SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Silvestar Mavrek

ITERATIVNO TRAŽENJE FRAZA I STATISTIKA SEMANTIČKOG INDEKSIRANJA

Diplomski rad

Voditelj rada:
doc. dr. sc. Pavle Goldstein

Zagreb, veljača 2019.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Sadržaj

Uvod

Klasifikacija proteina jedan je od najvažnijih zadataka molekularne biologije. Potreba za točnim, automatiziranim metodama klasifikacije nastavlja se povećavati kako napredak biotehnologije otkriva nove proteine. Proteini su složene tvari prisutne u svim živim bićima izravno uključene u procese važne za život. Molekula proteina sastoji se od mnogo aminokiselina spojenih zajedno oblikujući duge lance koje možemo zamišljati kao kuglice nanizane na žici. Proteinske familije su skupine evolucijski povezanih proteina, slične trodimenzionalne strukture, sličnih funkcija i sličnih nizova aminokiselina. Upravo je sličnost aminokiselinskih nizova najlakše provjeriv pokazatelj homologije i najjasniji pokazatelj zajedničkog podrijetla. Prepoznavanje karakterističnih motiva, odnoso kraćih nizova aminokiselina, standardna je tehnika prepoznavanja proteinskih familija. U ovom radu koristit ćemo jednostavniji pristup i proteine promatrati na razini još kraćih blokova aminokiselina. Takve blokove zovemo n-gramima, gdje n označava njihovu duljinu.

Identifikacija takvih segmenta proteina koji mogu točno razlikovati proteinske familije zanimljivije je znanstveno pitanje od same klasifikacije. Brojne metode klasificiranja usmjerene su na točnu klasifikaciju ali ne objašnjavaju koje to značajke doista pridonose razlikovanju familija. Diskriminativni n-grami su kratki peptidni nizovi koji su vrlo česti u jednoj familiji ali su minimalno prisutni ili ih nema u drugim familijama. U ovom radu predstavljamo neke metode identifikacije takvih diskriminativnih n-grama i analiziramo učinkovitost klasifikacije proteinskih familija takvim pristupom.

Poglavlje 1

Klasifikacija proteina kao klasifikacija teksta

Korištenje n-grama kao obilježja, dobro je poznata tehnika u obradi jezika, odnosno teksta, između ostalog i u dohvatu informacija. Dohvat informacija je znanost o traženju informacija u dokumentu, traženja samih dokumenata te traženju metapodataka koji opisuju podatke u bazama tekstova. Proces dohvaćanja informacija počinje kada korisnik unese upit u sustav. Upit je formalna izjava koja opisuje informacije koje korisnik treba, na primjer, tekst koji unosimo u tražilicu kada pretražujemo internet. Jasno je da takvi upiti ne određuju jednoznačno objekte, posebno tekstove, iz kolekcije. Umjesto toga, nekoliko tekstova može odgovarati istom upitu, vjerojatno s različitim stupnjevima podudaranja. Salton i Buckley su u [1] pokazali da sustav indeksiranja teksta temeljen na dodjeljivanju odgovarajućih težina pojedinačnim pojmovima daje bolje rezultate dohvaćanju informacija od onih dobivenih s drugim, složenijim tekstualnim prikazima.

Sustav indeksiranja teksta je pristup u kojem se svakom dokumentu ili upitu pridružuje vektor pojmova ili izraza oblika:

$$D = (t_i w_{di}, t_j w_{dj}, ..., t_p w_{dp})$$

gdje svaki t_k označava izraz dodjeljen dokumentu D iz uzorka, a w_{dk} težinu koju termin t_k ima u dokumentu D. Na primjer, nekom dokumentu M iz područja matematike mogao bi se pridružiti vektor

$$M = (..., \text{linearno, dimenzija,2, potprostor,2, jednadžba, teorem,3...})$$

Isto tako nekom upitu, odnosno zahtjevu za informacijom pridružen je vektor njegovih karakterističnih izraza oblika:

$$Q = (q_a w_{qa}, q_b w_{qb}, ..., q_r w_{qr})$$

Unaprijed definiranim mjerama sličnosti, svaki dokument iz baze uspređuje se s upitom i oni s najvećom sličnošću izlaze kao rezultat.

Budući da su aminokiseline označene znakovima, preciznije slovima engleskog alfabeta, protein u nekoj mjeri možemo promatrati kao tekst, a kratke nizove aminokiselina riječima. Kada tome dodamo da za težinu neke riječi u dokumentu, jednostavno, možemo uzeti broj njenog pojavljivanja, onda je jasno da je ovakav model primjenjiv i u klasifikaciji proteinskih familija. Slično kao u primjeru, proteinskim familijama dodijelit ćemo njoj karakteristične n-grame. Proteine za validaciju rješenja promatrat ćemo kao upite i pridruživati im vektore s frekvencijama pojavljivanja pojedinih n-grama u tom proteinu. Na kraju, nekom mjerom sličnosti odlučivat ćemo koji protein pridružiti kojoj familiji. Podaci korišteni u ovom radu preuzeti su s internetske stranice Pfam, velike zbirke proteinskih familija od kojih je svaka prikazana aminokiselinskim nizovima, višestrukim poravnanjima i skrivenim Markovljevim modelima.

Poglavlje 2

Matematička podloga

2.1 Linearna algebra

Prije nego što počnemo govoriti o rješenju problema, koji je tema ovog rada, potrebno je definirati određene pojmove i apstraktne matematičke strukture koje ćemo susretati u daljnjem čitanju. Stoga, u ovom poglavlju usko slijedimo izvor [2]. Najprije, trebat će nam matematički model u kojem ćemo promatrati i obrađivati naše objekte od interesa. Za to će nam poslužiti vektorski prostor. Vektorski prostor je skup na kojem su zadane binarna operacija zbrajanja i operacija množenja skalarima koje poštuju "uobičajena" računska pravila. Kako bi definiciju mogli iskazati precizno, podsjetimo se pravila računanja s brojevima.

Napomena 2.1.1. *Binarne operacije zbrajanja* $+ : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ *i množenja* $\cdot : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ *na skupu realnih brojeva imaju sljedeća svojstva:*

1. $\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma, \quad \forall \alpha, \beta, \gamma \in \mathbb{R}$;
2. *postoji* $0 \in \mathbb{R}$ *sa svojstvom* $\alpha + 0 = 0 + \alpha = \alpha, \quad \forall \alpha \in \mathbb{R}$;
3. *za svaki* $\alpha \in \mathbb{R}$ *postoji* $-\alpha \in \mathbb{R}$ *tako da je* $\alpha + (-\alpha) = -\alpha + \alpha = 0, \quad \forall \alpha \in \mathbb{R}$;
4. $\alpha + \beta = \beta + \alpha, \quad \forall \alpha, \beta \in \mathbb{R}$;
5. $\alpha(\beta\gamma) = (\alpha\beta)\gamma, \quad \forall \alpha, \beta, \gamma \in \mathbb{R}$;
6. *postoji* $1 \in \mathbb{R}$ *sa svojstvom* $1 \cdot \alpha = \alpha \cdot 1 = \alpha, \quad \forall \alpha \in \mathbb{R}$;
7. *za svaki* $\alpha \in \mathbb{R}, \alpha \neq 0$, *postoji* $\alpha^{-1} \in \mathbb{R}$ *tako da je* $\alpha\alpha^{-1} = \alpha^{-1}\alpha = 1$;
8. $\alpha\beta = \beta\alpha, \quad \forall \alpha, \beta \in \mathbb{R}$;
9. $\alpha(\beta + \gamma) = \alpha\beta + \alpha\gamma, \quad \forall \alpha, \beta, \gamma \in \mathbb{R}$

Kad god imamo neki skup $\mathbb{F}$ na kojem su zadane binarne operacija zbrajanja $+ : \mathbb{F}\times\mathbb{F} \to \mathbb{F}$ i množenja $\cdot : \mathbb{F} \times \mathbb{F} \to \mathbb{F}$ koje imaju navedenih devet svojstava, kažemo da je $\mathbb{F}$ polje. Prema tome, prethodna napomena se može preformulirati: skup realnih brojeva s uobičajenim operacijama zbrajanja i množenja je polje. Odmah vidimo da su skup racionalnih brojeva $\mathbb{Q}$, kao i skup kompleksnih brojeva $\mathbb{C}$ primjeri polja.

