Predictia semnalelor

False Nearest Neighbors

Tema proiectului

- implementarea unei metode de predictie a semnalelor utilizand metoda False Nearest Neighbors
- structura proiectului:
    I : functiile de predictie a semnalului si reconstructie a atractorului
    II: serviciul web si apelarea functionalitatilor deja implementate

De ce analiza seriilor de timp?

- utilizata in domenii diverse: de la economie, predictia vremii, predictia cutremurelor pana la preditia semnalelor
- una dintre cele mai importante probleme: predictia
- problema inversa: date fiind valorile unei serii de timp, trebuie indentificata functia nonlineara care le produce

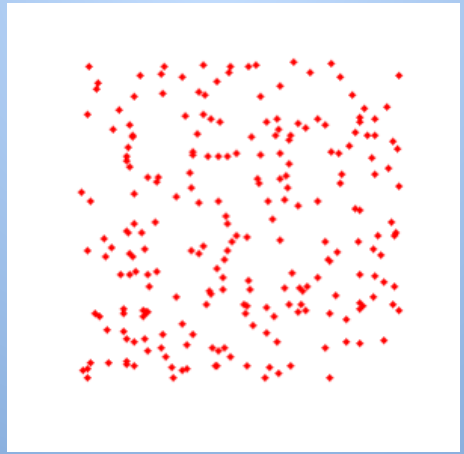
Despre predictia in haos

- haos: atunci cand prezentul determina viitorul, dar prezentul aproximativ nu aproximeaza viitorul
- principalul atribut al haosului: determinismul
- diferenta dintre un sistem haotic si unul aleator: posibilitatea de a putea face predictii pe termen scurt
- tehnici utilizate pentru a prezice seriile de timp haotice:
    I : globale (incearca reconstructia atractorului intr-un singur pas identificand functii globale de reprezentare ce fac aproximari locale)
    II: locale (indentifica functii sau mapari locale)

- metoda utlizata in cadrul acestui proiect (FNN) este o metoda locala (una dintre cele mai efective metode locale): se ajusteaza o functie pentru predictia in pasul de timp urmator (un pas inainte) luand in considerare cei mai apropiati n vecini

Atractor?!

- un atractor este un set catre care o variabila, ce se misca conform legilor dictate de un sistem dinamic, evolueaza de-a lungul timpului: punctele ce se apropie destul de mult de atractor raman aproape de acesta chiar daca sunt modificate putin
- intr-un sistem finit variabila poate fi reprezentata ca un vector n-dimensional
- attracting set pentru un sistem dinamic este o submultime inchisa A a spatiului fazelor (phase space) al sistemului, astfel incat pentru “mai multe” alegeri ale punctelor initiale sistemul va evolua catre multimea A
- ec: z’ =(1+2i-|z|^2)z



Spatiul fazelor

- starea unui sistem poate fi descrisa de variabilele sale de stare $x^1(t), x^2(t), ... ,x^d(t)$

ex: temperatura si presiunea pentru un sistem termodinamic

- cele d variabile la momentul t formeaza un vector spatial d-dimensional ce este numit spatiul fazelor

Model general predictie in haos

I : determinarea dimensiunii spatiului fazelor - m - spatiu in care se gaseste atractorul unui sistem haotic determinist (sursa semnalului) plecand de la esantionarea unui semnal y[n] = minimum embedding phase space

II : reconstructia traiectoriei sursei de semnal pentru toate momentele de timp pentru care avem esantionare

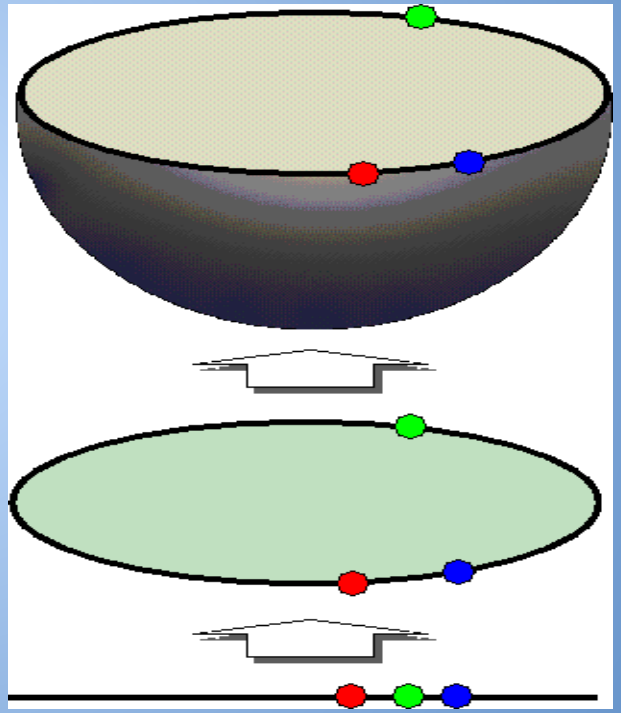
III : utilizand traiectoria reconstruita facem extrapolari pentru urmatorul segment din spatiul fazelor revenim in R si obtinem predictia semnalului cu un pas inainte, la momentul t+1

