Višerezolucijska detekcija anomalija korištenjem difuzijskih preslikavanja

Doris Đivanović i Karlo Gjogolović

7. svibnja 2025.

Uvod

U raznim problemima koji se odnose na analizu podataka, podaci iz stvarnog svijeta često su velikih dimenzija, pa je s njima teško raditi. Zbog toga je korisno pronaći eventualni način za preslikati takve podatke u neki prostor puno manje dimenzije, odnosno reprezentirati ih podacima iz takvog prostora. Tada, umjesto s originalnim, možemo raditi s tim novodobivenim podacima, a, zbog male dimenzije, to je puno lakše i praktičnije, što u smislu vizualizacije i intuicije, što u smislu veće efikasnosti računskih operacija nad njima.

Standardna klasa problema u analizi podataka su problemi klasteriranja. Jedan takav problem je i problem detekcije anomalija u podacima. U takvim problemima, pretpostavlja se da su podaci koji su anomalija daleko od ostalih normalnih podataka, i njih želimo odvojiti u zaseban klaster. Za probleme klasteriranja veliku važnost igraju koncepti povezanosti, sličnosti ili bliskosti podataka. Stoga bi nam bilo korisno kada bismo podatke velike dimenzije mogli reprezentirati nižedimenzionalnim podacima preslikavanjem koje čuva bliskost, tj. udaljenost među podacima.

Kreiranje/konstrukcija težinskog grafa skupa originalnih podataka

Neka je $\Gamma = \{x_1, ..., x_n\}$ skup od n visokodimenzionalnih točaka podataka. Konstruira se težinski graf gdje su čvorovi točke podataka x_i, a težina brida $w(x_i, x_j)$ mjeri sličnost između točaka x_i i x_j. Graf je usmjeren, tezinska funkcija je mjera slicnosti (afiniteta) izmedu dvaju podataka, stoga je logicno da je simetricna i nenegativna, nadalje logicno je da ako uzmemo da je ≤ 1 (tj. $\in [0, 1]$), da je je za (svaki) sami podatak $= 1$, jer je on potpuno slican sam sebi. Prirodno je da slicnost dvaju podataka povežemo s njihovom medusobnom (npr. Euklidskom) udaljenosti, odnosno standardno (samo ako je Euklidska?) njezinim kvadratom, i to da je slicnost veca sto je udaljenost manja, tj. da su obrnuto proporcionalne. Udaljenost je također kao takva mjera svakako simetrična i nenegativna, a ta će svojstva ostati ukoliko za tezinsku fju uzmemo $1/e^{\text{kvadrat udaljenosti}}$, tj. $e^{-\text{kvadrat udaljenosti}}$, a time će biti da je za sam podatak mjera slicnosti $= 1$, jer je udaljenost do samog sebe $= 0$, te da su obrnuto proporcionalne. Funkcija $e^{\text{kvadrat udaljenosti}}$ je ≥ 1, pa je $\frac{1}{e^{\text{kvadrat udaljenosti}}} \leq 1$. Ovakva bi tezinska funkcija za svaka dva podatka ovisila isključivo/samo o međusobnoj euklidskoj udaljenosti bas ta dva podatka, dakle nikako ne bi ovisila o ostatku skupa podataka. Nadalje, tezinsku funkciju bismo mozda htjeli skalirati nekim odgovarajućim faktorom. Možemo za taj faktor uzeti neki varijabilni parametar, pa gledati njegov utjecaj s obzirom na njegove razlicite vrijednosti. Ako želimo da težinska funkcija bude proporcionalna tom parametru, recimo da je to $\sigma > 0$, tezinsku fju mozemo definirati ovako: $e^{-\text{kvadrat udaljenosti}/\sigma}$, tj. ako zelimo da oba podatka doprinose tezini s tim faktorom, ovako:

