UNIVERZITET CRNE GORE

ELEKTROTEHNIČKI FAKULTET

SANJA BULATOVIĆ

DETEKCIJA ANOMALIJA U RADU INDUSTRIJSKIH MAŠINA PRIMJENOM AUTOENKODERA

- MAGISTARSKI RAD -

Podgorica, 2025. godine

PODACI I INFORMACIJE O MAGISTRANDU

Ime i prezime: **Sanja Bulatović**

Datum i mjesto rođenja: **22. april 1994. godine, Podgorica**

Prethodno završene studije:

Elektrotehnički fakultet, Elektronika, telekomunikacije i računari, 180 ECTS kredita, 2016.

Elektrotehnički fakultet, Računari, 60 ECTS kredita, 2017.

INFORMACIJE O MASTER RADU

Naziv master studija: **Magistarske studije, studijski program Elektronika, telekomunikacije i računari**

Naslov rada: **Detekcija anomalija u radu industrijskih mašina primjenom autoenkodera**

Fakultet na kojem je rad odbranjen: **Elektrotehnički fakultet Podgorica**

UDK, OCJENA I ODBRANA MASTER RADA

Datum prijave master rada: **16.05.2024.**

Datum sjednice Vijeća na kojoj je prihvaćena tema: **31.05.2024.**

Komisija za ocjenu teme i podobnosti magistranta:

1. **Prof. dr Vesna Popović - Bugarin, predsjednica**
2. **Prof. dr Slobodan Đukanović, mentor**
3. **Doc. dr Miloš Brajović, član**

Mentor: **Prof. dr Slobodan Đukanović**

Komisija za ocjenu rada:

4. **Prof. dr Vesna Popović - Bugarin, predsjednica**
5. **Prof. dr Slobodan Đukanović, mentor**
6. **Doc. dr Miloš Brajović, član**

Komisija za odbranu rada:

1. **Prof. dr Vesna Popović - Bugarin, predsjednica**
2. **Prof. dr Slobodan Đukanović, mentor**
3. **Doc. dr Miloš Brajović, član**

Datum odbrane: **24.04.2025**

Datum promocije:

Etička izjava

U skladu sa članom 22 Zakona o akademskom integritetu i članom 18 Pravila studiranja na postdiplomskim studijama, pod krivičnom i materijalnom odgovornošću, izjavljujem da je magistarski rad pod naslovom:

„ Detekcija anomalija u radu industrijskih mašina primjenom autoenkodera"

moje originalno djelo.

Podnosilac izjave:

Sanja Bulatović

U Podgorici, dana 21.02.2025. godine

Sadržaj

Sažetak 1
Abstract 2
1 Uvod 3
2 Mašinsko učenje 5
2.1 Uvod 5
2.2 Tipovi mašinskog učenja 6
2.3 Procesi i tehnike u mašinskom učenju 10
2.4 Treniranje, validacija i testiranje modela 12
2.5 Metrike performansi 15
2.6 Preprilagođavanje i potprilagođavanje 20
3 Neuronske mreže i duboko učenje 22
3.1 Uvod 22
3.2 Osnovni elementi neuronskih mreža 23
3.3 Arhitektura i tipovi neuronskih mreža 23
3.3.1 Osnovna arhitektura neuronskih mreža 24
3.3.2 Perceptron i napredne arhitekture neuronskih mreža 24
3.4 Propagacija unaprijed 27
3.5 Aktivacione funkcije 29
3.6 Obučavanje neuronskih mreža algoritmom propagacije unazad i algoritmi optimizacije 31
3.6.1 Metod gradijentnog spusta 31
3.6.2 Adam optimizator 33
3.7 Duboko učenje 34
3.7.1 Problemi nestajućih i eksplodirajućih gradijenata 35
3.7.2 Tehnike za poboljšanje performansi modela 37
3.7.3 Autoenkoderi 42
4 Metodologija i implementacija modela 48
4.1 Uvod 48
4.2 MIMII skup podataka 49
4.3 Mel spektrogram i ekstrakcija karakteristika 50
4.4 Arhitektura autoenkodera 53
4.5 Proces treniranja i evaluacija modela 54
5 Eksperimentalni rezultati 56
6 Zaključak 63

Sažetak

Precizna detekcija anomalija u industrijskim mašinama od ključnog je značaja za očuvanje efikasnosti, pouzdanosti i sigurnosti sistema u savremenim industrijskim postrojenjima. Ovaj rad istražuje primjenu autoenkodera za detekciju anomalija, koristeći MIMII skup podataka. Fokus je na unaprjeđenju performansi modela iz originalnog rada [1], u kom se inicijalno obrađuje ovaj skup, kroz modifikacije arhitekture i optimizaciju parametara treniranja.

Testirane su četiri vrste mašina: ventili, pumpe, ventilatori i klizne šine, a performanse su evaluirane pomoću AUC metrike na različitim nivoima odnosa energije signala i šuma (6 dB, 0 dB i -6 dB). Rezultati pokazuju značajna poboljšanja performansi, u odnosu na originalan rad, posebno za klizne šine (do 70% za model 06 pri SNR = 0 dB). Štaviše, naprednije arhitekture modela pokazale su otpornost i na ekstremno nepovoljne uslove pri SNR = -6 dB, gdje su postignuta značajna poboljšanja performansi u detekciji anomalija, što ukazuje na njihovu primjenjivost u realnim industrijskim scenarijima.