Algoritm general

- esantionarea semnalului y[n], n = 0...N in y[i] = (y[i], y[i-1],...,y[i-(m-1)], i = (m-1)...N
- aplicare False Nearest Neighbors si obtinere m = minimum embedding dimension
- calculul Minimum Delay Embedding Phase Space (= R^m)
- reconstructia traiectoriei sursei de semnal y[m-1],y[m],...,y[N-1] (traiectoria sursei de semnal in embedding phase space)
- predictie in R^m
- revenire in R si calculul lui y[N+1] estimat

False Nearest Neighbors

- avantaj: nu necesita seturi de date foarte mari!
- efectul: traiectoriile nu se mai auto-intersecteaza, nu exista “noduri” in haos



Algoritm False Nearest Neighbors

- parametrii: rTol = 10, aTol = 2, fTol = 0.1, m=1
repeta:
    calculeaza y[i], i = m-1,...,N-1 pentru m si y'[i], i = m,...,N pentru m+1
    calculeaza
f = [(dist(y'[i] - y'[i_nearest])^2 - dist(y[i] - y[i_nearest])^2) / dist(y'[i] - y'[i_nearest])^2)]^(½) > rTol
    m = m +1
pana cand f < fTol

return m - 1

Implementarea predictiei I

- clase folosite:
  MainKNN - clasa principala unde se realizeaza calculu
  embedding dimension si predictia noilor valori
  NDimVec - clasa ajutatoare, reprezinta un vector N dimensional
  de vectori
  KNeighbours - clasa ajutatoare, reprezinta un vector de vecini
  cu valorile vecinilor si distantele pana la acestia

Implementarea predictiei II

- determinarea minimum embedding dimension: calculez numarul de false nearest neighbours (FNN) pentru fiecare valoare de embedding dimension intre 1 si o valoare maxima data, 10 in acest caz; pentru fiecare dimensiune d, se construiesc vectori de dimensiune d ce reprezinta secvente de d elemente de pe pozitii consecutive din seria initiala; se calculeaza distantele euclidiene dintre acesti vectori si apoi se verifica FNN, determinandu-se m
- constructia predictorului utilizand k-nearest-neighbors: se ia o secventa formata din ultimele m valori din serie si se doreste predictia valorii m+1; pentru aceasta se identifica k secvente similare pentru care m+1 este cunoscuta, cele k secvente similare sunt cei mai apropiati k vecini (vecinii cei mai apropiati in distante euclidiene); se face o medie ponderata a valorilor acestor k+1 vecini si se afla m+1
- impartirea datelor de intrare in 2 parti egale, prima jumatate este folosita ca training set pentru antrenarea predictorului, iar cealalta jumatate ca test set; pe masura ce datele de test sunt parcurse valorile sunt adaugate in predictor(asa cum cere algoritmul)
- predictia valorilor si aflarea erorii patratice medii

Implementarea serviciului web

- serviciul web a fost implementat ca o aplicatie Java .war folosind WebServlets si serverul web Jetty
- exista mai multe metode de utilizare a aplicatiei web:

utilizarea interfetei grafice si accesarea proiectului in browser la adresa http://127.0.0.1:8080/knn-demo/ de unde se incarca un fisier de test, se face o cerere de tip HTTP POST catre servlet-ul Upload care va salva fisierul intr-un fisier temporar in /tmp/request.txt

in momentul in care se face submit cererii, esti redirectat catre o alta pagina unde este prezentat un fisier in format png, un grafic cu cele 2 seturi de valori, valorile reale si cele prezise, fisierul folosit este cel din /tmp/req.txt incarcat anterior

se poate accesa si un serviciu extern, un client extern poate accesa serviciul web la adresa http://127.0.0.1:8080/knn-demo/fnn/, are aceeasi functionare ca si pagina de vizualizare a graficului, se foloseste tot fisierul din /tmp/req.txt daca primeste un parametru in cererea de tipul HTTP GET se va folosi file_name=/cale/catre/fisier; FNNService va intoarce un JSON ce contine meanSquaredError si valorile prezise si cele dorite ce vor fi parsate in pagina

Resurse

- articolele puse la dispozitie pe cs.curs.pub.ro
- http://help.ixellence.com/dataplore
- Phase Space Reconstruction from Time Series Data : Where History Meets Theor by Ray Huffaker
- http://www.scholarpedia.org/article/Attractor_reconstruction
- wikipedia.org

- resurse pentru serviciul web:
    - biblioteca pentru HTTPServlets (servlet-api-2.5.jar)
        - http://mvnrepository.com/artifact/javax.servlet/javax.servlet-api/3.0.1
    - biblioteca pentru reprezentarea graficului (schart-1.2.1.jar)
        - http://xeiam.com/xchart.jsp
    - tutorial de UploadServlet
        - http://www.journaldev.com/1964/servlet-upload-file-and-download-file-example
    - tutorial xeiam xchart
        - http://xeiam.com/xchange_examplecode.jsp
    - Oracle Servlets tutorials