Definicija 2.1.2. *Neka je V neprazan skup na kojem su zadane binarna operacija zbrajanja $+ : V \times V \to V$ i operacija množenja skalarima iz polja $\mathbb{F}$,$\cdot : V \times V \to V$. Kažemo da je uređena trojka $(\mathbb{F}, \cdot, +)$ vektorski prostor nad poljem $\mathbb{F}$ ako vrijedi:*

1. $a + (b + c) = (a + b) + c, \forall a, b, c \in V$;
2. *postoji* $0 \in V$ *sa svojstvom* $a + 0 = 0 + a = a, \forall a \in V$;
3. *za svaki* $a \in V$ *postoji* $-a \in V$ *tako da je* $a + (-a) = -a + a = 0$;
4. $a + b = b + a, \forall a, b \in V$;
5. $\alpha(\beta a) = (\alpha\beta)a, \forall \alpha, \beta \in \mathbb{F}, \forall a \in V$;
6. $(\alpha + \beta)a = \alpha a + \beta a, \forall \alpha, \beta \in \mathbb{F}, \forall a \in V$;
7. $\alpha(a + b) = \alpha a + \alpha b, \forall \alpha \in \mathbb{F}, \forall a, b \in V$;
8. $1 \cdot a = a, \forall a \in V$;

Dodajmo još kako je običaj da se elementi vektorskog prostora nazivaju vektorima. Neka je $n \in \mathbb{N}$, te neka $\mathbb{R}^n$ označava skup svih uređenih n-torki realnih brojeva (drugim riječima $\mathbb{R}^n$ je Kartezijev produkt od n kopija skupa $\mathbb{R}$). Definirajmo

$$(a_1, a_2, \ldots, a_n) + (b_1, b_2, \ldots, b_n) = (a_1 + b_1, a_2 + b_2, , a_n + b_n)$$

i za $\alpha \in \mathbb{R}$

$$\alpha(a_1, a_2, \ldots, a_n) = (\alpha a_1, \alpha s_2, \ldots, \alpha a_n)$$

Jasno je da je uz ovako definirane operacije $\mathbb{R}^n$ vektorski prostor.

Definicija 2.1.3. *Neka je V vektorski prostor nad $\mathbb{F}$ i $S = \{a_1, a_2, \ldots, a_k\}, k \in \mathbb{N}$, konačan skup vektora iz V. Kažemo da je skup S linearno nezavisan ako vrijedi:*

$$\alpha_1, \alpha_2, \ldots, \alpha_k \in \mathbb{F}, \sum_{i=1}^{k} \alpha_i a = 0 \implies \alpha_1, \alpha_2, \ldots, \alpha_k = 0.$$

U suprotnom kažemo da je skup S linearno zavisan.

Linearna nezavisnost skupa S znači da se nulvektor može prikazati kao linearna kombinacija elemenata skupa S samo na trivijalan način.

Definicija 2.1.4. *Neka je V vektorski prostor nad poljem $\mathbb{F}$ i $S \subseteq V, S \neq \emptyset$. Linearna ljuska skupa S označava se simbolom $[\,S\,]$ i definira kao:*

$$[\,S\,] = \{\sum_{i=1}^{k} \alpha_1 a_i : \alpha_i \in \mathbb{F}, a_i \in S, k \in \mathbb{N}\}.$$

Dodatno, definira se $[\,\emptyset\,] = \{0\}$.

Linearna ljuska nepraznog skupa S je, dakle, skup svih mogućih linearnih kombinacija elemenata skupa S.

Definicija 2.1.5. *Neka je V vektorski prostor i $S \subseteq V$. Kaže se da je S sustav izvodnica za V (ili da S generira V) ako vrijedi $[\,S\,] = V$.*

Primijetimo, skup S je sustav izvodnica za V ako se svaki vektor iz V može prikazati kao linearna kombinacija elemenata skupa S. Poželjno je naći čim manji sustav izvodnica te time moći opisati sve vektore prostora sa što manje elemenata.

Definicija 2.1.6. *Konačan skup $B = \{b_1, b_2, \ldots, b_n\},\ n \in \mathbb{N}$, u vektorskom prostoru V se naziva baza za V ako je B linearno nezavisan sustav izvodnica za V.*

Sljedeći teorem je fundamentalan rezultat linearne algebre. Smisao je u tome da svaki vektor danog prostora možemo na jedinstven način predočiti kao linearnu kombinaciju vektora baze.

Teorem 2.1.7. *Neka je V vektorski prostor nad poljem $\mathbb{F}$, te neka je $B = \{b_1, b_2, \ldots, b_n\}$ baza za V. Tada za svaki vektor $v \in V$ postoje jedinstveno određeni skalari $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}$ takvi da vrijedi $v = \sum_{i=1}^{n} \alpha_i b_i$.*

Dokaz. Jer je po definiciji B sustav izvodnica za V, svaki vektor $v \in V$ dopušta prikaz u obliku $v = \sum_{i=1}^{n} \alpha_i b_i$. Ako bi za neki $v \in V$ vrijedilo $v = \sum_{i=1}^{n} \alpha_i b_i$ i također $v = \sum_{i=1}^{n} \beta_i b_i$ oduzimanjem bismo dobili $\sum_{i=1}^{n} (\alpha_i - \beta_i) b_i = 0$. Jer je skup B i linearno nezavisan, po definiciji slijedi $\alpha_i - \beta_i = 0, \quad \forall i = 1, 2, \ldots, n.$ □

Definicija 2.1.8. *Neka je V vektorski prostor nad $\mathbb{F}$ i neka je $M \subseteq V$, $M \neq \emptyset$. Ako je i $(M, +, \cdot)$ vektorski prostor nad $\mathbb{F}$ uz iste operacije iz V, kažemo da je M potprostor od V.*

Ovakvu situaciju možemo zamišljati kao jedan vektorski prostor smješten u drugome. Kasnije u radu biti će situacija kada ćemo vektore iz vektorskog prostora preslikavati u njegove potprostore. U tu svrhu uvodimo sljedeće pojmove.

Definicija 2.1.9. *Neka su V i W vektorski prostori nad istim poljem $\mathbb{F}$. Preslikavanje $A : V \to W$ zove se linearan operator ako vrijedi*

$$A(\alpha x + \beta y) = \alpha Ax + \beta Ay, \quad \forall \alpha, \beta \in \mathbb{F}, x, y \in V.$$

Vektorski prostor linearnih operatora sa skupa V u V označavat ćemo s $L(V)$.

