Uniwersytet Łódzki

Wydział Matematyki i Informatyki

Temat ćwiczenia:

Zastosowanie sieci o radialnych funkcjach bazowych do prognozowania dobowego przebiegu zapotrzebowania na energię elektryczną

Sprawozdanie z przedmiotu: ***Zaawansowane Metody Eksploracji***

Studia stacjonarne 2-go stopnia

Kierunek: *Analiza Danych, sem. 1*

Prowadzący: *prof. dr hab. inż. Grzegorz Dudek*

|  |  |
| --- | --- |
| Data wykonania ćwiczenia: | 21.09.2023, 28.09.2023 |
| Data złożenia sprawozdania: |  |
| Data zaliczenia: |  |

|  |  |
| --- | --- |
| Wykonali: | Ocena: |
| **Jan Kowalski** |  |
| Stefan Nowak |  |

Nr sekcji: **7.2**

**A. Cel ćwiczenia**

Zapoznanie się z budową, funkcjonowaniem i metodą uczenia sieci neuronowej o radialnych funkcjach bazowych. Zwrócenie uwagi na istotę wstępnego przetworzenia danych i selekcji wejść. Zapoznanie się z problematyką prognozowania procesów wykazujących cechy quasi-periodyczności.

**B. Treść zadania**

Zaprojektuj sieć o radialnych funkcjach bazowych do prognozowania dobowego przebiegu obciążenia systemu elektroenergetycznego (SEE). Dane są przebiegi dobowe (godzina po godzinie) obciążenia SEE z okresu trzech lat. Określ średnie bezwzględne błędy procentowe prognoz, a dla najlepszego wariantu sieci także bezwzględne błędy procentowe w rozbiciu na dni tygodnia oraz godziny doby.

Z wykorzystaniem programu RBF\_pro przeprowadź eksperymenty numeryczne (trening i test sieci) dla następujących wariantów:

1. przyjmujemy różną liczbę neuronów w warstwie ukrytej – K = 50, 60, ..., 300, i stałą wartość rozpiętości funkcji radialnych σ = 1,
2. dla optymalnej liczby neuronów K z wariantu z p. 1 przyjmujemy różne wartość rozpiętości σ = 0.6, 0.7, ..., 1.3.

**C. Budowa i działanie sieci neuronowej o radialnych funkcjach bazowych**

Sieci o radialnych funkcjach bazowych (*radial basis function network - RBF*) składają się z trzech warstw (rys. 1).



Rys.1. Architektura sieci RBF.

Neurony w warstwie ukrytej stanowią bazę dla wektorów wejściowych **x**, odwzorowując je nieliniowo. Każdy neuron tej warstwy realizuje funkcję zmieniającą się radialnie wokół centrum **C**i, np. funkcję Gaussa:

, (1)

w której: ||.|| - norma euklidesowa, σi– parametr decydujący o kształcie (szerokości) funkcji radialnej.

Neurony warstwy wyjściowej realizują superpozycję sygnałów wychodzących z neuronów ukrytych. Wyjście sieci jest liniową kombinacją sygnałów warstwy ukrytej. Funkcję jaką realizuje sieć można zapisać wzorem:

, (2)

w którym: i – numer neuronu ukrytego, K – liczba neuronów ukrytych, **w**i – wektor wagowy łączący i-ty neuron warstwy ukrytej z warstwą wyjściową, **w**0 reprezentuje stały składnik przesunięcia funkcji.

Sieć RBF realizuje odwzorowanie zbioru wektorów wejściowych **x** w zbiór wektorów **y**d poprzez określenie funkcji aproksymującej F(**x**), czyli rozciągnięcie nad zbiorem wektorów **x** hiperpłaszczyzny wielowymiarowej dopasowującej się do wartości żądanych.

Proces uczenia sieci RBF przebiega trójetapowo: w pierwszym etapie należy ustalić liczbę neuronów w warstwie ukrytej oraz dobrać centra funkcji radialnych korzystając z dowolnego algorytmu klasteryzacji, w drugim etapie wyznacza się szerokości pól recepcyjnych każdego neuronu ukrytego i w etapie trzecim wyznacza się wagi powiązań między warstwą ukrytą i wyjściową. Przy uczeniu sieci korzysta się z informacji zawartych w zbiorze treningowym.

