieci neuronowej zaprojektować należy dla niej testy i obserwować, jak przebiega proces uczenia. Testy przedstawić można w postać tabel:

Tabela 1 - Przykładowe testy dla sztucznej sieci neuronowej (1)

Liczba % danych uczących	Liczba % danych testowych	Liczba współrzędnych na wejściu	Normalizacja
30	70	15	Nie
40	60	15	Nie
50	50	15	Nie
60	40	15	Nie
70	30	15	Nie
80	20	15	Nie
90	10	15	Nie

Tabela 2 - Przykładowe testy dla sztucznej sieci neuronowej (2)

Liczba % danych uczących	Liczba % danych testowych	Liczba współrzędnych na wejściu	Normalizacja
30	70	15	Tak
40	60	15	Tak
50	50	15	Tak
60	40	15	Tak
70	30	15	Tak
80	20	15	Tak
90	10	15	Tak

Po takich doświadczeniach można zobaczyć, czy normalizacja ma wpływ na dokładność uczenia się sieci neuronowej. Jeśli okaże się, że tak, to nie ma sensu robić testów dla danych nieznormalizowanych. Również im większa jest liczba danych uczących, tym sieć zachowuje się lepiej. Wobec tego wystarczy już teraz zmieniać liczbę współrzędnych na wejściu przy takich ustawieniach jak niżej. Przykładowo:

Tabela 3 - Przykładowe testy dla sztucznej sieci neuronowej (3)

Liczba % danych uczących	Liczba % danych testowych	Liczba współrzędnych na wejściu	Normalizacja
70	30	18	Tak
80	20	19	Tak
90	10	20	Tak

Tabela 4 - Przykładowe testy dla sztucznej sieci neuronowej (4)

Liczba % danych uczących	Liczba % danych testowych	Liczba współrzędnych na wejściu	Normalizacja
70	30	21	Tak
80	20	22	Tak
90	10	23	Tak

Obserwując uczenie się sieci w końcu użytkownik dojdzie do takiej konfiguracji poprzez redukcję, dla której sieć nauczy się najlepiej (np. najdokładniej). Dobrze jest mieć

doświadczenie w pracy z sieciami neuronowymi, ponieważ osoba doświadczona szybciej znajdzie odpowiednią konfigurację sieci niż osoba początkująca. Nie jest to jednak proste i jest czasochłonne. W przykładzie zostały pokazane tylko cztery parametry. Gdy okaże się, że musimy eksperymentować na zmianie cech wewnętrznych sieci, to testów pojawi się o wiele więcej, zanim użytkownik wytrenuje się tak, aby spełniała postawiony cel.

Istnieje sporo narzędzi, które posiadają zaimplementowane sztuczne sieci neuronowe. Wiele z nich jest niekomercyjnych. Program Matlab preferowany prze uczelnie wyższe również umożliwia zaimplementowanie testów sztucznych sieci neuronowych.

4. Uczenie (testy) sieci w praktyce

Załóżmy, że chcemy sprawdzić, jak sieć wielowarstwowa MLP nauczy się oceniać, czy pacjent może dostać zawał serca czy też nie. Jest to jeden z przykładów zastosowania sieci wielowarstwowej w diagnostyce medycznej. Nie będziemy zajmować się opisem działania sieci MLP oraz jej parametrami, ponieważ nie to jest tematem pracy. Interesują nas same testy. Opis działania sieci MLP znajduje się w literaturze wymienionej w bibliografii.

W skład poszczególnych danych (wejść) należą wiek, płeć, rodzaj bólu klatki piersiowej oraz ciśnienie krwi w spoczynku. Na podstawie tych danych należy ocenić ryzyko dostania zawału serca. Dane wejściowe posiadają 20 zmiennych (wiek (dwie), płeć (dwie), rodzaj bólu klatki piersiowej (siedem), ciśnienie krwi w spoczynku (osiem)). Mamy 270 danych. Została wykonana analiza dla różnych podziałów danych na uczące i testujące w stosunku 50/50, 60/40, 70/30, 80/20, 90/10 dla trzech i czterech warstw z kolejno jedną i dwoma ukrytymi. W celu obliczenia dokładności. Ponadto została wykonana walidacja 5- i 10-częściowa krzyżowa. Ostatecznym celem jest wybranie najlepszego modelu. Model ten wybierany jest na podstawie analizy dokładności wszystkich przypadków, policzenia z nich średniej i wybraniu modelu najbliższemu tej średniej.

Zbiorcze Wyniki:

a) 3 warstwy (1 ukryta)

1. 3 warstwy (1 ukryta) 50/50 - 80.00,
2. 3 warstwy (1 ukryta) 60/40 - 71.30,
3. 3 warstwy (1 ukryta) 70/30 - 77.78,
4. 3 warstwy (1 ukryta) 80/20 - 66.67,
5. 3 warstwy (1 ukryta) 90/10 - 77.78,
6. 3 warstwy (1 ukryta) V-fold 10 - 74.44,
7. 3 warstwy (1 ukryta) V-fold 5 - 76.67,
8. 3 warstwy (1 ukryta) LOO - 74.44,

Średnia arytmetyczna dokładności = 74.885
Odchylenie standardowe $S \approx 4.25686$

b) 4 warstwy (2 ukryte)

9. 4 warstwy (2 ukryte) 50/50 - 76.30,
10. 4 warstwy (2 ukryte) 60/40 - 71.30,
11. 4 warstwy (2 ukryte) 70/30 - 79.01,
12. 4 warstwy (2 ukryte) 80/20 - 66.67,
13. 4 warstwy (2 ukryte) 90/10 - 81.48,
14. 4 warstwy (2 ukryte) V-fold 10 - 75.56,
15. 4 warstwy (2 ukryte) V-fold 5 - 75.56,
16. 4 warstwy (2 ukryte) LOO - 75.56

Średnia arytmetyczna dokładności = 75.18
Odchylenie standardowe $S \approx 4.52802$

Ostatnia kolumna to dokładność. Najlepszy model wybrany zostanie na podstawie analizy dokładności wszystkich przypadków, policzenia z nich średniej i wybraniu modelu najbliższemu tej średniej:

Średnia arytmetyczna = 75.0325
Odchylenie standardowe $S \approx 4.24826$

5. Wnioski z artykułu

Celem napisania pracy było uświadomienie czytelnikowi, że próba nauczenia sieci neuronowej to nic innego jak **testowanie.** Nie jest to testowanie takie jak w przypadku zastosowanie konkretnej technologii wytwarzania oprogramowania. Tutaj charakter tych testów jest nieco inny. Testy powstają dynamicznie i poprzez obserwację. Dzieje się tak dlatego, że nie da się przewidzieć jakie dostaniemy dane, jak one będą wpływać na uczenie sieci oraz nie da się napisać przypadków testowych do uczenia sieci.

Zdaniem autora, próba wytrenowania sieci neuronowej to pewien rodzaj dynamicznych testów oprogramowania, ponieważ sztuczna sieć neuronowa to zaimplementowany program.

6. Bibliografia:

1. Kosiński Robert, *Sztuczne Sieci Neuronowe*, Wydawnictwa Naukowo-Techniczne, 2008.
2. Osowski Stanisław, *Sieci neuronowe do przetwarzania informacji*, Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa, 2000.

Autor

Marek Żukowicz jest absolwentem matematyki na Uniwersytecie Rzeszowskim. Obecnie pracuje jako tester. Jego zainteresowania skupiają się wokół testowania, matematyki, zastosowania algorytmów ewolucyjnych oraz zastosowania matematyki w procesie testowania. Interesuje się również muzyką, grą na akordeonach oraz na perkusji.