SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

SEMINAR

Poravnanje parova sekvenci korištenjem HMM

Petra Dunja Grujić Ostojić , Tea Čutić
Voditelj: *izv. prof. dr. sc. Mirjana Domazet-Lošo*

Zagreb, siječanj 2024.

SADRŽAJ

1. Uvod

Jedan od najtežih, a time i najistraživanijih, problema u području bioinformatike je problem poravnanja dviju sekvenci. Poravnanje je ključan korak u uspoređivanju i analizi DNA, RNA i proteinskih sekvenci te omogućuje uvid u zajedničke evolucijske korijene ili indikatore genetskih bolesti.

Ovaj rad bavi se implementacijom algoritma za poravnanje koji koristi skrivene Markovljeve modele (eng. Hidden Markov Models - HMM). Skriveni Markovljevi modeli su probabilistički modeli korišteni za predstavljanje stohastičkih procesa. Ovaj model sastoji se od više različitih stanja koja, u kontekstu modeliranja odnosa dvaju znakovnih nizova, mogu predstavljati stanja podudaranja, umetanja i brisanja. Vezama između tih stanja označene su vjerojatnosti prelaska iz jednog stanja u drugo ili ostanka u istom stanju. Za dobivanje inicijalne procjene parametara modela HMM korištena je javno dostupna baza poravnanja HIV sekvenci [1]. U ovom su radu korištena tri algoritma: Baum-Welch algoritam, Viterbijev algoritam te Needleman-Wunsch algoritam. Od navedenih algoritama, za algoritme Baum-Welch i Viterbi pisana je vlastita implementacija dok je algoritam Needleman-Wunsch preuzet iz javno dostupnog izvora te se koristi za procjenu kvalitete poravnanja kojeg dobijemo u našim rezultatima.

Drugo poglavlje rada bavi se detaljnijim opisom algoritama Baum-Welch i Viterbi. Treće poglavlje bavi se analizom dobivenih rezultata te njihovom usporedbom s rezultatima dobivenima s pomoću algoritma Needleman-Wunsch.

[1] https://www.hiv.lanl.gov/content/sequence/NEWALIGN/align.html

2. Opis algoritama

2.1. Baum-Welch algoritam

Baum-Welch algoritam, poznat i kao algoritam "forward-backward", je iterativni postupak koji se koristi za treniranje skrivenih Markovljevih modela odnosno za procjenu njegovih parametara pi, A i E. A je matrica vjerojatnosti prijelaza između stanja modela, E je matrica vjerojatnosti emisije simbola x u nekom stanju y, a pi je vektor početnih vjerojatnosti stanja. Algoritam se sastoji od tri koraka, inicijalizacije parametara HMM, računanje očekivanja te maksimizacije očekivanja. Implementacija algoritma rađena je po uzoru na paket HMM iz R-a [1].

Ulaz u Baum Welch algoritam je HMM s postavljenim inicijalnim vrijednostima navedenih parametara, niz parova simbola, vrijednost delta te zadani broj iteracija. Parametri HMM-a mogu se inicijalizirati nasumično, a u ovom smo ih radu procijenili na uzorku od 500 sekvenci odnosno 249 500 parova sekvenci. Prije inicijalizacije te prije pokretanja algoritma potrebno je obraditi sekvence. Najprije se izbace sve sekvence koje sadrže bilo koje znakove koji nisu znakovi A,C,T,G i -. Iz parova sekvenci potrebno je izbaciti praznine ako se one nalaze na istom indeksu u svakoj od sekvenci te uzastopne praznine (npr. sekvence -AG i C-T postaju AG i CT). Zatim je potrebno napraviti listu parova simbola od svakog para sekvenci. Neka imamo par sekvenci ACTA i GCAT pripadajuća lista parova simbola bila bi AG, CC, TA, AT.