Dobór wag powiązań warstwy ukrytej z warstwą wyjściową sprowadza się do rozwiązania równania o postaci:

. (3)

Rozwiązanie, minimalizujące kwadrat normy ||**y**d– **wG**’|| (**y**d– macierz pożądanych odpowiedzi dla wszystkich wektorów uczących, **G**’ – macierz odpowiedzi warstwy ukrytej na wszystkie wektory uczące, rozszerzona o wiersz jedynek – składnik stały przesunięcia funkcji, **w** – poszukiwana macierz wag), można znaleźć stosując operację pseudoinwersji macierzy prostokątnej **G**’:

. (4)

Do określenia centrów funkcji radialnych użyto algorytmu aglomeracyjnego [Duda73]. Na starcie algorytmu przyjmuje się, że każdy wektor ze zbioru danych stanowi centrum grupy. W kolejnych krokach algorytmu znajduje się dwa najbliższe sobie centra łącząc je w jedno. Po każdym kroku liczba centrów zmniejsza się o jeden. Algorytm kończy działanie, jeśli liczba centrów osiągnie założoną wartość K.

**D. Metodyka rozwiązania zadania**

Etapy rozwiązania zadania obejmują:

1. Konstrukcję zbioru treningowego i testowego.
2. Trening sieci dla różnej liczby centrów K = 50, 60, ..., 300 i stałej wartości rozpiętości funkcji radialnych σ = 1.
3. Trening sieci dla optymalnej liczby neuronów i różnej wartości rozpiętości σ = 0.6, 0.7, ..., 1.3.
4. Trening sieci dla optymalnej liczby neuronów K i rozpiętości σ celem obliczenia błędów prognoz na poszczególne dni tygodnia i godziny doby.

*D.1. Konstrukcja zbioru treningowego i testowego. Metody przetwarzania zbiorów danych*

Zbiór danych obejmuje rzeczywiste przebiegi dobowe (obciążenia systemu elektroenergetycznego w poszczególnych godzinach doby) z lat 1992, 1993 i 1994. Przyjęto, że przebiegi z lat 1992 i 1993 stanowią dane treningowe, natomiast przebiegi z 1994 – dane testowe. Zbiory wykorzystywane do treningu i testu zawierają wektory rzeczywistych obciążeń systemu w poszczególnych godzinach

, , (5)

przy czym: dla macierzy Ptrn wykorzystywanej do treningu n = 731, a dla macierzy Ptst wykorzystywanej do testu n = 365.

Wprowadza się oznaczenia Pd,h – obciążenie w godzinie h doby d, Pd – wektor obciążeń w poszczególnych godzinach doby d (Pd = [Pd,1, Pd,2, ..., Pd,24]), Ph – wektor obciążeń o godzinie h w poszczególnych dobach (Ph = [P1,h, P2,h, ..., Pn,h]).

Z macierzy P konstruuje się wektory wejściowe **x** oraz wektory pożądanych odpowiedzi **y**d. Składowymi wektorów **x** i **y**d są obciążenia systemu przetworzone poprzez standaryzację.

***Standaryzacja***

Wyznacza się wartość średnią oraz odchylenie standardowe dla wektorów Ph ze zbioru treningowego wg zależności:

 (6)

 (7)

gdzie: n – liczba wektorów w zbiorze treningowym, d – numer doby (d = 1, 2, ..., n),   
h – numer godziny (h = 1, 2, ..., 24),  - obciążenie systemu w dobie d i godzinie h.

Składowe wektorów wejściowych oraz wektorów pożądanych odpowiedzi dla zbioru treningowego i testowego otrzymujemy z zależności (dla h= 1, 2, ..., 24):

 (8)

 (9)

 (10)

 (11)

Po wytrenowaniu sieci na jej wejścia prezentuje się wektor  otrzymując na wyjściu wektor . Prognozowane wartości obciążeń w poszczególnych godzinach doby (d+1) -  wyznacza się z przekształconej zależności (11).