Pored analize rezultata i demonstracije unaprjeđenja performansi modela, u radu su identifikovani izazovi u detekciji anomalija i predložene smjernice za dalja istraživanja, uključujući testiranje modela na drugim skupovima podataka, integraciju različitih izvora podataka, poput vibracionih i temperaturnih podataka, kao i optimizaciju modela za implementaciju u realnom vremenu, čime se otvara prostor za dalja istraživanja i razvoj u ovoj oblasti.

Ključne riječi: mašinsko učenje, duboko učenje, detekcija anomalija, autoenkoderi, industrijski sistemi

Abstract

Accurate anomaly detection in industrial machines is crucial for maintaining efficiency, reliability, and safety in modern industrial facilities. This paper explores the application of autoencoders for anomaly detection using the MIMII dataset. The focus is on improving the performance of models from the original study [1], where this dataset was initially introduced, through architectural modifications and optimization of training parameters.

Four types of machines were tested: valves, pumps, fans, and slide rails, with performance evaluated using the AUC metric across various signal-to-noise ratio (SNR) levels (6 dB, 0 dB, and -6 dB). The results demonstrate significant performance improvements compared to the original study, particularly for slide rails (up to 70% for model 06 at SNR = 0 dB). Moreover, the more advanced model architectures exhibit remarkable resilience under extremely adverse conditions at an SNR of –6 dB, achieving significant improvements in anomaly detection performance. This resilience indicates their applicability in real-world industrial scenarios.

In addition to analyzing the results and demonstrating performance enhancements, this study identifies challenges in anomaly detection and proposes guidelines for addressing these challenges, including testing models on other datasets, integrating diverse data sources such as vibration and temperature data, and optimizing models for real-time implementation, thus creating opportunities for further research and development in this field.

Keywords: machine learning, deep learning, anomaly detection, autoencoders, industrial systems

1 Uvod

Industrijska postrojenja se, u velikoj mjeri, oslanjaju na pouzdanost i efikasnost mašina kako bi obezbijedila neprekidnu proizvodnju, smanjila troškove održavanja i izbjegla neočekivane zastoje. Tradicionalni pristupi detekciji anomalija, poput fizičkih inspekcija i senzorskih sistema, iako korisni, imaju svoja ograničenja. Fizičke inspekcije su često skupe, zavise od ljudskog faktora i detektuju kvarove tek u naprednijim fazama, dok senzorski sistemi zahtijevaju kompleksnu instalaciju i mogu biti ograničeni u prepoznavanju određenih vrsta anomalija.

Ovi izazovi ukazuju na potrebu za primjenom naprednijih i efikasnijih metoda koje omogućavaju rano prepoznavanje problema i proaktivno održavanje. Primjena metoda dubokog učenja značajno je unaprijedila sposobnost detekcije anomalija u industrijskim sistemima. Jedan od najsavremenijih pristupa u ovoj oblasti je upotreba autoenkodera, koji omogućavaju analizu složenih obrazaca u podacima i identifikaciju odstupanja od normalnih radnih uslova. Ovaj model ima potencijal za rano otkrivanje problema, smanjujući rizik od neočekivanih kvarova i omogućavajući efikasnije preventivno održavanje.

U ovom radu, istraživanje je usmjereno na analizu performansi autoenkodera koristeći MIMII skup podataka. Ovaj skup podataka sadrži informacije o normalnim i abnormalnim uslovima rada industrijskih mašina, što omogućava detaljnu evaluaciju performansi modela u realnim industrijskim scenarijima i čini ga izuzetno pogodnim za proučavanje metoda detekcije anomalija. Fokus istraživanja je na četiri vrste industrijskih mašina – ventilima, pumpama, ventilatorima i kliznim šinama – s ciljem analize uticaja različitih modifikacija arhitekture modela i optimizacije parametara treniranja na performanse detekcije anomalija.

Identifikovani su ključni izazovi i predložene su smjernice za njihovo prevazilaženje, čime se otvara prostor za dalja istraživanja i praktičnu primjenu ovih metoda u realnim industrijskim okruženjima.

Rad je organizovan u šest poglavlja. Prvo poglavlje predstavlja uvod, u okviru kog je opisan predmet istraživanja, cilj rada i značaj detekcije anomalija u industrijskim mašinama.

Drugo poglavlje rada pruža pregled mašinskog učenja, postavljajući temelj za razumijevanje ključnih koncepata koji su osnova za implementaciju autoenkodera u detekciji anomalija. Ovo poglavlje započinje uvodom u mašinsko učenje i njegove osnovne principe, nakon čega je predstavljena klasifikacija različitih tipova mašinskog učenja. Dalje su opisani procesi i tehnike koje se koriste u mašinskom učenju, pri čemu je poseban naglasak stavljen na treniranje, validaciju i testiranje modela, čije su dobro razumijevanje i implementacija ključni za postizanje preciznosti i sposobnosti generalizacije samih modela. Diskutuju se i metrike performansi koje se koriste za procjenu uspješnosti modela, dok se završni dio poglavlja bavi problemima preprilagođavanja (*overfitting*) i potprilagođavanja (*underfitting*).