$$w(x_i, x_j) = e^{\frac{-||x_i - x_j||^2}{\sigma^2}}. \tag{1}$$

Ova fja je vrlo poznata i u literaturi se naziva **gaussovska ili Gaussova jezgra**. Ovime taj faktor σ skalira, tj. povećava ili umanjuje utjecaj euklidske udaljnosti dvaju podataka na mjeru njihove slicnosti. Ključno ga je empirijski odrediti tako da je umanjuje ili uvecava bas kad treba/onda kad je logicno. Budući da je vrijednost ovakve tezinske funkcije proporcionalna vrijednosti parametra σ, ako za σ uzmemo neku jako malu vrijednost, većina će težina u grafu biti jako mala - u smislu jako bliska 0, osim težine svakog podatka za samog sebe, koja je, neovisno o parametru σ, uvijek $= 1$, pa bismo dobili graf koji je jako nepovezan, tj. u kojem je većina podataka povezana samo sa sobom, tj. jedini njihov susjed su oni sami sebi. Analogno, ako za σ uzmemo neku jako veliku vrijednost, većina će težina u grafu biti jako velika - u smislu jako bliska 1, pa bismo dobili graf u kojem su gotovo svi podaci međusobno povezani. To je loše za detekciju anomalija bla, bla, ...PISE bas argument zasto...–ono dolje o gustoj i riejtkoj naseljenosti... Ako hoćemo neku "usrednjenu" vrijednost za taj parametar, možemo, na primjer, uzeti

medijan svih međusobnih udaljenosti podataka u skupu, ili, na primjer, standarndu devijaciju udaljenosti (tada je σ^2 varijanca). Također, budući da je takav σ "globalan" za skup podataka, tj. ovisi o svim međusobnim udaljenostima, time bismo postigli i da tezinska funkcija ovisi o međusobnim udaljenostima svih podataka u skupu. Međutim, tada bi za svaka dva podatka njihova težina ovisila o ostalim podacima na isti način. Možda bi bilo dobro skalirati tezinu izmedu dva podatka nekim faktorom koji ovisi o bas tim konkretnim podacima. Budući da za detekciju anomalije zelimo da mjera skicnosti oslikava lokalnu geometriju, stovise takvim faktorom koji odrazava lokalna svojstva tih podataka (ovu recenicu bolje napisati). Mozemo definirati preslikavanje koje svakom podatku pridruzuje faktor koji ovisi samo o njegovoj maloj okolini, a ne cijelom skupu podataka, te definirati tezinsku fju tako da oba podatka doprinose sa svojim faktorom, dakle ovako: $e^{-\text{kvadrat udaljenosti}/\sigma_i\sigma_j}$. ((Mala okolina podatka bi npr. bila odredena s k najblizih susjeda podatka, u smislu euklidske udaljenosti.)) U literaturi za ovaj seminarski rad koristi se preslikavanje koje svakom podatku x_i pridruzuje njegovu euklidsku udaljenost do njegovog K-tog najblizeg susjeda: $\sigma_i = ||x_i - x_K||^2$. Dakle, ovime se postize da tezina za dva podatka u grafu kojeg kreiramo ovisi i o udaljenostima s ostalim podacima u skupu, ali tako da odrazava lokalnu geometriju tih dvaju podataka.

Zašto je baš to/takvo preslikavanje σ prikladno za problem detekcije anomalija u podacima, ili opcenito problem klasteriranja? Jer pretpostavljamo da se podaci koji nisu anomalija, odnosno u tom smislu "normalni" podaci, nalaze u "gusto naseljenom" podrucju, tj. nalaze se blizu vecine ostalih podataka iz skupa podataka, tj. vecina ostalih podataka im je bliska, susjedi su im, povezani su s njima, a podaci koji jesu anomalija nalaze se u "rijetko naseljenom" podrucju, odnosno daleko od skoro svih ostalih podataka iz skupa podataka. Lokalnu geometriju mozemo mjeriti/odrediti na vise nacina/pristupa (je li ?). Npr. mogli bismo unaprijed definirati neki radijus za okolinu podatka koju smatramo malom, te za svaki podatak odrediti broj susjeda u njegovoj bliskoj okolini. Drugi pristup je određivanjem k najbližih susjeda, čime za svaki podatak dobijemo radijus u kojem se nalazi njemu najbližih k susjeda. Iz toga da je za neki podatak taj radijus velik, tj. da je k-ti najbliži susjed jako udaljen, možemo zaključiti da su i svi preostali podaci iz skupa (dakle, preostalih $n-k-1$ podataka) jako udaljeni od njega, tj. barem više od tog radijusa, i da se on nalazi u rijetko naseljenoj okolini, dakle da je anomalija, a za podatak za koji je k-ti najbliži susjed jako blizu, možemo zaključiti da se nalazi u gusto naseljenoj okolini, odnosno da je "normalan" podatak koji pripada pozadini. Dakle, kada računamo tezinu, tj. mjeru slicnosti dvaju podataka, formulom $e^{-\text{kvadrat udaljenosti}/\sigma_i\sigma_j}$, $\sigma_i = ||x_i - x_K||^2$, uzimamo u obzir koliko je ostatak skupa podataka udaljen od svakog od ta dva podatka/njihov odnos s ostatkom skupa podataka, odnosno tezinska fja stvarno prenosi lokalnu geometriju. Očito, ako su i σ_i i σ_j veliki, dakle ako su i x_i i x_j jako udaljeni od ostatka skupa podataka, onda će i produkt $\sigma_i\sigma_j$ biti velik, pa će tezina/mjera sličnosti za ta dva podatka biti jako velika, što ima smisla, jer ako su oba daleko od ostatka skupa, oba vjerojatno pirpadaju anomaliji. Nadalje, ako su i σ_i i σ_j mali, tj. i x_i i x_j su gusto okruženi podacima, onda će i produkt $\sigma_i\sigma_j$ biti mali, pa će tezina/mjera sličnosti za ta dva podatka biti mala, što ima smisla, jer ako su oba podatka okružena drugim podacima, oba vjerojatno pirpadaju pozadini, i ne moraju biti bliski, jer gotovo svi podaci pripadaju pozadini, tj. pozadina zauzima većinu slike.