Definicija 2.1.10. *Neka je V vektorski prostor nad poljem $\mathbb{F}$. Skalarni produkt na V je preslikavanje $\langle \cdot, \cdot \rangle : V \to \mathbb{F}$ koje ima sljedeća svojstva:*

1. $\langle x, x \rangle \geq 0, \forall x \in V$;
2. $\langle x, x \rangle = 0 \iff x = 0$;
3. $\langle x_1 + x_2, y \rangle = \langle x_1, y \rangle + \langle x_2, y \rangle, \quad \forall x_1, x_2, y \in V$;
4. $\langle \alpha x, y \rangle = \alpha \langle x, y \rangle, \quad \forall \alpha \in \mathbb{F}, \forall x, y \in V$;
5. $\langle x, y \rangle = \overline{\langle y, x \rangle}, \quad \forall x, y \in V$;

Definicija 2.1.11. *Vektorski prostor na kojem je definiran skalarni produkt zove se unitaran prostor.*

U $\mathbb{R}^n$ skalarni produkt definiran je s

$$\langle (x_1, \ldots, x_n), (y_1, \ldots, y_n) \rangle = \sum_{i=1}^{n} x_i y_i$$

Definicija 2.1.12. *Neka je V konačnodimenzionalan unitaran prostor i A linearan operator na V. Operator A^* sa svojstvom*

$$\langle Ax, y \rangle = \langle x, A^* y \rangle, \quad \forall x, y \in V$$

zove se hermitski adjungiran operator operatoru A.

Propozicija 2.1.13. *Neka je V konačnodimenzionalan vektorski prostor i $P \in L(V)$. Operator P je ortogonalni projektor ako i samo ako vrijedi $P^2 = P = P^*$.*

Preslikavanje $P : V^3(O) \to V^3(O)$ definirano s $P(\overrightarrow{OT}) = \overrightarrow{OT'}$, gdje je $T = (x, y, z)$ i $T' = (x, y, 0)$, je linearan operator; P se naziva ortogonalni projektor prostora $V^3(O)$ na $V^2(O)$ pri čemu smo prostor $V^2(O)$ identificirali s potprostorom od $V^3(O)$ kojeg čine svi radijvektori čije završne točke leže u xy-ravnini.

Definicija 2.1.14. *Neka je V unitaran prostor. Norma na V je funkcija $\| \cdot \| : V \to \mathbb{R}$ definirana s $\|x\| = \sqrt{\langle x, x \rangle}$.*

Upravo ovako definiranu normu uzimamo za mjeru sličnosti kojom ćemo, na neki način odrediti koliko je neki protein blizu kojoj proteinskoj familiji.

2.2 Statistika

U radu će se spominjati i neki pojmovi iz statistike. Kako bi kasniji tekst bio razumljiv sada ih definiramo.

Definicija 2.2.1. *Podatke $x_1, x_2, ...x_n$ poredane po veličini označavamo s $x_{(1)} \leq x_{(2)}, ..., \leq x_{(n)}$. Pri tome za $s = k + r$, gdje je $k \in \mathbb{Z}$, $r \in [0, 1\rangle$, vrijedi formula:*

$$x_s = x_k + r[x_{k+1} - x_k]$$

Nadalje:

1. *medijan uzorka je $x_{(\frac{n+1}{2})}$.*
2. *95%-kvantil je $x_{(\frac{95(n+1)}{100})}$.*
3. *99%-kvantil je $x_{(\frac{99(n+1)}{100})}$*

Odnosno, za brojčani niz podataka $x_1, x_2, \ldots, x_n$ Medijan M je realan broj sa svojstvom da je pola podataka veće ili jednako M, a pola podataka je manje ili jednako M. Također, 95%-kvantil je realan broj $q_{0.95}$ sa svojstvom da je:

1. udio podataka manjih ili jednakih $q_{0.95}$ približno 95%
2. udio podataka većih ili jednakih $q_{0.95}$ približno 5%

Slično zaključujemo i za 99%-kvantil.

Poglavlje 3

Klasifikacija proteinskih familija

3.1 Diskriminativni n-grami

Lipaze i Walkerovi motivi

Na početku promatramo problem klasifikacije dvije familije proteina, lipoproteinske lipaze i Walkerove motive. Lipoproteinske lipaze su enzimi koji ubrzavaju kemijsku reakciju razgradnje masti u doticaju s vodom. Walkerovi motivi imaju ulogu u vezanju molekula ATP-a. Običaj je da se kod ovakvih zadataka skup podataka podijeli na dva dijela. Veći dio podataka, kojeg zovemo trening set, služi za analizu i izgradnju rješenja. Na drugom dijelu podataka testiramo naše rješenje i tako ocjenjujemo koliko je ono dobro. Taj drugi dio podataka zovemo test set. Iz svake familije skupili smo kolekciju od 400 proteina. Po 300 za trening set i 100 za ocjenu rješenja, odnosno test set. U procesu razvijanja algoritma za klasifikaciju važno nam je da znamo koji proteini dolaze iz koje familije. Prema tome, unutar trening seta poredani su redom; najprije svi iz jedne familije, potom oni iz druge. Već ranije smo spomenuli kako proteine promatramo kao nizove znakova što nam omogućuje, recimo to tako, šetnju kroz protein i promatranje nekih njegovih manjih dijelova. Upravo je to naš prvi cilj. Iterativno prolaziti kroz svaki protein u trening skupu, promatrati njegove četveročlane blokove, tj. uređene četvorke znakova i svaki novi takav blok koji se pojavi spremiti u bazu.

$$\ldots \underbrace{VIFN}\,\underbrace{FGDS}\,NSD\ldots$$

Na taj način, dobili smo sve različite uređene četvorke koje se pojavljuju u našoj kolekciji proteina.

Sljedeće što želimo saznati je koliko se puta svaka uređena četvorka iz baze pojavljuje u nekom proteinu? Taj broj pojavljivanja izraza t u dokumentu d još zovemo i frekvencija od t u d. Kako bismo to saznali, opet moramo prolaziti kroz protein i promatrati blokove od

četiri znaka. Na koju god kombinaciju znakova naiđemo, ona se nalazi u našoj bazi iz prvog dijela. Jednostavno, za svaki protein prebrojimo koliko se puta svaka uređena četvorka iz baze pojavljuje u njemu. Time mu pridružujemo vektor čiji elementi odgovaraju broju pojavljivanja uređenih četvorki iz baze u samom proteinu. Za ilustraciju, zamislimo da je neki protein zadan kao *AGABAGAB*. Tada su *AGAB*, *GABA*, *ABAG*, *BAGA* sve četvorke koje se u njemu pojavljuju. U tom slučaju, vektor koji bismo mu pridružili je $[2, 1, 1, 1]$ jer se *AGAB* pojavljuje dva puta, a sve ostale četvorke jednom.

$$\begin{matrix} AGAB \\ GABA \\ ABAG \\ BAGA \end{matrix} \begin{bmatrix} 2 \\ 1 \\ 1 \\ 1 \end{bmatrix} = v$$