Treće poglavlje rada pruža teorijsku osnovu za razumijevanje neuronskih mreža i dubokog učenja, ključnih tehnika koje stoje iza primjene autoenkodera za detekciju anomalija. U okviru ovog poglavlja razmatraju se osnovni elementi neuronskih mreža, njihova arhitektura i različiti tipovi, kao i funkcije aktivacije, algoritam i optimizatori koji se koriste za obučavanje mreža. Dalje, diskutuju se specifični izazovi dubokog učenja, poput problema nestajućih i eksplodirajućih gradijenata, kao i tehnike za unaprjeđenje performansi modela. Na kraju, detaljno su predstavljeni autoenkoderi kao ključna komponenta ovog istraživanja.

Četvrto poglavlje predstavlja metodologiju istraživanja, sa fokusom na karakteristike MIMII skupa podataka, proces ekstrakcije karakteristika, arhitekturu autoenkodera i parametre treniranja.

U petom poglavlju prikazani su eksperimentalni rezultati, uključujući analizu uticaja različitih arhitektura i parametara na performanse modela.

Konačno, šesto poglavlje predstavlja zaključak, ističući glavne doprinose rada, identifikovane izazove i preporuke za buduća istraživanja.

2 Mašinsko učenje

2.1 Uvod

Mašinsko učenje predstavlja oblast računarskih nauka i poddomen vještačke inteligencije (slika 1) koji se fokusira na razvoj algoritama i statističkih modela koji omogućavaju računarskim sistemima da automatski uče i poboljšavaju svoje performanse na osnovu iskustva, tj. podataka. Ova disciplina istražuje kako računari mogu imati sposobnost učenja bez eksplicitnog programiranja za svaki zadatak.

Ključna karakteristika mašinskog učenja je njegova sposobnost da identifikuje obrasce i donosi odluke sa minimalnom ljudskom intervencijom. Temelji se na tome da računari mogu obrađivati i analizirati velike količine podataka, otkrivati skrivene uzorke i zakonitosti u tim podacima, te na osnovu toga vršiti predviđanja, odnosno izvoditi klasifikacije, prepoznavati šablone, itd.

Slika 1: *Odnos između vještačke inteligencije, mašinskog učenja i dubokog učenja*

Počeci mašinskog učenja sežu do 1950-ih godina, kada je Alan Turing, u svom radu [2], uveo koncept Turing-ovog testa. Ovaj test predstavljao je prvi jasno definisan i strukturiran pristup za

procjenu sposobnosti mašine da imitira ljudsku inteligenciju kroz razmjenu informacija u obliku pisanih odgovora. Turing-ov rad je postavio teorijske osnove za razvoj vještačke inteligencije, otvarajući ključna pitanja o mogućnostima mašina da uče i adaptiraju se. Ove ideje kasnije su postale suština razvoja algoritama mašinskog učenja.

Nekoliko godina kasnije, Arthur Samuel razvio je jedan od prvih programa za mašinsko učenje koristeći igru dama [3]. Njegov program, koji je samostalno učio iz prethodnih partija, postavio je temelje za razvoj algoritama mašinskog učenja. Tokom 1960-ih i 1970-ih, razvoj u ovoj oblasti bio je ograničen zbog tehničkih ograničenja, poput nedovoljne računarske snage i ograničenih skupova podataka. Međutim, s pojavom interneta i napretkom računarske infrastrukture u 1980-im i 1990-im, oblast mašinskog učenja doživjela je značajan procvat. Eksponencijalni rast u dostupnosti podataka, poznat kao „*Big Data*", zajedno sa povećanjem računarske snage, omogućio je modelima mašinskog učenja da uče iz obimnijih i raznovrsnijih skupova podataka. Razvoj naprednih algoritama, posebno u oblastima dubokog učenja, dodatno je ubrzao razvoj mašinskog učenja. Period od 2010. godine nadalje, obilježen je izuzetnim rastom, što je mašinsko učenje postavilo kao jednu od temeljnih tehnologija savremenog doba, sa širokom primjenom u raznovrsnim industrijskim i naučnim sferama.

Danas se mašinsko učenje široko koristi u raznim aplikacijama, od prepoznavanja govora, preko sistema za preporučivanje u *online* trgovini, do autonomnih vozila i dijagnostike u medicini. Razvojem i napretkom u ovoj oblasti, računari postaju sve sposobniji za obavljanje kompleksnih zadataka, što otvara nove mogućnosti u skoro svim segmentima industrije i nauke.

2.2 Tipovi mašinskog učenja

Mašinsko učenje može se klasifikovati u nekoliko osnovnih tipova [4][5]:

- Nadgledano učenje (*Supervised Learning*)
- Nenadgledano učenje (*Unsupervised Learning*)
- Polunadgledano učenje (*Semi-Supervised Learning*)
- Učenje sa podrškom (*Reinforcement Learning*)

Nadgledano učenje, kao jedan od ključnih pravaca u oblasti mašinskog učenja, temelji se na korišćenju označenih skupova podataka za treniranje algoritama. U ovom kontekstu, označeni podaci podrazumijevaju to da svaki primjer u skupu podataka sadrži ulazne varijable - *features* (npr. slike, tekst, numeričke vrijednosti) i pripadajuće izlazne varijable. Primarni cilj