Normalizacija tezinske fje (tj jezgre) –> nova fja u dva podatka - vjerojatnost prijelaza iz jednog / prvog stanja u drugo –> slucajna setnja –> ta fja (svakako) kreira/definira kvadratnu matricu –> a matrica je retcano stohasticka –> matrica prijelaza Markovljeva lanca na skupu nasih podataka –> potencije te matrice su matrice ciji elementi predstavljaju vjerojatnosti prijelaza iz jednog stanja u drugo u odgovarajucem broju koraka

Spektralna dekompozicija matrice prijelaza Markovljeva lanca - lijevi i desni svojstveni vektori, padajuće svojtvene vrijednosti, jedna svojstvena vrijednost (najveca ?) je sigurno 1 za jedinicni

vektor - i samo za taj? (jer je matrica retcano stohasticka): ... -> svakom od n podataka pridruziti tezinsku sumu njegovih pripadnih lijevih vektora s odgovarajućim svojstvenim vrijednostima kao tezinama. Budući da je vektor za svojstvenu vriejdnost 1, oznaceno bas s λ_0, konstantan, izuzeti taj vektor. Nadalje, budući da svojstvene vrijednoti padaju ((*) i to brzo - zasto?), za dovoljnu tocnost sume dovoljno je uzeti u obzir samo prvih nekoliko svojestvenih vektora - neki l koji je ((*) ==> puno?) manji od n ===> bas ovo preslikavanje je dobar kandidat za redukciju dimenzije / ulaganje originalnih podataka velike dimenzije u prostor puno manje dimenzije, tj. za difuzijsko preslikavanje !!!!

Na temelju težina definira se slučajna šetnja na skupu podataka normalizacijom funkcije težine:

$$p(x,y) = \frac{w(x,y)}{d(x)}, \quad \text{gdje je } d(x) = \sum_{y \in \Gamma} w(x,y) \tag{2}$$

$p(x,y)$ predstavlja vjerojatnost prijelaza iz podatka x u podatak y u jednom koraku. Matrica P s elementima $p(x,y)$ je matrica prijelaza Markovljeva lanca na Γ. Potenciranje matrice P^t odgovara izvođenju lanca t koraka unaprijed, a $p_t(x,y)$ je vjerojatnost prijelaza iz x u y za t koraka.

Matrica P ima potpun skup biortogonalnih lijevih (ϕ_j) i desnih (ψ_j) svojstvenih vektora s pripadajućim svojstvenim vrijednostima λ_j, uređenim kao $1 = |\lambda_0| \geq |\lambda_1| \geq ....$ Spektralna dekompozicija od P^t (matrica prijelaza u t koraka) je:

$$p_t(x,y) = \sum_{l \geq 0} \lambda_l^t \psi_l(x) \phi_l(y) \tag{3}$$

Zbog brzog opadanja spektra svojstvenih vrijednosti, često je dovoljno samo prvih nekoliko članova za dobru aproksimaciju.