Time smo definirali preslikavanje $\Psi : \{\text{Lipaze, Walkerovi motivi}\} \to \mathbb{R}^n$, gdje je n broj uređenih četvorki. Nakon što ovaj postupak provedemo za sve proteine iz trening seta, rezultate možemo posložiti u matricu čiji su stupci upravo dobiveni vektori. U tom slučaju, prvih tristo stupaca matrice odnosi se na lipaze, a drugih tristo na Walkerove motive. Svaki redak iz matrice odnosi se na određenu četvorku iz baze pa se lako može pročitati koliko se koja uređena četvorka puta pojavljuje u kojem proteinu. Na primjer, na donjoj slici s $f_{i,j}$ je označen broj pojavljivanja uređene četvorke $GFIS$ u proteinu koji se nalazi na j-tom mjestu u trening setu.

$$M = \begin{bmatrix} f_{1,1} & \dots & f_{1,300} & f_{1,301} & \dots & f_{1,600} \\ & \ddots & & & & \\ & & f_{i,j} & & & \\ & & & \ddots & & \\ f_{n,1} & \dots & f_{n,300} & f_{n,301} & \dots & f_{n,600} \end{bmatrix} \begin{matrix} MATL \\ \vdots \\ GFIS \\ \vdots \\ TVVA \end{matrix}$$

Budući da želimo razlikovati ove dvije familije proteina, korisno nam je da ih počnemo promatrati odvojeno. Početnu matricu podijelimo na dva dijela tako da prvih tristo stupaca uzmemo kao jednu matricu, a drugih tristo kao drugu. Time smo dobili dvije matrice od kojih svaka sadrži informacije za jednu familiju proteina. U prvoj matrici, koju označavamo M_1, su frekvencije pojvljivanja vektora baze u lipazama, a u drugoj matrici, M_2, su pojavljivanja u Walkerovim motivima. Zanima nas postoje li neke karakteristične četvorke iz baze koje se mnogo češće pojavljuju u jednoj nego li u drugoj familiji proteina. Time bi na neki način odgovorili na pitanje jesu li proteinske familije okarakterizirane nekim skupom uređenih četvorki.

Kako bi na to odgovorili, za svaku uređenu četvorku iz baze potrebno je zbrojiti brojeve pojavljivanja u svim proteinima i to za svaku familiju posebno. Jasno je da ćemo to postići ako zbrojimo stupce prethodno dobivenih matrica M_1 i M_2. Tako smo dobili dva vektora od

kojih svaki nosi informaciju koliko se puta svaka uređena četvorka baze pojavila u proteinima odgovarajuće familije. Preciznije, iz matrice M_1 dobili smo vektor stupac v_1, u čijem i-tom retku stoji broj pojavljivanja i-te uređene četvorke iz baze u lipazama. Analogno za vektor v_2.

$$M = \begin{bmatrix} f_{1,301} & \cdots & f_{1,600} \\ & \ddots & \\ & f_{1,j} & \\ & & \ddots \\ & & f_{n,600} \end{bmatrix} \rightarrow v_2 = \begin{bmatrix} \vdots \\ f_i \\ \vdots \end{bmatrix} \begin{matrix} MATL \\ \vdots \\ GFIS \\ \vdots \\ TVVA \end{matrix}$$

Na gornjoj slici je s f_i označen ukupan broj javljanja uređene četvorke $GFISS$ u Walkerovim motivima iz trening seta. Sada možemo promatrati razlike u broju ponavljanja uređenih četvorki u lipazama i Walkerovim motivima. Za one kod kojih je takva razlika najveća, naslućujemo da se u jednoj proteinskoj familiji pojavljuju češće nego li u drugoj. Za svaku familiju, pokušat ćemo pronaći takve karakteristične uređene četvorke. Jasno je da se ista četvorka ne može nalaziti u skupu, nazovimo to tako, dobrih četvorki i za jednu i za drugu familiju. Dobit ćemo skup četvorki koje karakteriziraju lipaze i onih koje karakteriziraju Walkerove motive, pri čemu su ta dva skupa disjunktna. Koliko će biti veliki ti skupovi? Odnosno, koliko takvih specifičnih četvorki tražimo? Pokušajmo najprije sa po sto za svaku grupu. Imajući na umu dimenziju prostora ovo će bit značajna restrikcija broja elemenata baze. Dakle, promatramo vektor $v_1 - v_2$ i uzimamo sto pozicija s najvećim elementima. Pamtimo koji elementi baze su pridruženi tim pozicijama i dobivamo novu, manju bazu uređenih četvorki značajnih za familiju lipaza, koju ćemo označavati s B_1. Analogni postupak ponavljamo za vektor $v_2 - v_1$ i dobivamo bazu četvorki za Walkerove motive B_2.

Pretpostavimo sada, da smo neki nepoznati vektor zapisali u bazi B_1 za lipaze, a potom i u bazi B_2 za Walkerove motive. Odnosno, kao ranije, pridružili mu vektore čiji elementi su frekvencije pojavljivanja karakteristične četvorke u samom proteinu pri čemu sada u obzir uzimamo samo one četvorke iz baze B_1 ili B_2. Intuicija nam govori da, ako je taj protein lipaza, elementi u vektoru koji je nastao brojanjem karakterističnih četvorki iz baze B_1 bit će veći od onih u drugom vektoru. To nam daje ideju kako će naše rješenje odlučiti kojoj familiji pridružiti nepoznati protein. Sjetimo se pojma norme koji smo uveli ranije. Upravo nam norma vektora daje njegovu veličinu, a ona je veća kako su njegovi elementi veći. Dakle, morat ćemo usporediti norme vektora koje smo pridružili novom proteinu. Spremni smo početi postupak validacije našeg rješenja. U test setu podataka imamo po sto proteina iz svake familije. Prvih sto su lipaze, a zatim slijedi sto Walkerovih motiva što nam omogućuje da znamo kojoj familiji koji od proteina pripada. Zna li to i naše rješenje? Hoće li ih pravilno klasificirati? Svakom proteinu iz test seta pridružujemo dva vektora.

Prvi dobijemo tako što promatramo pojavljivanja uređenih četvorki iz B_1, a drugi onih u B_2. Nakon toga izračunamo euklidsku normu svakog od ova dva vektora. Ako je norma veća, protein ćemo pridružiti lipazama. U drugom slučaju ga pridružujemo Walkerovim motivima. Kako bi provjerili točnost rješenja, jednostavno, za svaki protein uspoređujemo stvarnu familiju iz koje dolazi s onom koju je odredio model. Dobiveni su sljedeći rezultati:

Familija	**Točno**	**Netočno**
Lipaze	98	2
Walkerovi motivi	97	3

Tablica 3.1: Rezultati klasifikacije sa 100 uređenih četvorki

Vidimo da je točnost modela poprilično dobra. Od sto lipaza iz test seta, 98 smo pravilno klasificirali. Rezultat za Walkerove motive je samo malo lošiji. Ovime možemo biti zadovoljni, no možemo li postići još bolje ili jednako dobre rezultate uz dodatnu redukciju baze? Kakve bi rezultate dobili kada bi, umjesto sadašnjih sto, uzeli samo pedeset najboljih uređenih četvorki za svaku familiju. Ponavljamo postupak kao ranije. Promatramo vektore $v_1 - v_2$ i $v_2 - v_1$ i iz svakog uzimamo pedeset pozicija s najvećim elementima. Tako smo dobili nove baze B_1 i B_2 u kojima ćemo zapisivati nepoznate vektore prilikom klasifikacije. Ponovno, svakom proteinu iz test seta podataka pridružujemo vektore koji sadrže informacije o pojavljivanju uređenih četvorki iz novih baza i uspoređujemo norme. Rezultati su sljedeći:

Familija	**Točno**	**Netočno**
Lipaze	99	1
Walkerovi motivi	97	3

Tablica 3.2: Rezultati klasifikacije sa pedeset uređenih četvorki

Vidimo da su rezultati slični, čak i malo bolji nego li u prvom slučaju. Za ovaj problem rješenje radi vrlo dobro. Međutim, moguće je da se ove dvije familije proteina toliko razlikuju da njihova klasifikacija nije velik izazov. Također, pokušavamo razlikovati samo dvije familje proteina što opet olakšava posao našem modelu.