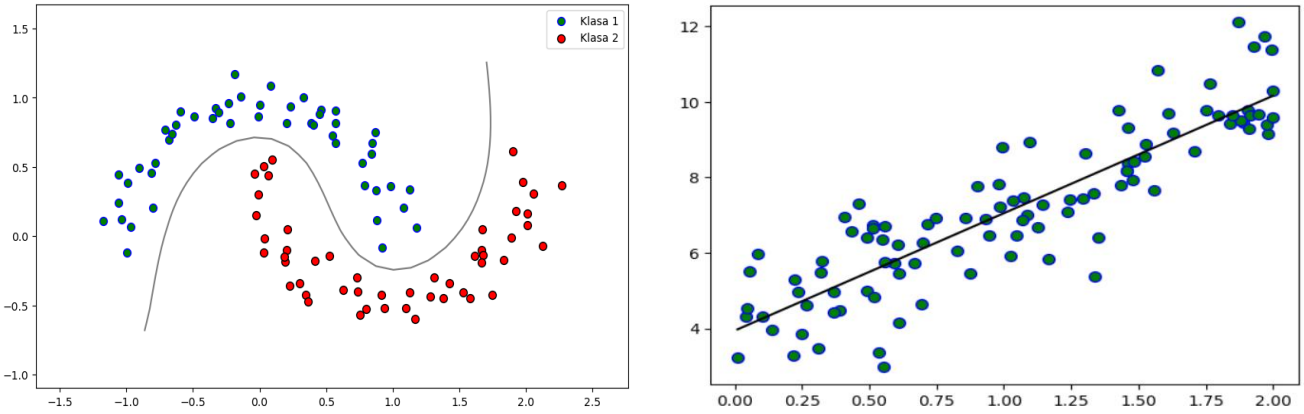
nadgledanog učenja je razvijanje modela koji može naučiti odnos između ulaznih i izlaznih podataka, kako bi se omogućilo precizno predviđanje ili klasifikacija za nove, neviđene ulazne podatke.

U praksi, proces nadgledanog učenja započinje podjelom skupa podataka na trening i testni skup. Model se trenira na trening skupu, gdje se algoritam uči da povezuje ulazne podatke sa odgovarajućim izlazima. Nakon treniranja, model se testira na testnom skupu, koji sadrži nove, neviđene podatke, kako bi se ocijenila njegova sposobnost da tačno predviđa ili klasifikuje ove nove podatke. Ocjena modela se vrši pomoću metrika kao što su srednja kvadratna greška, srednja apsolutna greška, tačnost, preciznost, odziv, F1 ocjena, AUC, itd.

Dvije glavne kategorije problema za koje se primjenjuje nadgledano učenje su klasifikacija i regresija (slika 2) [6]. Klasifikacija se odnosi na probleme kod kojih je cilj predvidjeti diskretnu klasu ili oznaku (npr. određivanje da li je *e-mail spam* ili nije), dok se regresija bavi predviđanjem kontinualne vrijednosti (npr. predviđanje cijene kuće na osnovu njenih karakteristika).

Nadgledano učenje je primjenljivo u mnogim realnim situacijama. Na primjer, u medicinskoj dijagnostici, nadgledano učenje se koristi za klasifikaciju medicinskih slika, kao što su rendgenski snimci ili MRI, u cilju prepoznavanja različitih bolesti ili abnormalnosti, kao što su tumori ili lezije. U prepoznavanju govora, modeli nadgledanog učenja koriste označene podatke govora da nauče kako da prepoznaju i transkribuju ljudski govor, što se koristi u digitalnim asistentima poput *Siri* ili *Google Assistant*-a. Takođe, u sistemima za preporučivanje, kao što je *Netflix* ili *Amazon*, modeli nadgledanog učenja koriste istoriju korisničkih interakcija (npr. ocjene filmova ili kupovine proizvoda) da preporuče nove proizvode ili sadržaje.

Efikasnost modela nadgledanog učenja direktno zavisi od kvaliteta i količine dostupnih trening podataka, kao i od izbora i podešavanja algoritma koji se koristi.



Slika 2: *Lijevi grafik prikazuje klasifikacioni model koji razlikuje dvije klase, dok desni grafik predstavlja regresijski model koji predviđa kontinualnu vrijednost.*

Nenadgledano učenje predstavlja jednu od ključnih oblasti mašinskog učenja koja se bavi analizom i obradom podataka koji nijesu unaprijed označeni ili klasifikovani. Za razliku od nadgledanog učenja, koje se oslanja na označene podatke za treniranje modela, nenadgledano učenje istražuje sirove podatke pokušavajući da otkrije uočljive obrasce, klastere ili zajedničke karakteristike među podacima.

Nenadgledano učenje ima široku primjenu u domenima poput analize podataka (*Data Mining*), obrade prirodnog jezika i računarskog vida. Na primjer, u analizi podataka koristi se za otkrivanje nepoznatih obrazaca u velikim skupovima podataka, kao što su segmentacija tržišta ili analiza ponašanja korisnika na *web* sajtovima. U oblasti obrade prirodnog jezika, koristi se za grupisanje sličnih dokumenata ili identifikovanje tematskih klastera u velikim korpusima tekstova. U računarskom vidu, nenadgledano učenje se koristi za segmentaciju slika, gdje model pokušava da identifikuje i grupiše piksele ili objekte unutar slike bez prethodnih oznaka.