Difuzijsko preslikavanje u vremenu t definira se kao preslikavanje koje ulaže svaki podatak x u $\mathbb{R}^l$ koristeći prvih l netrivijalnih desnih svojstvenih vektora (svojstveni vektor ψ_0 je konstantan i ne koristi se):

$$\Psi_t : x \mapsto (\lambda_1^t \psi_1(x), \lambda_2^t \psi_2(x), ..., \lambda_l^t \psi_l(x))^T \tag{4}$$

Dimenzija novog prostora l ovisi o intrinzičnoj strukturi podataka, a ne o dimenzionalnosti originalnih podataka.

Difuzijska udaljenost $D_t(x,z)$ između točaka x i z mjeri sličnost njihovih distribucija vjerojatnosti nakon t vremenskih koraka:

$$D_t^2(x,z) = \sum_{y \in \Gamma} \frac{(p_t(x,y) - p_t(z,y))^2}{\phi_0(y)} \tag{5}$$

Intuitivno, difuzijska udaljenost je mala ako postoji mnogo kratkih puteva koji povezuju x i z u grafu. Ova metrika je robusna na šum jer ovisi o svim putevima duljine t između točaka, uzimajući u obzir cjelokupnu strukturu podataka. Može se izračunati i pomoću svojstvenih vektora:

$$D_t^2(x,z) = \sum_{j \geq 1} \lambda_j^{2t} (\psi_j(x) - \psi_j(z))^2 \tag{6}$$

Zbog opadanja spektra, udaljenost se može aproksimirati koristeći samo prvih l svojstvenih vektora, što je računski efikasno. Pokazuje se da je difuzijska udaljenost jednaka Euklidskoj udaljenosti u prostoru difuzijskog preslikavanja:

$$D_t^2(x,z) = ||\Psi_t(x) - \Psi_t(z)||^2 \tag{7}$$

Ovo svojstvo koristi se kasnije za definiranje mjere sličnosti u difuzijskim koordinatama. Svojstvo difuzijskih preslikavanja da grupiraju (klasteriraju) slične točke korisno je za detekciju anomalija, jer se očekuje da će podaci pozadine formirati guste klastere, dok će anomalije biti udaljene od njih.

0.1 Proširenje funkcija izvan uzorka

Za velike skupove podataka poput slika, računanje difuzijskog preslikavanja za sve podatke je nepraktično. Umjesto toga, preslikavanje se računa za podskup uzoraka $\Gamma \subseteq \overline{\Gamma}$, a zatim se ulaganje proširuje na sve podatke u $\overline{\Gamma}$ pomoću metoda proširenja izvan uzorka. Poznate metode su Nyströmova ekstenzija i geometrijske harmonike.

Ovaj rad koristi ekstenziju temeljenu na višerezolucijskoj **Laplaceovoj piramidi**. Na svakoj razini piramide l, konstruira se grublja aproksimacija funkcije f (u našem slučaju, jedne difuzijske koordinate Ψ_j) pomoću Gaussove jezgre W_l s progresivno manjim fakrtorom skaliranja $\epsilon_l = \epsilon_0/2^l$:

$$W_l(x_i, x_j) = \exp(-||x_i - x_j||^2/\epsilon_l) \tag{8}$$

$$K_l = q_l^{-1} W_l, \quad q_l(x_i) = \sum_j W_l(x_i, x_j) \tag{9}$$

Laplaceova piramidalna reprezentacija funkcije f definirana je iterativno po komponentama s_l:

$$s_0(x_k) = \sum_{i=1}^{n} K_0(x_i, x_k) f(x_i) \tag{10}$$

$$s_l(x_k) = \sum_{i=1}^{n} K_l(x_i, x_k) d_l(x_i), \quad l \geq 1 \tag{11}$$

$$d_l(x_k) = f(x_k) - \sum_{m=0}^{l-1} s_m(x_k), \quad l \geq 1 \tag{12}$$

gdje d_l predstavlja razliku (detalje) na razini l. Piramida se gradi dok pogreška aproksimacije ne padne ispod zadanog praga. Proširenje funkcije f na novu točku $\overline{x}_k \in \overline{\Gamma}$ dobiva se sumiranjem proširenja komponenti s_l:

$$f(\overline{x}_k) = \sum_{l \geq 0} s_l(\overline{x}_k), \quad \text{gdje se } s_l(\overline{x}_k) \text{ računaju analogno formulama 10-12.} \tag{13}$$

Ovo proširenje se provodi za svaku difuzijsku koordinatu Ψ_j zasebno. Glatke koordinate zahtijevaju manje razina piramide (grublje skale), dok oscilirajuće koordinate zahtijevaju više razina (finije skale).