Što ako povećamo broj familija koje želimo klasificirati? Što ako uzmemo neke druge familije koje su možda sličnije pa ih je teže razlikovati?

Klasifikacija četiri i šest familija proteina

Opet ćemo ponavljati sličan postupak, ovaj puta uz nešto drugačiji set podataka. Pokušat ćemo razlikovati četiri familije proteina. *Cytochrome bc1 complex* koji se nalazi u stanicama svih životinja i aerobnih eukariota, a ima ulogu u transportu elektrona. Familiju

proteina *Upf2* iz kvasca *Saccharomyces cerevisiae* koji sudjeluju u procesu nadziranja i i spravljanja mRNA. Zatim, *Endoribonukleaze XendoU* koje se nalaze u vrsti žaba *Xenopus*, a imaju ulogu u sintezi RNA. Zadnja familija koju promatramo su peptidaze koje se nalaze u životinjama, biljkama, bakterijama, virusima, a kataliziraju proces razgradnje proteina na manje gradivne jedinice. U daljenjem tekstu ove familije označavat ćemo rednim brojem kojim smo je spomenuli. To znači da, ako kažemo prva familija, mislimo na *cytochrome bc1 complex*, druga familija je *Upf2*. Opet ćemo od svake familije uzeti po 300 proteina za trening i poredati ih na način da je prvih 300 iz prve familije koju smo spomenuli, drugih 300 iz druge i tako dalje. U test setu nalazi se po sto proteina iz svake familije. Prvi korak je, ponovno, gradnja baze svih uređenih četvorki koje se pojavljuju u bilo kojem proteinu iz trening seta. Kako ovaj put promatramo četiri familije, više je različitih proteina pa prema tome smisleno je očekivati da će i broj različitih četvorki biti veći. Prolaskom kroz protein promatramo blokove od četiri znaka i svaki novi na koji naiđemo spremamo u bazu. Na redu je prebrojavanje elemenata baze u proteinima i stvaranje matice. Postupak je identičan onom od ranije. Svakom proteinu pridružujemo vektor čiji su elementi frekvencije pojavljivanja četvorki iz baze.

$$M = \begin{bmatrix} f_{1,1} & \dots & f_{1,300} & \dots & f_{1,901} & \dots & f_{1,1200} \\ & & \ddots & & & & \\ & & & f_{i,j} & & & \\ & & & & \ddots & & \\ f_{n,1} & \dots & f_{n,300} & \dots & f_{n,901} & \dots & f_{n,1200} \end{bmatrix} \begin{matrix} MGGA \\ \vdots \\ PNIL \\ \vdots \\ QESD \end{matrix}$$

U matrici M stupci od jedan do tristo pridruženi su prvoj familiji proteina. Sljedećih tristo pridruženo je proteinima druge familije. Promatramo odgovarajuće grupe stupaca matrice i zbrajamo ih. Time smo dobili informacije o ukupnom broju pojavljivanja uređenih četvorki iz baze u svakoj od familija odvojeno. Podsjetimo, ako zbrojimo stupce od 600. do 900. matrice M, dobit ćemo vektor stupac čiji su elementi brojevi ukupnih pojavljivanja uređenih četvorki baze u trećoj familiji koju promatramo.

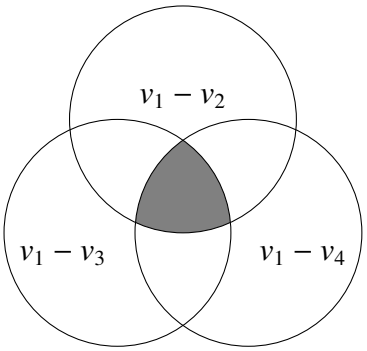
$$M = \begin{bmatrix} \dots & f_{1,601} & \dots & f_{1,900} & \dots \\ & & \ddots & & \\ \dots & f_{n,601} & \dots & f_{n,900} & \dots \end{bmatrix}$$

$$\downarrow$$

$$v_3 = \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \begin{matrix} MGGA \\ \vdots \\ QESD \end{matrix}$$

Na sličan način dobijemo vektore v_1, v_2 i v_4. Sada postaje malo kompliciranije. Sjetimo

se da je sljedeći korak izvlačenje baze dobrih četvorki za svaku proteinsku familiju iz test seta. U slučaju kad smo promatrali samo dvije familije to je bilo lako. Promatrali bi razlike u pojavljivanjima četvorki i uzeli one četvorke za koje su te razlike najveće. U ovoj situaciji u interesu nam je naći četvorke u čijim se pojavljivanjima jedna familija ističe u odnosu na druge tri. Jednostavno, ako tražimo bazu četvorki koje su češće u prvoj familiji, za njih vrijedi da se češće pojavljuj u prvoj nego li u drugoj pa možemo postupiti kao u prvom dijelu. Također, vrijedi i da se one javljaju češće u prvoj nego li u trećoj familiji. Isto vrijedi i za četvrtu familiju. Tada bi presjek sva ova tri skupa bila baza uređenih četvorki koje favoriziraju prvu familiju iz test seta.



Ne možemo znati koliko će takav presjek sadržavati elemenata pa ovdje ne možemo manipulirati brojem uređenih četvorki koje ćemo uzeti kao što smo to radili u prvom problemu kad smo najprije uzeli baze od sto pa zatim od pedeset elemenata.

Kako bi bilo jasnije, opisat ćemo proces izvačenja baze uređenih četvorki za prvu familiju. Isti se postupak kasnije jednako primjenjuje i na druge familije. Kako bi našli četvorke koje se više pojavljuju u prvoj nego li u drugoj familiji, slično kao u prvom dijelu, promatramo vektor $v_1 - v_2$. Na mjestima s najvećim elementima u tom vektoru su četvorke za koje je razlika u pojavljivanju najveća. Kod odluke koliko ćemo takvih pozicija odabrati, valja imati na umu da ćemo kasnije gledati presjeke triju skupova koji vjerojatno nisu previše slični. Prema tome, kako bi bili sigurniji da će rezultantni skup biti neprazan, valjalo bi uzeti više elemenata u ovim koracima. Uzmimo tisuću pozicija s najvećim elementima u vektoru $v_1 - v_2$ i njima pridružene četvorke čine skup B_{12}. Nadalje, promatramo vektore $v_1 - v_3$ i $v_1 - v_4$, u svakom uzimamo po tisuću pozicija s najvećim elementima i dobivamo skupove B_{13} i B_{14}. Konačno, naša baza četvorki za prvu familiju je presjek ovako dobivenih triju skupova. Odnosno

$$B_1 = B_{12} \cap B_{13} \cap B_{14}$$

Istim postupkom dobijemo baze B_2, B_3 i B_4 za ostale familije iz seta podataka.