Nakon inicijalizacije HMM te pripreme sekvenci za obradu, slijedi iterativno računanje očekivanja te njegova maksimizacija. Trajanje algoritma određeno je brojem iteracija x te vrijednosti delta d koje se postavljaju proizvoljno. Algoritam može prestati s radom nakon nekog određenog broja iteracija ili se nakon svake iteracija može računati vrijednost delta d_i. Vrijednost d_i definirana je kao zbroj kvadratne razlike početne te izračunate matrice A i kvadratne razlike početne i izračunate matrice E. Uvjet

[1] `https://rdrr.io/cran/HMM/src/R/HMM.r`

zaustavljanja na temelju vrijednosti delte glasi d_i < d, a provjerava se na kraju svake iteracije.

Unutar svake iteracije provodi se jedan forward i jedan backward algoritam. Forward algoritam koristi se za izračun vjerojatnosti da je u trenutku *t* model u stanju *i*, koristeći pri tom znanje o prethodno emitiranim simbolima. Rezultat forward algoritma je matrica koju označavamo s α. Backward algoritam koristi se za izračun β matrice koja također izračunava vjerojatnost da je u trenutku *t* model u stanju *i* te da je emitirao *T-t* simbola od trenutka *T* do trenutka *t+1*. Na temelju izračunatih matrica α i β dalje računamo vjerojatnosti γ i ξ. γ je posteriorna vjerojatnost da se sustav nalazi u stanju Si u trenutku t za promatrani niz i parametre HMM. ξ je vjerojatnost da se model, u trenutcima *t* i *t+1* nalazi u stanjima *i* i *j* uz promatrani niz i parametre modela. Na kraju iteracije, s dobivenim matricama ξ i γ ažuriramo vrijednosti parametra modela.

Budući da računamo s vjerojatnostima koje se kreću u rasponu od 0 do 1, za računanje se koriste logaritmi te pripadne logaritamske implementacije računskih operacija. Implementacije funkcija za zbroj i umnožak logaritama prikazane su kodom 1 (napomena: LOGZERO je varijabla postavljena na broj -INFINITY, a predstavlja broj 0).

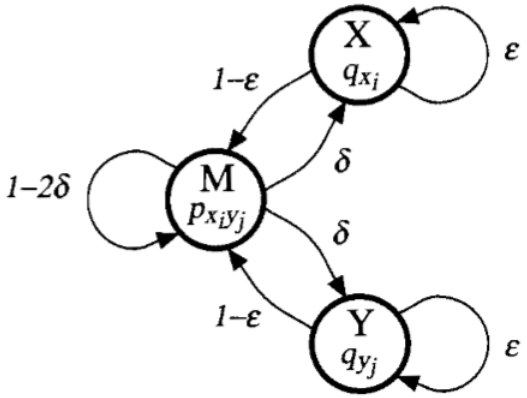
```
double log_sum(double x, double y){
        if (x==LOGZERO || y==LOGZERO){
                if (x==LOGZERO){
                        return y;
                }
                return x;
        } else if (x > y){
                return x + compute_log(1 + compute_exp(y-x));
        } else {
                return y + compute_log(1 + compute_exp(x-y));
        }
}

double log_product(double x, double y){
        if (x==LOGZERO || y==LOGZERO){
                return LOGZERO;
        }
        return x + y;
}
```

Listing 2.1: Funkcije za računanje s logaritamskih vrijednostima

2.2. Viterbi algoritam

Viterbijev algoritam pronalazi najvjerojatniji niz stanja u skrivenom Markovljevom modelu (HMM) na temelju opaženog niza simbola i parametara modela. Implementirale smo inačicu Viterbijeva algoritma koja pronalazi najvjerojatniji slijed skrivenih stanja na temelju opaženih parova simbola. Ta stanja ujedno predstavljaju poravnanje dva slijeda na ulazu algoritma.