Primjena

Cilj nam je na digitalnoj slici detektirati dijelove koji odskaču od ostatka slike, odnosno od pozadine slike, tj. koji su u određenom smislu anomalija u donosu na ostatak. Pokazat ćemo na koji se način takav problem svodi na problem klasteriranja podataka velike dimenzije koristeći ulaganje tih podataka u potprostor puno manje dimenzije.

Podaci

Ulaz našeg algoritma je digitalna slika nekih dimenzija u pikselima, slikovni objekt, objekt tipa *Image*. Mi ćemo promatrati 'grayscale' digitalne slike. Takva slika učitana u Matlab funkcijom **imread** je pravokutna matrica cjelobrojnih elemenata iz intervala [0, 255] čije vrijednosti reprezentiraju intenzitet sive boje odgovarajućeg piksela slike. Takvu ćemo matricu najprije normalizirati, da bismo kao sliku promatrali matricu realnih brojeva iz intervala [0,1]. Podaci među kojima ćemo pokušati detektirati one koji su anomalija, odnosno klasterirati ih u klaster zaseban od ostalih podataka, neće biti pojedinačni pikseli slike, tj. elementi matrice, već

kvadratne blok-matrice (podmatrice) odabrane dimenzije. Dimenziju pojedinog bloka postavit ćemo kao parametar **patchDim**. Takve ćemo pod-matrice vektorizirati, dakle primijeniti na njima tzv. *flattening*. Time će podaci koje promatarmo biti vektori realnih vrijednosti duljine koja je kvadrat dimenzije bloka, dakle $\text{patchDim}^2 = m$, dakle vektori iz prostora $\mathbb{R}^m$, dakle stvarno podaci "velike dimenzije".

Skup ovakvih podataka iz naše ćemo matrice dobiti izdvajanjem svih mogućih preklapajućih kvadratnih blokova dimenzije patchDim na način da prolazimo svim elementima matrice po retcima te uzimamo blok dimenzije patchDim za kojeg je promatrani element središte. Budući da takvi blokovi nisu definirani, tj. nije ih moguće izdvojiti iz matrice za sve elemente koji su od nekog ruba matrice udaljeni za manje od patchDim, umjesto matrice koja predstavlja sliku, promatrat ćemo matricu nastalu njezinim proširivanjem za floor(patchDim/2) elemenata od svakog ruba prema van. Proširivanje znači da će elementi unutar tog okvira imati jednake vrijednosti kao elementi početne matrice, a za elemente okvira definirat ćemo neke vrijednosti. Za ovakve pothvate postoji Matlab funkcija **padarray** koja kao argumente uzima originalnu matricu, duljine proširenja matrice izvan donjeg i gornjeg ruba, te izvan lijevog i desnog ruba, te način proširivanja, tj. definiranja vrijednosti elemenata "okvira". Za posljednje postoje tri opcije: 'constant', koja sve elemente okvira postavlja na nulu, 'replicate', koja elementima "okvira" pridružuje vrijednosti koje se nalaze na rubovima početne matrice, te 'symmetric', koja na okvir zrcali elemente originalne matrice. Za našu se primjenu najprirodnija čini treća opcija. Primijetimo sada da prolazeći po svim indeksima redaka i stupaca originalne matrice, dakle prolazeći središtima podblokova, prolazimo po indeksima redaka i stupaca proširene matrice kao gornjim lijevim vrhovima odgovarajućih podblokova, pa možemo istovremeno izdvajati odgovarajuće blokove podataka iz proširene matrice i koordinate njihovih središta unutar originalne matrice. Svaki od ovih blokova vektoriziramo i opisanim ih redoslijedom spremamo kao retke jedne matrice. Širina te matrice je, dakle, duljina vektora-podataka, dakle patchDim^2, a visina je ukupan broj mogućih ovakvih pod-blokova matrice, dakle broj elemenata originalne matrice. Prilikom ovog postupka, kreiramo također i matricu koordinata središta svakog podbloka, na način da odgovarajućim redoslijedom koordinate spremamo u retke te matrice. Ovime pamtimo koji podaci odgovaraju kojem središtu, tj. pamtimo gdje se određeni pod-blok nalazi u početnoj (odnosno proširenoj) matrici.