Sada kad smo dobili baze u kojima su četvorke znakova karakteristične za pojedine familije, možemo u njima zapisivati proteine iz test seta podataka. Svakom proteinu iz test seta pridružujemo vektor pojavljivanja uređenih četvorki iz baze u tom proteinu. Preciznije,

svakom proteinu pridružit ćemo četiri vektora. Prvi vektor, kojeg ćemo označiti s p_1 sadrži brojeve pojavljivanja četvorki iz baze B_1. Na isti način dobivamo vektore p_2, p_3 i p_4 koji odgovaraju bazama B_2, B_3 i B_4. Izračunamo euklidsku normu tako dobivenih vektora. Pretpostavimo da za neki protein najveću normu ima vektor p_j za neki $j \in \{1, 2, 3, 4\}$ tada taj protein smještamo u j-tu familiju iz trening seta. Jednako kao što smo radili u slučaju prvog problema. Nakon što to napravimo za sve proteine iz test seta, možemo uspoređivati stvarne familije proteina i one koje je odredilo naše rješenje. Ovaj put su rezultati dosta lošiji. Pogledajmo kako je naš model razmišljao.

Familija 1			
100	0	0	0
Familija 2			
18	80	1	1
Familija 3			
46	0	54	0
Familija 4			
36	0	2	62

Tablica 3.3: Rezultati klasifikacije

U tablici je prikazano kako je model klasificirao pojedine familije. Najprije je ispisan naslov stvarne familije, a zatim u retku ispod, je broj proteina te familije koje je svrstao u prvu, zatim u drugu pa u treću i na kraju četvrtu familiju. Vidimo da je prvu familiju odlično prepoznao. Svi proteini te familije su pravilno klasificirani. Nažalost, to je jedini dobar slučaj. Kasnije su rezultati lošiji, posebno za treću familiju iz koje je uspješno klasificirano tek nešto više od pola proteina. Primijietimo da je model sklon proteine pridruživati prvoj familiji. Kada pogriješi, uglavnom se događa to da je protein, umjesto svojoj pravoj, pridružen prvoj familiji. Broj pogodaka i promašaja modela prikazan je u tablici ispod.

Familija	**Točno**	**Netočno**
Familija 1	100	0
Familija 2	80	20
Familija 3	54	46
Familija 4	62	38

Tablica 3.4: Točnost modela

Ovakvi rezultati su loši i sljedeći korak bi trebao biti poboljšanje modela. Prisjetimo se kako smo gradili baze za pojedine familije. Krajnja baza je bila presjek triju skupova koje smo dobili promatranjem najvećih elemenata vektora $v_i - v_j, \quad i, j \in \{1, 2, 3, 4\}, i \neq$

j. Što kada bi, umjesto razlike ta dva vektora, promatrali njihov kvocijent po točkama? Odnosno, međusobno podijeljene elemente na istim pozicijama tih vektora. Tada bi imali informaciju koliko se puta više neka četvorka pojavila u nekoj familiji. Zašto bi nam to pomoglo? Zamislimo ovakvu potencijalnu situaciju. Neka četvorka se u jednoj familiji pojavila sto puta, a u drugoj pedeset. S druge strane, neka druga četvorka se u prvoj familiji pojavila pedeset puta, a u drugoj pet. U prvoj situaciji razlika u pojavljivanjima je veća ali četvorke se relativno često pojavljuju u obje familije, dok se u drugom slučaju četvorka u jednoj familiji pojavila samo pet puta, a u drugoj deset puta više. Ovakav pristup će ipak zahtijevati neke male korekcije. Budući da ćemo sigurno naići na četvorke koje se u nekoj familiji nisu niti jednom pojavile, moramo nekako izbjeći dijeljenje s nulom. Tome ćemo doskočiti dodavanjem jedinice svakom elementu svakog vektora prije dijeljenja. Točnije, za neke vektore v_i i v_j promatrat ćemo zapravo kvocijent

$$v_{ij} = \frac{v_{i+1}}{v_{j+1}} = \begin{bmatrix} (v_{i1}+1)/(v_{j1}+1) \\ \vdots \\ (v_{ik}+1)/(v_{jk}+1) \\ \vdots \\ (v_{in}+1)/(v_{jn}+1) \end{bmatrix}$$

Na taj način izbjeći ćemo dijeljenje s nulom, a opet nećemo puno poremetiti omjere. Sada kad je jasno kako dobivamo vektore v_{ij}, $\quad i, j \in \{1,2,3,4\}, i \neq j$, jednako kao u postpku ranije dolazimo do skupova B_{ij}, a onda presjecima i do konačnih baza za familije. Također, na isti način provodimo validaciju modela i dolazimo do sljedećih rezultata:

Familija 1			
100	0	0	0
Familija 2			
42	57	0	1
Familija 3			
60	2	38	0
Familija 4			
35	1	2	62

Tablica 3.5: Rezultati klasifikacije promatrajući kvocijene broja pojavljivanja četvorki

Što u sažetku daje konačan broj pogodaka i promašaja:

Familija	**Točno**	**Netočno**
Familija 1	100	0
Familija 2	57	43
Familija 3	38	62
Familija 4	62	38

Tablica 3.6: Točnost modela

Nažalost, rezultati su ovaj put još lošiji. Čini se da razlika u pojavljivanjima četvorki ipak donosi bolje rezultate od kvocijenta.

Što kada bi, umjesto blokovima od četiri znaka, proteinske familije pokušali opisati uređenim trojkama znakova? Broj svih različitih mogućnosti za uređene trojke znakova je manji pa će možda i njihova raspršenost među familijama biti manja. Ovu teoriju testirat ćemo na setu od šest proteinskih familija. Podaci su kao i ranije odvojeni u trening set u kojem je po tristo proteina iz svake familije, grupiranih tako da su proteini iz iste familije jedan do drugog. Također, u test setu je oko sto proteina iz svake familije. Prvo ćemo napraviti standardni postupak kao i u prethodnom slučaju, promatrajući uređene četvorke znakova u proteinima. Nakon toga izvući ćemo sve različite trojke znakova iz svih proteina test seta i napraviti identičan postupak. Sve radimo na jednak način samo što u drugom procesu baratamo drugom bazom, onom svih različitih uređenih trojki. Budući da smo sada već bolje upoznati s postupkom i nema nikakvih tehničkih promjena, izlaganje možemo malo ubrzati i odmah prijeći na rezultate. Najprije iznosimo rezultate koje smo dobili promatranjem pojavljivanja uređenih četvorki.

Familija 1					
99	0	0	1	0	0
Familija 2					
1	104	4	1	0	1
Familija 3					
1	2	90	0	3	4
Familija 4					
2	0	4	98	0	2
Familija 5					
2	3	8	4	82	2
Familija 6					
0	3	8	2	2	85

Tablica 3.7: Rezultati promatranja uređenih četvorki

Zanimljivo je primijetiti da su ovdje rezultati bolji nego za prethodne četiri familije. Na to bi mogla utjecati građa proteina iz familija.

Pogledajmo sad kakve je rezultate dalo promatranje pojavljivanja uređenih trojki znakova.

Familija 1					
69	0	0	1	1	29
Familija 2					
0	89	6	0	0	16
Familija 3					
0	1	75	0	1	23
Familija 4					
0	1	1	78	0	26
Familija 5					
0	0	9	0	80	12
Familija 6					
0	0	0	0	0	100

Tablica 3.8: Rezultati promatranja uređenih trojki

I dalje uređene četvorke daju bolje rezultate. Jedina zanimljivost ovdje je da je model koji je promatrao pojavljivanja uređenih trojki dao odlične rezultate za šestu familiju. Sve proteine te familije točno je klasificirao. Primijetimo još da model, kada pogriješi u klasifikaciji, uglavnom umjesto stvarnoj familiji protein pridruži familiji 6. Slično kao što je bilo s prvom familijom u prošlom primjeru.