Nenadgledano učenje omogućava dublji uvid u strukturu i obrasce kompleksnih skupova podataka, te nove pristupe njihovoj analizi. Iako može biti manje precizno u identifikovanju specifičnih obrazaca u poređenju sa nadgledanim učenjem, nenadgledano učenje ima značajnu sposobnost generalizacije. Ovo ga čini neprocjenjivim u situacijama gdje je označavanje podataka neizvodljivo ili skupo.

Jedna od njegovih glavnih prednosti jeste dostupnost veće količine neoznačenih podataka, što omogućava analizu ogromnih skupova podataka bez potrebe za ručnim označavanjem.

Polunadgledano učenje predstavlja hibridni pristup mašinskom učenju koji kombinuje elemente nadgledanog i nenadgledanog učenja. Ono koristi veliku količinu neoznačenih podataka u kombinaciji sa manjom količinom označenih podataka za treniranje modela. Za razliku od nadgledanog učenja, koje se oslanja isključivo na označene podatke, i nenadgledanog učenja, koje koristi samo neoznačene podatke, polunadgledano učenje balansira između ova dva pristupa. Na ovaj način, spaja preciznost nadgledanog učenja sa fleksibilnošću i raznovrsnošću nenadgledanog učenja, omogućavajući efikasno iskorišćavanje dostupnih resursa. Na ovaj način model može bolje identifikovati i analizirati specifične obrasce, dok istovremeno koristi prednosti većih i raznovrsnijih skupova podataka.

U kontekstu polunadgledanog učenja, označeni podaci se koriste za usmjeravanje procesa učenja, dajući modelu osnovne informacije o tome kako klasifikovati ili analizirati podatke, npr. informacije o broju klasa i nekim njihovim karakteristikama. Neoznačeni podaci, koji su obično mnogo obimniji i lakše dostupni, koriste se za dodatno unaprjeđenje i podešavanje modela. Ova kombinacija omogućava modelima da bolje generalizuju i efikasnije se prilagode novim podacima.

Polunadgledano učenje je posebno korisno u situacijama gdje je teško, skupo ili vremenski zahtjevno dobiti velike količine označenih podataka, što je čest slučaj u realnim aplikacijama. Njegove primjene uključuju obradu prirodnog jezika, računarski vid i biomedicinska istraživanja, gdje kombinovanje označenih i neoznačenih podataka može značajno poboljšati performanse i tačnost modela.

Učenje sa podrškom je pristup u mašinskom učenju koji se fokusira na razvoj algoritama koji omogućavaju agentu (obično softverskom) da nauči kako da se ponaša u određenom okruženju, tako što maksimizuje neku vrstu nagrade. Agent donosi odluke ili izvodi akcije unutar svog okruženja i na osnovu povratnih informacija, u obliku nagrada ili kazni, uči optimalnu strategiju ponašanja. Ovaj proces je sličan načinu na koji ljudi i životinje uče iz iskustva, postepeno poboljšavajući svoje ponašanje kroz sistem pokušaja i grešaka.

Učenje sa podrškom je posebno efikasno u rješavanju problema koji uključuju sekvencijalno donošenje odluka, gdje se posljedice svake odluke manifestuju kroz kontinualne serije interakcija. Primjene učenja sa podrškom obuhvataju razvoj naprednih sistema za igranje igara (kao što su šah i *Go*), tehnike za optimizaciju strategija vožnje u realnom vremenu kod autonomnih vozila, kao i optimizaciju logističkih operacija, uključujući raspodjelu resursa, planiranje ruta i upravljanje skladištima.

Jedan od izazova u učenju sa podrškom je potreba za velikom količinom interakcija s okruženjem, što može biti vremenski i računarski zahtjevno. Takođe, izbor i dizajn odgovarajućih sistema nagrađivanja su ključni za uspjeh ovih modela.

2.3 Procesi i tehnike u mašinskom učenju

Prvi korak u formiranju nekog modela mašinskog učenja predstavlja temeljni proces prikupljanja i obrade podataka. Efikasnost modela mašinskog učenja u velikoj mjeri zavisi od kvaliteta i relevantnosti podataka na kojima se obučava.

Prikupljanje podataka obuhvata razne izvore, uključujući senzore, digitalne medije, finansijske zapise, društvene mreže, javno dostupne baze podataka i mnoge druge. Važno je osigurati da su podaci što raznovrsniji i da odražavaju realne scenarije u kojima će model biti primijenjen.

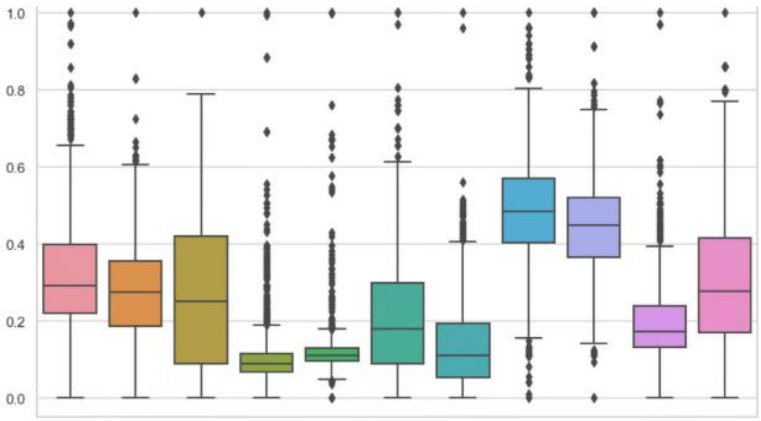
Nakon prikupljanja, podaci se obrađuju da bi bili prikladni za mašinsko učenje. Ovo uključuje čišćenje podataka od nepravilnosti i nedostataka, što obuhvata ispravljanje grešaka, obradu nedostajućih vrijednosti i uklanjanje duplikata radi osiguranja tačnosti i potpunosti podataka. Zatim slijedi transformacija podataka, gdje se izdvajaju dva ključna procesa – normalizacija i standardizacija (slike 3 – 5).