Ovo naravno nije jedini mogući način za dobivanje skupa preklapajućih podblokova matrice. Jedan mogući način, bez kreiranja proširene matrice, je sljedeći: Krenemo od gornjeg lijevog elementa matrice. Taj će element biti gornji lijevi vrh prvog bloka dimenzije patchDim. Zatim ćemo se pomicati udesno duž retka matrice i time dobivati blokove čiji su gornji lijevi vrhovi odgovarajući elementi. Ovo ćemo činiti dok god je na taj način iz matrice moguće izdvojiti blok željene dimenzije, dakle do elementa na indeksu jednakom širini matrice, tj. broju stupaca matrice, umanjenoj za patchDim. Zatim prelazimo u redak ispod i krećemo s analognim postukom duž tog retka. U smjeru odozgo prema dolje, analogno, krećemo se dok god je iz matrice moguće izdvojiti blok željene dimenzije, dakle do elementa na indeksu jednakom visini matrice, tj. broju redaka matrice, umanjenoj za patchDim. Svaki od ovih blokova vektoriziramo i opisanim ih redoslijedom spremamo kao stupce jedne matrice. Visina te matrice je, dakle, duljina vektora-podataka, dakle patchDim^2, a širina je ukupan broj mogućih ovakvih pod-blokova matrice, koji se lako izračuna. Prilikom ovog postupka, kreiramo također i matricu koordinata gornjeg lijevog vrha svakog podbloka, na način da odgovarajučim redoslijedom koordinate spremamo u retke te matrice. Ovime, analogno, pamtimo koji podaci odgovaraju kojem gornjem lijevom vrhu, tj. ponovno pamtimo gdje se određeni podblok nalazi u početnoj matrici.

Računanje difuzijskog preslikavanja

Kako bismo smanjili složenost računanja, difuzijsko preslikavanje nećemo računati na cijelom skupu podataka, dakle na svim mogućim podblokovima matrice, odnosno dobivenim vektorima. Umjesto toga, definirat ćemo neki postotak, tj. udio, ukupnog broja podataka za koje ćemo

vrijednosti difuzijskog preslikavanja egzaktno izračunati, a zatim na slučajan način izdvojiti toliko podataka iz skupa. To ćemo implementirati na način da odabrani postotak ukupnog broja podataka zaokružimo na cijeli broj, a zatim na slučajan način permutiramo indekse redaka matrice s podacima, te odaberemo dobiveni broj prvih elemenata permutacije. U Matlab-u to radimo s pomoću metoda **round** i **randperm**. Iz matrice svih podataka izdvojit ćemo odgovarajuće retke u novu matricu koja će predstavljati skup podataka na kojem računamo difuzijsko preslikavanje, ali i zapamtiti koje smo točno podatke odabrali, tj. indekse redaka matrice svih podataka.

Sada za odabrani skup podataka računamo matricu sličnosti (W), odnosno afiniteta. Matricu ćemo računati kako je opisano u teorijskom uvodu. Naime, opisano preslikavanje

$$\sigma_i = ||x_i - x_k||_2^2,$$

gdje je x_i podatak, a x_k njegov k-ti najbliži susjed, prikladno je upravo za naš problem klasteriranja, detekciju anomalije na slikama, u kojem podatke koji su anomalija pretpostavljamo (jako) udaljenim od svih ostalih podataka u skupu, koji pripadaju normalnoj "pozadini" na slici. Budući da su Matlab-metode koje ćemo koristiti pri računanju difuzijskog preslikavanja posebno efikasne za rijetke matrice (eng. *sparse*), što su matrice kojima je većina elemenata jednaka nuli, matricu sličnosti kreirat ćemo kao takvu. To ćemo postići na način da na gore opisani način nećemo računati težine, tj. međusobne mjere sličnosti, za sve promatrane podatke, čime bismo dobili *punu* matricu, već ćemo za svaki podatak iz skupa najprije pronaći njegovih K najbližih susjeda, te njihove međusobne sličnosti izračunati po gore opisanoj formuli, gdje će broj najbližih susjeda k za preslikavanje σ biti (dosta) manji od K. Budući da je težinska funckija, tj. jezgra, simetrična, vrijednost težine za neki podatak i neki njegov najbliži susjed automatski će definirati i vrijednost težine za taj susjedni podatak i podatak kojeg promatramo, tj. i taj element matrice sličnosti. Sve preostale ovim postupkom neobuhvaćene elemente matrice sličnosti, tj. mjere sličnosti dvaju podataka, postavit ćemo na 0. Za dovoljno velik parametar K, ovakva defincija jezgrene funkcije i dalje će dovoljno dobro oslikavati i globalnu i lokalnu geometriju podataka. U Matlab-u je rijetku matricu poželjno alocirati funkcijom **sparse(...)**, koja na vrlo prostorno štedljiv način reprezentira rijetku matricu u memoriji računala, također takav da je upotreba Matlab funkcija na takvoj strukturi još efikasnija. Traženje najbližih susjeda možemo provesti efikasnim Matlab funkcijama **knnsearch** ili **pdist2**, pri čemu pdist2 računa sve međusobne udaljenosti podataka i sprema ih u kvadratnu matricu, pa, da bismo za svaki podatak izdvojili k najmanjih udaljenosti, a time i indekse pripadnih susjeda, moramo retke dobivene matrice sortirati uzlazno i izdvojiti skup stupaca od drugog do $k-1$-og. Funkcija knnsearch sve to radi za nas.