Naš skriveni Markovljev model može se opisati slikom **??**. Razlikujemo 3 skrivena stanja: M, X i Y. U dijelu literature stanje X se naziva I_x, a stanje Y I_y upravo jer predstavlja umetanje (engl. insertion) u odgovarajuću sekvencu. Pretpostavlja se da je početno stanje M. Generiranje nizova ovisi o vjerojatnostima prijelaza između stanja i vjerojatnosti emisije parova simbola. Ovako opisani model ima dva stupnja slobode: vjerojatnost prijelaza (delta) iz stanja M u X ili Y i vjerojatnost ostanka (epsilon) u stanju X, tj. Y. Ostale vjerojatnosti proizlaze iz činjenice da zbroj izlaznih vjerojatnosti iz svih stanja mora biti 1. Stanje M odgovara emisiji para simbola x_i, y_j, a stanja X i Y predstavljaju emisiju simbola jedne sekvence i praznine u drugom slijedu. Pomoću Baum Welch algoritma i javno dostupne baze poravnanja HIV sekvenci odredile smo parametre modela. Ovisno o kriteriju zaustavljanja BW algoritma dobivena su 2 HMM modela.



Slika 2.1: HMM za poravnanje sljedova, slika preuzeta iz [**?**]

U nastavku je opisan Viterbijev algoritam. Iterativno se popunjavaju 3 matrice veličine $(n+1)\cdot(m+1)$, pri čemu je n veličina prvog, a m veličina drugog slijeda kojeg želimo poravnati . Inicijalno se postavi: $V^M(0,0) = 1, V^{I_x}(i,0) = 0, V^{I_y}(0,j) = 0$.

U praksi su slijedovi koje trebamo poravnati dugački te je navedeni algoritam neupotrebljiv jer je množenje velikog broja vjerojatnosti (brojeva između 0 i 1) numerički

Algorithm 1 Viterbijev algoritam

$$V^M(i,j) = p_{x_i y_j} \cdot \max \begin{cases} (1-2\delta) \cdot V^M(i-1, j-1) \\ (1-\varepsilon) \cdot V^{I_x}(i-1, j-1) \\ (1-\varepsilon) \cdot V^{I_y}(i-1, j-1) \end{cases}$$

$$V^{I_x}(i,j) = q_{x_i} \cdot \max \begin{cases} \delta \cdot V^M(i-1, j) \\ \varepsilon \cdot V^{I_x}(i-1, j) \end{cases}$$

$$V^{I_y}(i,j) = q_{y_j} \cdot \max \begin{cases} \delta \cdot V^M(i, j-1) \\ \varepsilon \cdot V^{I_y}(i, j-1) \end{cases}$$

nestabilno. Zato smo implementirale logaritamsku inačicu.

Algorithm 2 Viterbijev algoritam - logaritamska inačica

$$\log V^M(i,j) = \log p_{x_i y_j} + \max \begin{cases} \log(1-2\delta) + \log V^M(i-1, j-1) \\ \log(1-\varepsilon) + \log V^{I_x}(i-1, j-1) \\ \log(1-\varepsilon) + \log V^{I_y}(i-1, j-1) \end{cases}$$

$$\log V^{I_x}(i,j) = \log q_{x_i} + \max \begin{cases} \log \delta + \log V^M(i-1, j) \\ \log \varepsilon + \log V^{I_x}(i-1, j) \end{cases}$$

$$\log V^{I_y}(i,j) = \log q_{y_j} + \max \begin{cases} \log \delta + \log V^M(i, j-1) \\ \log \varepsilon + \log V^{I_y}(i, j-1) \end{cases}$$

Potrebno je iterativno ažurirati 3 matrice (V^M, V^{Ix}, V^{Iy}). Radi se o velikom zauzeću memorije te u slučaju dugačkih sekvenci moguće je da one neće stati u radnu memoriju. Već iz samih jednadžbi može se primijetiti da su nam za izračun vrijednosti u koraku i, j potrebni samo podaci iz prethodnog koraka $i-1, j-1$. Tako da umjesto da pohranjujemo 3 velike matrice možemo iz koraka u korak mijenjati 1 vektor veličine drugog slijeda i pamtiti takav vektor iz prethodnog koraka i. U našoj implementaciji element tog vektora je struktura ViterbiInfo koja sadrži 3 broja koja predstavljaju dotadašnju log vjerojatnost za svako skriveno stanje. Zadnji element posljednjeg vektora koji izračunamo predstavlja log vjerojatnost završetka dobivenog poravnanja u stanju M, I_x i I_y. Odredi se najvjerojatnije stanje i rekonstruira se optimalno poravnanje.