Normalizacija podataka podrazumijeva prilagođavanje vrijednosti atributa na određeni opseg, obično [0,1]. Na taj način održavaju se relativne razlike među vrijednostima, iako se apsolutni raspon svodi na željeni interval. Za datu vrijednost x, formula izgleda ovako:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

gdje su x_{min} i x_{max} minimalna i maksimalna vrijednost tog atributa u skupu podataka.

Ova metoda je posebno korisna u algoritmima mašinskog učenja koji se oslanjaju na metriku udaljenosti (npr. k najbližih susjeda), jer osigurava da svaki atribut ima podjednak uticaj prilikom računanja rastojanja. Međutim, treba voditi računa da ekstremne vrijednosti značajno utiču na normalizaciju, što ponekad, u slučaju *outliers* može iskriviti rezultate normalizacije.



Slika 3: *Boxplot za normalizovane podatke [7]*

Standardizacija podataka, s druge strane, transformiše podatke tako da imaju srednju vrijednost 0 i standardnu devijaciju 1. Formula za standardizaciju vrijednosti x je:

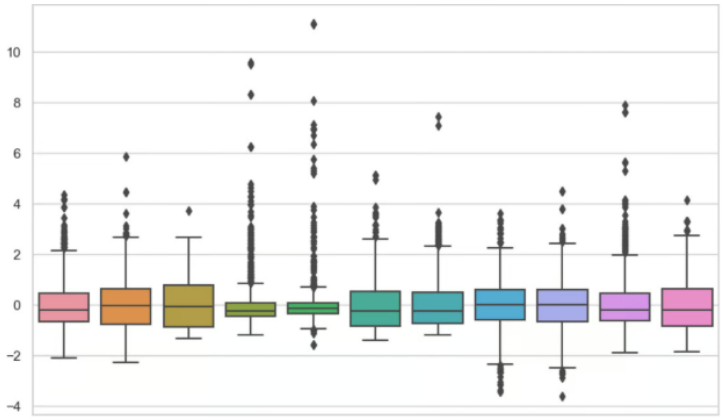
$$x_{sd} = \frac{x - \mu}{\sigma}$$

gdje je μ srednja (prosječna) vrijednost atributa u skupu podataka, a σ standardna devijacija istog atributa. Standardna devijacija predstavlja mjeru rasipanja vrijednosti atributa u skupu podataka u odnosu na srednju vrijednost tog atributa i data je izrazom:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}}$$

gdje su x_i vrijednosti atributa, μ srednja vrijednost, a N je ukupan broj vrijednosti atributa.

Veća standardna devijacija ukazuje na veće rasipanje podataka, dok manja standardna devijacija znači da su vrijednosti bliže srednjoj vrijednosti.



Slika 4: *Boxplot za standardizovane podatke [7]*

Slika 5: *Prikaz primjera originalnih podataka (lijevo), normalizovanih podataka (u sredini) i standardizovanih podataka (desno)*

Ključni proces u mašinskom učenju, pored normalizacije i standardizacije, jeste inženjering karakteristika (*Feature Engineering*). On uključuje tehnike poput kreiranja novih karakteristika radi boljeg predstavljanja složenosti podataka, selekcije karakteristika radi izbjegavanja preprilagođavanja, kao i kodiranja kategoričkih varijabli kako bi se olakšala njihova obrada.

Nakon obrade i inženjeringa karakteristika, podaci su spremni za naredne faze u mašinskom učenju, uključujući trening, validaciju i testiranje modela. Ove faze su ključne za procjenu performansi modela, gdje se posebna pažnja posvećuje izbjegavanju preprilagođavanja i osiguravanju da model adekvatno generalizuje na novim, neviđenim podacima. Procesi kao što su evaluacija i optimizacija modela, uz primjenu različitih metrika, omogućavaju detaljnu analizu i fino podešavanje modela, dok se istovremeno rješavaju izazovi poput preprilagođavanja i potprilagođavanja.

2.4 Treniranje, validacija i testiranje modela

Treniranje i testiranje modela ključni su procesi u mašinskom učenju za razvoj efikasnih i pouzdanih modela. Trening skup, koji obuhvata većinu dostupnih podataka, koristi se za obuku modela. Tokom ove faze, model uči da prepoznaje obrasce i odnose unutar podataka, prilagođavajući svoje parametre, s ciljem minimizacije funkcije gubitka, odnosno greške predviđanja. Model se iterativno usavršava na trening skupu kroz proces optimizacije, pri čemu je važno osigurati da model ne postane previše prilagođen trening skupu, što bi ograničilo njegovu sposobnost da efikasno funkcioniše na novim, neviđenim podacima.