Nakon što smo izračunali matricu sličnosti, možemo izračunati matricu P_s, čiju spektralnu dekompoziciju koristimo za definiciju difuzijskog preslikavanja. Budući da je

$$P_s = D^{-1/2} W D^{-1/2}$$

, gdje je W dobivena matrica sličnosti. Matrica $D^{-1/2}$ očito se može dobiti računanjem suma svih redaka matrice W te kreiranjem dijagonalne matrice od vektora čiji su elementi redom dobivene sume, ali na koje je primijenjen drugi korijen, te su invertirane. To možemo napraviti Matlab funkcijom **spdiags(...)**. Spektralnu dekompoziciju matrice možemo izračunati efikasnom Matlab funkcijom **eigs(..)**, takvom (s argumnetom 'LM' - *largest magnitude*) koja vraća podskup vodećih svojstvenih vektora i svojstvenih vrijednosti. Ona kao argument prima željeni broj svojtvenih parova. Željenu dimenziju potprostora manje dimenzije u koji želimo uložiti podatke, dakle broj vodećih svojstvenih vektora podataka koje uzimamo u obzir, također želimo na neki način kontrolirati. Zbog toga ćemo tu dimenziju definirati kao parametar *maxInd*. Kako podatke želimo uložiti u potprostor neke puno manje dimenzije, poželjno ga je definirati kao minimum dimenzije originalnih podataka i nekog broja kojeg smatramo zadovoljavajuće malim, npr. 7, ili npr. 10. Budući da metoda eigs vraća matricu vodećih lijevih svojstvenih vektora

posloženih redom po stupcima, te dijagonalnu matricu s vodećim svojstvenim vrijednostima redom na dijagonali, vrijednosti svih koordinata difuzijskog preslikavanja za cijeli promatrani skup podataka možemo dobiti skaliranjem svakog stupca matrice svojstvenih vektora odgovarajućom svojstvenom vrijednosti.

Premda smo difuzijsko preslikavanje izračunali samo na podskupu svih podataka, sada ćemo to "proširiti" i na sve preostale originalne podatke, i to koriteći prethodno opisanu Laplaceovu piramidu, koju ćemo implementieati u Matlab-u izravno kako je opisano, postavljajući parametre.

Anomaly score

Određivanje stupnja anomalije određenog podatka x_i temelji se također na pristupu najbližih susjeda, ali u prostoru difuzijskih koordinata $\Psi(x_i)$. Računa se afinitet ili sličnost $\overline{w}(i,j)$ između podatka x_i i podatka x_j unutar prozora N_i oko x_i, koristeći difuzijsku udaljenost i Gaussovu jezgru s globalnim faktorom $\overline{\sigma}$:

$$\overline{w}(i,j) = e^{\frac{-||\Psi(x_i) - \Psi(x_j)||^2}{\overline{\sigma}}}. \tag{14}$$

Faktor $\overline{\sigma}$ određuje se na temelju empirijski dobivene varijance σ^2_{pair} difuzijskih udaljenosti između nasumično odabranih parova podataka: $\overline{\sigma} = r\sigma^2_{pair}$ (u glavnoj literaturi ovog seminara: $r = 20$). Stupnaj anomalije, tzv. *anomaly score*, za podatak x_i može se definirati kao:

$$C_l(i) = 1 - \frac{1}{m}\sum_{j \in N_i} \overline{w}(i,j), \tag{15}$$

gdje je N_i skup m susjeda u prozoru oko podatka x_i. Unutarnji dio prozora oko x_i se "maskira" kako bi se izbjegla usporedba podatka s neposrednim susjedima, pod pretpostavkom da je anomalija veća od jednog podatka. Veličina prozora W i maske M^{mask} ovise o očekivanoj veličini anomalije. Visoka vrijednost $C_l(i)$ (tj. blizu 1) ukazuje na malu sličnost s okolinom u difuzijskom prostoru, tj. na potencijalnu anomaliju.