3.2 Najčešće pojavljivani n-grami

Što bi još mogli napraviti kako bi poboljšali rješenje? Izvrsno bi bilo kada bi bazu za neku familiju proteina mogli dobiti promatrajući samo proteine te familije, bez da ih uspoređujemo s drugim familijama i brojimo koliko se koja četvorka znakova gdje pojavila. Takvo rješenje bi u praksi sigurno bilo brže. Iako možda intuicija sumnja u takav pristup jer jasno je da se iste četvorke mogu često pojavljivati u više proteinskih familija, krenimo u tom smjeru. Promatrat ćemo iste podatke kao u problemu prije. Iste proteine iz istih šest familija, jednako podijeljene na trening i test set podataka. Početni koraci su uvijek isti. Tu pronalazimo sve različite uređene četvorke iz trening seta podataka, skupljamo ih u bazu, potom ponovno prolazimo kroz protein i brojimo koliko se puta koja četvorka iz baze u kojem proteinu pojavila. Time zapravo proteinima pridružujemo vektore koje kao stupce postavljamo u matricu tako da su oni koji predstavljaju proteine iste familije jedan

do drugog. Slijedi zbrajajne stupaca matrice koji su došli iz iste familije. Time dobijemo vektore koji govore koliko se ukupno puta koja četvorka iz baze pojavila u proteinima neke familije. Sve razlike u procesima koje radimo događaju se na ovom mjestu. To jest, naši pristupi se razlikuju, uglavnom, prema onome što sada radimo s tim vektorima kako bi dobili baze uređenih četvorki kojima pokušavamo opisati promatrane familije. Sjetimo se da je ideja pokušati dobiti takve baze promatranjem samo odgovarajuće familije, one za koju želimo naći bazu. Bez međusobnog uspoređivanja familija. Pokušajmo za svaku četvorku promatrati familije koje su se najviše puta pojavile. Želja nam je naći neku granicu i sve četvorke čiji je broj pojavljivanja veći od te granice smatramo bazom za tu familiju. Postavlja se pitanje kako odrediti tu granicu i koliko bi takve baze elemenata imale? Naš pristup je promatrati k, $k \in \mathbb{N}$ najčešće pojavljivanih četvorki i naći nekakvu mjeru srednje vrijednosti koju ćemo postaviti za traženu granicu. Pokušajmo koristeći medijan, mjeru srednje vrijednosti definiranu u drugom poglavlju.. Za ilustraciju, pogledajmo kako dolazimo do baze za prvu familiju proteina koju promatramo. Zbrajanjem stupaca, od prvog do tristotog, matrice M dolazimo do vektora v_1. Promatramo $k, k \in \mathbb{N}$ njegovih najvećih elemenata. Medijan tog skupa označimo s m_1 i sve elemente vektora v_1 veće od m_1 pohranjujemo u bazu B_1 za prvu familiju. Ovime nije odmah jasno koliko će takva baza imati elemenata ali to nam nije važno. To što dobivene baze vjerojatno neće biti jednakobrojne nam neće smetati. Isti postupak ponavljamo za ostale tri familije i dolazimo do baza B_2, B_3 i B_4. Validaciju modela opet radimo na isti način, zapisivanjem svakog proteina iz test seta u bazama i traženja vektora s najvećom euklidsokm normom. Protein potom pridružujemo familiji u čijoj bazi njemu pridružen vektor ima najveću normu. Sada je pitanje koliko velik k uzeti? Eksperimentalnim metodama došli smo do zaključka da se najbolji rezultati postižu za $k = 700$. To što je rješenje nastalo eksperimentalnom metodom jest mala mana ali tome ćemo doskočiti u sljedećem koraku. U nastavku prikazujemo rezultate za $k \in \{400, 700, 1000\}$

Familija 1					
100	0	0	0	0	0
Familija 2					
2	108	0	1	0	0
Familija 3					
0	3	97	0	0	0
Familija 4					
1	2	0	103	0	0
Familija 5					
1	2	3	2	92	1
Familija 6					
4	1	5	2	1	87

Tablica 3.9: Rezultati za četvorke čiji je broj pojavljivanja veći od medijana najvećih sedamsto elemenata

Familija 1					
100	0	0	0	0	0
Familija 2					
3	107	0	0	0	1
Familija 3					
0	2	96	0	0	2
Familija 4					
3	2	2	97	1	1
Familija 5					
2	1	6	1	90	1
Familija 6					
5	2	3	1	1	88

Tablica 3.10: Rezultati za četvorke čiji je broj pojavljivanja veći od medijana najvećih četiristo elemenata

Familija 1					
100	0	0	0	0	0
Familija 2					
2	108	0	1	0	0
Familija 3					
1	4	95	0	0	0
Familija 4					
1	2	1	102	0	0
Familija 5					
0	1	1	4	93	2
Familija 6					
4	1	3	3	2	87

Tablica 3.11: Rezultati za četvorke čiji je broj pojavljivanja veći od medijana najvećih tisuću elemenata

Ovakvi rezultati su na tragu onoga što bismo htjeli. U sva tri slučaja klasifikacija je vrlo dobra. Rješenje za $k = 700$ proglasili smo najboljim jer je broj točno klasificiranih proteina najveći.

Ranije smo spomenuli da ovakvo rješenje ima jednu manu. Zašto uzimamo baš $k = 700$? Zašto ne neki drugi broj? Naglasili smo kako je do toga došlo eksperimentalnim metodama, što nije poželjno jer nije ekonomično svaki puta pokretati model za različite vrijednosti parametara kako bi pronašli najbolju kombinaciju. Zato u nastavku donosimo još jednu inačicu rješenja kojom smo taj problem izbjegli.

Kvantili

I dalje se bavimo klasifikacijom istih šest proteinskih familija koje smo promatrali u gornjem primjeru. Budući da radimo sa istim trening i test podacima, izlaganje možemo malo ubrzati. Sjetimo se da već imamo izračunate vektore u kojima su podaci o ukupnom pojavljivanju uređenih četvorki u svakoj od familija. Za bazu kojom pokušavamo opisati familiju, ideja je uzeti neki dio najčešće pojavljivanih uređenih četvorki u toj familiji. Pokušat ćemo s 1% najčešće javljanih uređenih četvorki pojedine familije. Za usporedbu, napravit ćemo i postupak s malo većim udjelom, odnosno s 5% najčešćih četvorki u svakoj familiji. U tu svrhu koristit ćemo 95%-kvantil i 99%-kvantil na podacima iz vektora v_i. Taj postupak će nam dati tražene najčešće javljane četvorke u i-toj familiji, koje onda smatramo bazom kojom opisujemo tu familiju. Nakon što smo baze B_1, B_2, B_3 i B_4 definirali na taj način, validacijom modela kao i ranije, dolazimo do zapanjujućih rezultata.

Familija 1					
100	0	0	0	0	0
Familija 2					
0	**109**	1	1	0	0
Familija 3					
0	0	**98**	1	1	0
Familija 4					
0	1	1	**104**	0	0
Familija 5					
1	1	0	0	**97**	2
Familija 6					
1	0	3	1	0	**95**

Tablica 3.12: 95%-kvantil

Udio točno klasificiranih proteina vrlo je visok za svaku familiju. Najslabiji postotak točno klasificiranih proteina došao je iz familije 6, a iznosi 95%. U praksi se to smatra dobrim rezultatom, tako da dobivene rezultate možemo proglasiti jako dobrima.