Sa druge strane, test skup služi konačnoj evaluaciji performansi modela na neviđenim podacima. Ovaj skup, u koji model do kraja procesa treniranja nema uvid, omogućava objektivnu procjenu sposobnosti modela da generalizuje, pružajući stvaran uvid u njegovu efikasnost u primjeni na novim, nepoznatim podacima.

Validacioni skup, iako nije uvijek obavezan, igra važnu ulogu u finom podešavanju hiperparametara modela. Njegova primjena pomaže u evaluaciji modela tokom treninga, kako bi se postigle optimalne performanse prije krajnje evaluacije na test skupu. Ovaj pristup se često koristi u scenarijima kad je na raspolaganju velika količina podataka i kad se može priuštiti dijeljenje skupa podataka na tri podskupa – za trening, validaciju i test.

Podjela podataka na trening, validacione i test skupove (slika 6) obično prati određeni odnos veličina tih skupova, koji se može prilagođavati u zavisnosti od ukupne veličine skupa podataka i specifičnih karakteristika problema. Tradicionalno, preporučljiv odnos bio bi 70% podataka za trening, 15% za validaciju i 15% za testiranje. Međutim, u doba velikih podataka, kada skupovi podataka mogu obuhvatati milione primjera, često se veći dio (npr. 95%) zadržava za trening, dok se manji dio (po 2,5%) koristi za validaciju i testiranje [4].

Slika 6: *Šematski prikaz pristupa podjele skupa podataka na dio za treniranje, validaciju i testiranje.*

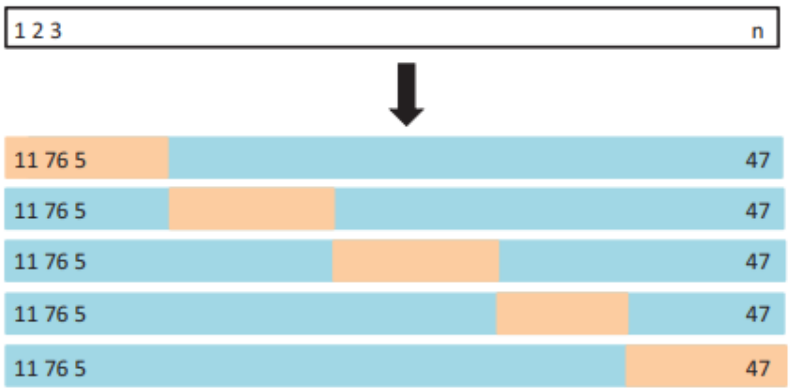
Metoda *k-fold* unakrsne validacije popularan je izbor za procjenu performansi modela, naročito u slučajevima kada obim podataka nije toliki da se može priuštiti formiranje dovoljno velikog skupa za testiranje, koji bi obezbijedio dobru procjenu modela. Riječ je o specifičnoj metodi koja omogućava izbor optimalnih hiperparametara modela, pri čemu se smanjuje rizik od pristrasnosti koja može proizaći iz specifičnosti odabranih podataka za validaciju. Ukoliko je prisutna količina podataka mala, nije potrebno izdvajati poseban dio za testiranje, već se dostupni podaci više puta koriste za proces treniranja i validacije, pri čemu se svaki put za validaciju koriste drugi podaci, omogućavajući objektivnu procjenu sposobnosti modela na novim podacima.

Konkretno, postupak *k-fold* unakrsne validacije započinje podjelom trening skupa podataka na *k* jednakih podskupova (*fold*-ova), obično pet ili deset [8], [9]. U petoslojnoj unakrsnoj validaciji, na primjer, trening podaci se nasumično raspodjeljuju u pet podskupova. Svaki od ovih podskupova jednom služi kao validacioni skup, dok preostali podskupovi čine trening skup. Na

ovaj način, model se iznova trenira i validira pet puta, pri čemu se u svakoj iteraciji koriste različiti podskupovi za trening i validaciju (slika 7).

Performanse modela se, nakon svake faze treniranja, procjenjuju na osnovu izabrane metrike koristeći odabrani *fold* za validaciju, čime se osigurava nezavisnost procjena za svaki *fold* i omogućava bolja procjena sposobnosti generalizacije modela. Konačna ocjena performansi modela izračunava se kao prosječna vrijednost metrike, dobijena zbrajanjem vrijednosti metrike za sve *fold*-ove i dijeljenjem s brojem *fold*-ova.

Nakon što je odabran model sa optimalnim hiperparametrima, trenira se novi, finalni, model, koristeći cijeli trening skup, tj. sve *fold*-ove. Konačna evaluacija vrši se na odvojenom test skupu, ukoliko on postoji, a u suprotnom model se direktno koristi u produkciji, jer je očekivanje da će se pokazati dobro na novim podacima, usljed zadovoljavajućih performansi procijenjenih *k-fold* unakrsnom validacijom.



Slika 7: *Šematski prikaz petoslojne unakrsne validacije. Skup opservacija n podataka se nasumično dijeli na pet nepreklapajućih grupa. Svaka od ovih petina predstavlja validacioni skup (prikazano u bež boji), a ostatak predstavlja trening skup (prikazano u plavoj boji)]. [9]*

Iako je podjela podataka neophodan korak u mašinskom učenju, ona nosi određene izazove. Jedan od ključnih problema je *data leakage*, gdje informacije iz test skupa mogu nenamjerno uticati na trening skup, što rezultuje nerealno visokim performansama modela na test skupu. Na primjer, ako model predviđa cijene akcija na osnovu istorijskih podataka, ali tokom treniranja dobije informacije o budućim kretanjima cijena (koje bi u stvarnom scenariju bile dostupne tek nakon treniranja), ovo predstavlja *data leakage*.