Kako bismo rekonstruirali optimalno poravnanje ulaznih nizova x i y, potrebno je

pamtiti iz kojeg se stanja došlo u najvjerojatnije stanje u svakom koraku. Rekonstrukcija se započinje od kraja. Indeksi i i j se postavljaju na duljinu prvog, odnosno drugog niza. Moguća su 3 slučaja:

1. prethodno stanje je M – emitiraj znakove x[i-1],y[i-1] i smanji indekse i i j za jedan

2. prethodno stanje je I_x – emitiraj znak x[i-1] i prazninu u drugu sekvencu, umanji i za jedan

3. prethodno stanje je I_y – emitiraj znak y[j-1] i prazninu u prvu sekvencu, umanji j za jedan

Postupak se ponavlja dok nisu emitirani svi simboli iz oba ulazna niza.

Viterbijev algoritam primjer je algoritama dinamičkog programiranja jer efikasno rješava problem optimalnog poravnanja sljedova razbijajući ga na manje podprobleme (manje dijelove sljedova) i koristeći njihova optimalna rješenja za konstrukciju globalnog optimalnog poravnanja. Naime, u svakom se koraku računa trenutno najvjerojatnije stanje s obzirom na emitirane simbole i prethodno stanje sustava.

3. Analiza rezultata

Pomoću algoritma Baum Welch procijenile smo parametre HMM modela koji opisuje poravnanje HIV sljedova. Korištena su 2 kriterija zaustavljanja algoritma:

1. delta = 0.001, broj iteracija = 100
2. delta = 0.01, broj iteracija = 100

Pomoću dobivenih modela generirale smo poravnanje nasumično odabranih 50 sljedova. U [**?**] navode da je primjena Viterbijeva algoritma na HMM modelu ekivalentna algoritmu dinamičkog programiranja za optimalno poravnanje sekvenci kao što je Needleman Wunsch.

Na odabranim sekvencama provele smo poravnanje pomoću NW algoritma uz 2 kriterija bodovanja poravnanja sljedova:

– +2 za podudaranje, -1 za nepodudaranje, -2 za prazninu

– +2 za podudranje, -1 za nepodudaranje, -4 za prazninu

MODEL	SREDNJA VRIJEDNOST BODOVANJA	STANDARDNA DEVIJACIJA
Model 1	25.4	3.2
Model 2	18.7	2.8
Model 3	22.1	4.5

Tablica 3.1: Srednje vrijednosti bodovanja i standardne devijacije poravnanja modela

treba dodati graf poravanja sekvenci za modele i komentirati rezultate

Rezultati bi bili bolji da se koristilo više sekvenci za inicijalnu procjenu modela pomoću Baum Welch algoritma. Bilo bi poželjno iskoristiti i sekvence različite od HIVa kako bi model bolje generalizirao. Zbog ograničenih resursa, korišten je manji broj sekvenci za učenje parametara modela i za testiranje Viterbijeva algoritma. Ipak, rezultati su pokazali da je ovaj pristup poravananja blizak rezultatima NW algoritma

ukoliko su sekvence koje treba poravnati slične onima koje su korištene za određivanje parametara HMM.

4. Zaključak

U ovom je radu isprobano poravnanje sljedova HIVa pomoću skrivenog Markovljeva modela. Za određivanje parametara modela korišten je algoritam Baum Welch, a za generiranje poravnanja sljedova Viterbijev algoritam. Dobiveni rezulati su uspoređeni s rezultatima dobivenim algoritmom Needleman Wunsch koji generira globalno optimalno poravnanje sljedova. Ključno poboljšanje bi bilo korištenje većeg broja slijedova za procjenu parametara modela te optimizacija vremenske i memorijske složenosti implementacije algoritama.

5. Literatura

[] Richard Durbin, Sean R Eddy, Anders Krogh, i Graeme Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.