Familija 1					
100	0	0	0	0	0
Familija 2					
0	**109**	1	1	0	0
Familija 3					
0	2	**97**	1	0	0
Familija 4					
0	0	0	**106**	0	0
Familija 5					
0	0	0	1	**95**	5
Familija 6					
1	1	2	2	0	**94**

Tablica 3.13: 99%-kvantil

Rješenje temeljeno na promatranju 1% najčešćih uređenih četvorki za svaku familiju dalo je za nijansu lošije ali i dalje zadovoljavajuće rezultate. Ovim pristupom dobivene baze imaju manje elemenata pa je redukcija dimenzije u odnosu na početnu bazu svih mogućih uređenih trojki još značajnija nego li u gornjoj inačici rješenja. Prema tome, u situaciji kada nam je u interesu da potprostor u koji preslikavamo proteine bude što manji,

možemo se odlučiti na ovakav pristup.

3.3 Klasifikacija većeg broja familija

Rješenje koje smo upravo dobili dalo je vrlo dobre rezultate ali za ovaj, ne pretjerano velik skup podataka. Uspješno smo klasificirali četiri familije proteina. No, u stvarnosti često se javlja potreba za razlikovanjem većeg broja familija. Nameće se pitanje kakve bi rezultate naš model pružio u takvoj situaciji? Na našu sreću, *Pfam* je ogromna baza različitih proteinskih familija pa to možemo vrlo lako provjeriti.

Izabrali smo deset različitih proteinskih familija i kreirali trening i test set podataka, na sličan način kao i svaki puta do sada. U trening set stavili smo po tristo proteina svake familije, a po sto ostavljamo za validaciju rješenja i stavljamo u test set. Model je izgrađen na jednak način i ponovno je dao dobre rezultate koje navodimo u sljedećim tablicama.

Familija 1									
100	0	0	0	0	0	0	0	0	0
Familija 2									
0	**98**	0	1	0	0	0	1	0	0
Familija 3									
0	1	**96**	2	0	0	0	1	0	0
Familija 4									
0	1	0	**99**	0	0	0	0	0	0
Familija 5									
0	1	0	0	**93**	5	0	0	1	0
Familija 6									
1	1	1	1	0	**96**	0	0	0	0
Familija 7									
0	0	0	0	0	0	**100**	0	0	0
Familija 8									
0	0	0	0	0	1	0	**99**	0	0
Familija 9									
0	0	0	2	0	1	0	0	**97**	0
Familija 10									
0	1	0	0	0	0	0	0	0	**99**

Tablica 3.14: 95%-kvantil

Familija 1									
100	0	0	0	0	0	0	0	0	0
Familija 2									
0	**98**	0	1	0	0	0	1	0	0
Familija 3									
0	1	**96**	2	0	0	0	1	0	0
Familija 4									
0	1	0	**99**	0	0	0	0	0	0
Familija 5									
0	1	0	0	**93**	5	0	0	1	0
Familija 6									
1	1	1	1	0	**96**	0	0	0	0
Familija 7									
0	0	0	0	0	0	**100**	0	0	0
Familija 8									
0	0	0	0	0	1	0	**99**	0	0
Familija 9									
0	0	0	2	0	1	0	0	**97**	0
Familija 10									
0	1	0	0	0	0	0	0	0	**99**

Tablica 3.15: 99%-kvantil

Pomalo iznenađujuće, rezultati su i dalje vrlo dobri. Primjećujemo da su promatranja 95%-kvantila i 99%-kvantila dala iste rezultate. Sve familije, osim jedne, prepoznate su s visokom razinom točnosti. Najslabiji rezultat dala je klasifikacija proteina pete familije koji su prepoznati u 93% slučajeva, što je i dalje dobar rezultat. Do modela smo došli na vrlo elegantan način, bez promatranja i uspoređivanja više familija istovremeno. Imajući to sve na umu, ovu metodu možemo proglasiti uspješnom.

Poglavlje 4

Zaključak

Na kraju svih ovih razmatranja došli smo do iznenađujuće dobrih rezultata. Pokušaj razlikovanja proteinskih familija pomoću blokova aminokiselina duljine tri nije se pokazao uspješnim te smo vrlo brzo odustali od takvog pristupa. Nastavili smo analizu proteina kao skupa uređenih četvorki znakova aminokiselinskog alfabeta. Cijelo vrijeme smo, zapravo, pokušavali pronaći efikasan načine izgradnje baza takvih četvorki za svaku familiju s ciljem da one što bolje razlikuju promatrane proteinske familije. Najboljim pristupom pokazao se, jednostavno, odabir najfrekventnijih uređenih četvorki aminokiselina. Došli smo do modela koji je, na našim podacima, isporučivao klasifikaciju proteina u proteinske familije s točnošću od barem 93%. Ovakvi rezultati su utoliko zanimljiviji jer su dobiveni bez ikakvog prethodno iskorištenog ekspertnog znanja o preteinskim familijama ili aminokiselinama.

Bibliografija

[1] G. Salton, C.Buckley *Term-weighting approaches in automatic text retrieval*, In Information Processing and Managment, Volume 24, Issue 5, 1988.

[2] D. Bakić, *Linearna Algebra*, `https://web.math.pmf.unizg.hr/~bakic/la/linearna_algebra_sk_7.pdf`, Sveučilište u Zagrebu, Matematički odjel PMF-a, Zagreb 2008.

[3] B. Y. M. Cheng, J. G. Carbonell, J. Klein-Seetharaman, *Protein classification based on text document classification techniques*, Volume 58, Issue 4, 2005.

[4] S. M. A. Islam, B. J. Heil, C. M. Kearney, E. J. Baker, *Protein classification using modified n-grams and skip-grams*, Volume 34, Issue 9, 2018.

Sažetak

Potreba za točnim, automatiziranim metodama klasifikacije nastavlja se povećavati kako napredak biotehnologije otkriva nove proteine. U radu je provedeno nekoliko analiza klasifikacije proteina u proteinske familije, koristeći vrlo jednostavan pristup preuzet iz semantičkog indeksiranja. Naglasak je bio na metodama identifikacije diskriminativnih uređenih četvorki aminokiselina s ciljem što boljeg razlikovanja familija proteina. Uz vrlo malu razinu prethodno iskorištenog biološkog znanja dobiveni su iznenađujuće dobri rezultati.

Summary

The need for accurate, automated protein classification methods continues to increase as advances in biotechnology uncover new proteins. We carried out protein classification by several methods, using a very simple approach taken from semantic indexing. The emphasis was on identifying discriminating blocks of amino acids to distinguish between various protein families. Our classification results are surprisingly good, especially considering no expert knowledge was used in the process.

Životopis

Rođen sam u Našicama, 13. rujna 1993. godine. U istom mjestu, u Srednjoj školi Isidora Kršnjavoga Našice, završio sam Prirodoslovno - matematičku gimnaziju nakon koje, 2012. godine upisujem Preddiplomski sveučilišni studij Matematika na Prirodoslovno - matematičkom fakultetu Sveučilišta u Zagrebu. Na istom fakultetu 2016. godine upisujem Diplomski sveučilišni studij Matematička statistika.