Sieć RBF posiada 24 wejścia, 24 wyjścia oraz K neuronów radialnych w warstwie ukrytej.

### D.2. Błędy prognozy

Do oceny układów prognostycznych stosuje się bezwzględny błąd procentowy liczony dla każdej prognozowanej wartości mocy godzinowej BBPd,h, wyznaczony dla zbioru testowego:

, (12)

przy czym  i oznaczają rzeczywiste i prognozowane obciążenie systemu o godzinie h rozpatrywanej doby d.

Średnie dobowe błędy prognozy oznaczono symbolem ŚBBPd, a średnie roczne błędy prognozy symbolem – ŚBBPr.

Wyznaczano odchylenie standardowe błędów określone wzorem:

 (13)

**E. Zestawienie wyników**

Tab. 1. ŚBBPr dla różnej liczby K neuronów w warstwie ukrytej i σ = 1.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| K | 100 | 110 | 120 | 130 | 140 | 150 | 160 | 170 | 180 | 190 | 200 |
| ŚBBPr [%] | *6,85* | *6,42* | *5,73* | *5,32* | *5,19* | *4,67* | *3,69* | *3,64* | *3,60* | *3,41* | *2,55* |
| std1 | *4,29* | *3,88* | *3,41* | *3,26* | *3,23* | *3,01* | *2,60* | *2,34* | *1,96* | *1,71* | *1,44* |
| K | 210 | 220 | 230 | 240 | 250 | 260 | 270 | 280 | 290 | 300 | 310 |
| ŚBBPr [%] | *2,53* | ***2,45*** | *3,04* | *3,90* | *4,11* | *4,69* | *5,46* | *5,77* | *5,89* | *6,68* | *7,53* |
| std1 | *1,41* | *1,33* | *1,37* | *1,72* | *2,08* | *2,44* | *2,45* | *2,56* | *2,84* | *3,31* | *3,32* |

Tab. 2. ŚBBPr dla K = 220 i różnej wartości σ.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| σ | 0,7 | 0,8 | 0,9 | 1,0 | 1,1 | 1,2 | 1,3 |
| ŚBBPr [%] | *2,51* | *2,44* | *2.49* | *2,45* | ***2,36*** | *2,56* | *2,59* |
| std1 | *1,23* | *1,36* | *1.27* | *1,33* | *1,34* | *1,46* | *1,59* |



Rys. 2. ŚBBPr w rozbiciu na dni tygodnia.



Rys. 3. ŚBBPr w rozbiciu na dni tygodnia i godziny doby.