Višerezolucijski pristup

Opisani algoritam primjenjuje se na jednu matricu koja predstavlja originalnu sliku. Međutim, u obradi slika poznat je takozvani višerezolucijski pristup, koji koristi takozvanu **Gaussovu piramidu** slike - familiju slika $\{G_l\}_{l=0}^{L}$, gdje je G_0 originalna slika, a svaka sljedeća slika dobivena je pogoršavanjem rezolucije prethodno dobivene slike. Slika G_L je slika najlošije rezolucije. Opisani algoritam odvija se iterativno po razinama. Kreće od razine najgrublje rezolucije - L:

1. Na razini l (počevši od L), odabire se podskup podataka Γ_l. Za razinu najgrublje rezoplucije L, Γ_L se bira nasumično (možda i svi podaci ako je G_L dovoljno mala). Za razine finije rezolucije ($l < L$), Γ_l obavezno uključuje podatke koji odgovaraju *sumnjivim* podacima s prethodne (grublje) razine $l + 1$, dok se ostatak Γ_l popunjava nasumičnim podacima.

2. Izračuna se difuzijsko preslikavanje za podskup Γ_l i proširi se na sve podatke u G_l.

3. Izračuna se stupanj anomalije $C_l(i)$ za svaki podatak x_i na razini l.

4. Podaci čija ocjena $C_l(i)$ prelazi određeni prag (eng. *treshold*)τ_l (npr. 95-ti percentil svih procjena na toj razini) označavaju se kao *sumnjivi*.

5. Ako je $l > 0$, prijeđi na sljedeću (finiju) razinu $l - 1$, koristeći informacije o sumnjivim podacima za bazu uzorkovanja u koraku 1.

6. Ako je $l = 0$ (originalna rezolucija), konačno pridruživanje stupnjeva anomalije C_0 koristi se za detekciju. Može se primijeniti konačni prag τ i prostorno filtriranje/zaglađivanje.

Ovakav proces osigurava da se informacije o potencijalnim anomalijama prenose s grubljih na finije razine, čime se povećava vjerojatnost da će anomalija biti prikladno zastupljena u uzorku na finijim razinama, čak i ako je bila slabo vidljiva ili slabo zastupljena u uzorcima na grubljim. Postavljanje nižeg praga τ_l na grubljim razinama je prihvatljivo jer konačna odluka pada tek na razini $l = 0$, čime se smanjuje rizik od pogreške na grubim razinama bez nužnog povećanja konačne stope *lažnih uzbuna*. Visok stupanj na razini l vodi do gušćeg uzorkovanja tih područja na razini $l - 1$, što rezultira boljim odvajanjem anomalije na finijoj razini. Pristup je također računski efikasniji od nekih jednorezolucijskih metoda jer se na grubljim razinama koriste manje značajke (manji isječci).

Literatura

[1] G. Mishne and I. Cohen, "Multiscale Anomaly Detection Using Diffusion Maps," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 1, pp. 111–121, Feb. 2013.

[2] [Anonymous Authors or Authors as cited in source [23]], "Detection of anomaly trends in dynamically evolving systems," in *Proc. AAAI Fall Symposium Series*, 2010.

[3] The MathWorks, Inc., *MATLAB Documentation*. Natick, Massachusetts, 2025. Available at: https://www.mathworks.com/help/matlab/

[4] G. Mishne and I. Cohen, "Code for Multiscale Anomaly Detection Using Diffusion Maps," 2018. Available at: https://israelcohen.com/wp-content/uploads/2018/05/multiscale_dm_ano_detect.zip

[5] R. R. Coifman and S. Lafon, "Diffusion maps," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 5–30, 2006.

[6] B. Nadler, S. Lafon, R. R. Coifman, and I. G. Kevrekidis, "Diffusion maps, spectral clustering and reaction coordinates of dynamical systems," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 113–127, 2006.

[7] Z. Drmač, *Difuzijska preslikavanja* (Prezentacija s predavanja). 2025.