**F. Wnioski**

1. Przebieg obciążenia SEE charakteryzuje się cyklicznością roczną i dobową. Tę właściwość należy uwzględnić w konstrukcji modeli prognostycznych, odpowiednio przetwarzając dane i przeprowadzając proces prognozy. Standaryzacja ujednolica przebiegi dobowe – po standaryzacji wartość średnia każdego przebiegu jest zerowa a odchylenia standardowe jednostkowe. Dane standaryzowane pozwoliły uzyskać znacznie lepsze rezultaty od danych nie przetworzonych, dla których zaobserwowano błąd ŚBBPr na poziomie ponad 6%.
2. Sieć o radialnych funkcjach bazowych łączy cechy sieci uczonych bez nadzoru (np. sieć Kohonena) oraz sieci uczonych z nadzorem (np. perceptron). Zastosowany algorytm uczenia sieci RBF jest trójetapowy: w pierwszym etapie należy ustalić liczbę neuronów w warstwie ukrytej oraz dobrać centra funkcji radialnych korzystając z algorytmu klasteryzacji, w drugim etapie wyznacza się szerokości pól recepcyjnych neuronów ukrytych i w etapie trzecim wyznacza się wagi powiązań między warstwą ukrytą i wyjściową. Algorytm ten jest znacznie szybszy niż algorytm wstecznej propagacji błędów w wielowarstwowym perceptronie. Trening sieci dla jej optymalnego wariantu trwał ok. 40 s (procesor 2 GHz, RAM 64 MB).
3. Najlepszy wynik – ŚBBPr = 2,36% – odnotowano dla 220 neuronów w warstwie ukrytej oraz parametru szerokości σ = 1,1. Parametr σ nie ma tak istotnego znaczenia na jakość prognoz jak liczba neuronów K.
4. Błędy prognozy zależne są od typu dnia tygodnia (rys. 2 i 3). Największą dokładność uzyskano dla środy i piątku, najmniejszą dla poniedziałku. Spowodowane jest to bardziej stabilnym przebiegiem obciążenia w dni robocze niż w dni świąteczne. Poniedziałek jest dniem, który ma bardziej nieregularny przebieg obciążenia w początkowych godzinach doby (przejście z dnia wolnego do roboczego) i w związku z tym jest trudny do prognozowania. Jednocześnie dla piątku – dnia o podobnym charakterze – obserwuje się znacznie mniejsze błędy.
5. Błąd prognozy zależny jest także od godziny doby. Obciążenia godzin wieczornych, od 9 do 21, prognozowane były z największym błędem, a prognozy dla godzin początkowych doby, od 1 do 6, były najdokładniejsze (rys. 3).
6. Prawdopodobnie dekompozycja prognozy przebiegów dobowych obciążenia i konstrukcja indywidualnych modeli prognostycznych na konkretne godziny i typ dnia pozwoliłaby zmniejszyć błędy.
7. Należy też zwrócić uwagę, że proces uczenia sieci i jej pracę w trybie odtworzeniowym zakłócają dni nietypowe – święta państwowe i religijne. Usunięcie tych dni z bazy danych uczących powinno poprawić działanie sieci – zmniejszyć błędy prognozy.

## Literatura

[Osowski96] Osowski S.: Sieci neuronowe w ujęciu algorytmicznym, Warszawa, WNT, 1996.

[Duda73] Duda R. O., Hart P. E.: Pattern Classification and Scene Analysis. New York, John Wiley&Sons, 1973.

#### G. Listing programu

% standaryzacja zbiorów danych

load trn.txt -ascii;

load tst.txt -ascii;

x\_trn1=trn(1:size(trn,1)-1,:)'; d\_trn1=trn(2:size(trn,1),:)';

x\_tst1=tst(1:size(tst,1)-1,:)'; d\_tst1=tst(2:size(tst,1),:)';

[x\_trn2,meanx\_trn,stdx\_trn]=prestd(x\_trn1);

d\_trn2=trastd(d\_trn1,meanx\_trn,stdx\_trn);

x\_tst2=trastd(x\_tst1,meanx\_trn,stdx\_trn);

d\_tst2=trastd(d\_tst1,meanx\_trn,stdx\_trn);

%szerokość funkcji radialnych

odl=dist(x\_trn2);

odl=sort(odl')';

%spread=odl(:,2); %dla 1 najbliższego sąsiada

spread=mean(odl(:,2:3)')'; %spread= średniej odległości od k najbliższych centrów

% trening sieci RBF

net=newrbe1(x\_trn2,d\_trn2,spread);

%test sieci

y=sim(net,x\_tst2);

y=poststd(y,meanx\_trn,stdx\_trn);

%błędy prognoz

BBPh\_tst=abs(y-d\_tst1)./d\_tst1.\*100; %bezwzględny błąd procentowy godzinowy

BBPd\_tst=mean(BBPh\_tst); %bezwzględny błąd procentowy dobowy

BBPs\_tst=mean(BBPd\_tst)

Odch\_std\_h=mean(std(BBPh\_tst)) %średnie dobowe odch. standardowe błędu

Odch\_std\_d=std(BBPd\_tst) %średnie roczne odch. standardowe błędu

czas=toc

y=sim(net,x\_trn2);

y=poststd(y,meanx\_trn,stdx\_trn);

BBPh\_trn=abs(y-d\_trn1)./d\_trn1.\*100; %bezwzględny błąd procentowy godzinowy

BBPd\_trn=mean(BBPh\_trn); %bezwzględny błąd procentowy dobowy

BBPs\_trn=mean(BBPd\_trn);