Još jedan od problema nastaje kada odvajanje podataka za testiranje i validaciju dovede do premale količine podataka dostupne za trening, što je izraženo u situacijama sa ograničenim podacima.

Osim toga, nepravilna randomizacija podataka može dovesti do pristrasnosti modela, a neadekvatna reprezentativnost skupova može prouzrokovati neuravnoteženost klasa, u slučaju problema klasifikacije. Unakrsna validacija, iako predstavlja efikasno rješenje za ove izazove, može povećati računarsku zahtjevnost i složenost, posebno u metodama kao što je *k-fold* validacija, gdje se model više puta iznova trenira na različitim podskupovima podataka. Ovi izazovi zahtijevaju pažljivo upravljanje procesom podjele podataka, kako bi se osigurala efikasnost i pouzdanost modela mašinskog učenja.

2.5 Metrike performansi

Metrike performansi u mašinskom učenju ključni su kvantitativni indikatori koji se koriste za procjenu i interpretaciju efikasnosti modela. Različite metrike pružaju uvid u specifične aspekte performansi modela, poput tačnosti, preciznosti, robusnosti, itd., omogućavajući preciznu evaluaciju i usmjeravanje istraživanja. Ove metrike su neophodne kako bi se osigurala objektivna procjena modela, bilo da je riječ o klasifikaciji, regresiji ili nekoj drugoj vrsti zadatka [10]. U literaturi se često nazivaju i funkcijama gubitka.

U kontekstu regresije, procjena modela je prilično jednostavna. Dobro prilagođen regresijski model rezultuje predviđenim vrijednostima bliskim posmatranim vrijednostima podataka. Srednja apsolutna greška (*MAE - Mean Absolute Error*) i srednja kvadratna greška (*MSE - Mean Squared Error*) su dvije fundamentalne metrike koje se koriste za procjenu tačnosti regresijskih modela u mašinskom učenju. Obije metrike kvantifikuju grešku između stvarnih i predviđenih vrijednosti, ali na različite načine.

Srednja apsolutna greška se izračunava kao prosječna vrijednost apsolutnih vrijednosti razlika između stvarnih i predviđenih vrijednosti, za sve primjere u skupu podataka:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

gdje N predstavlja ukupan broj primjera u skupu podataka na kojem se procjenjuje greška, y_i predstavlja stvarnu vrijednost za *i*-ti primjer u skupu podataka, koju model treba da predvidi, a $\hat{y}_i$ predstavlja predviđenu vrijednost koju model generiše za *i*-ti primjer.

Ova metrika je robusna i otporna na primjere koji značajno odstupaju od predviđenih vrijednosti (*outliers*), dajući jednaku težinu svim greškama, čime pruža jasan uvid u prosječnu grešku koju model pravi u svojim predviđanjima.

S druge strane, srednja kvadratna greška mjeri prosječnu vrijednost kvadrata razlika između stvarnih i predviđenih vrijednosti:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

gdje za N, y_i, $\hat{y}_i$ važi isto kao kod MAE.

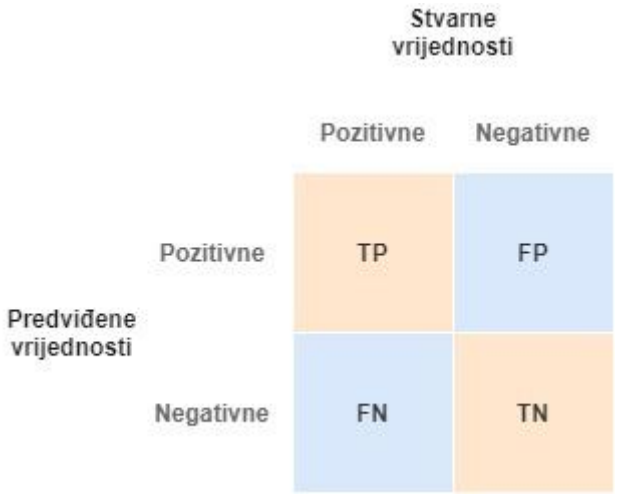
Budući da se greške kvadriraju prije nego što se uprosječe, *MSE* daje veću težinu većim greškama, čineći je osetljivom na *outliers* u podacima. Ova karakteristika čini *MSE* posebno korisnom u situacijama gdje su velike greške posebno nepoželjne i gdje ih je, usljed toga, važno strogo kazniti. Na primjer, u medicinskoj dijagnostici, kada model predviđa vjerovatnoću bolesti, velika greška u predviđanju može imati ozbiljne posljedice, pa je stoga korisno koristiti *MSE* kako bi se kaznile veće greške (npr. rezultati koji mogu dovesti do neprepoznavanja bolesti).

Oba pokazatelja, i *MAE* i *MSE*, pružaju korisne uvide u kvalitet i tačnost regresijskih modela, i izbor između njih treba da se bazira na specifičnim ciljevima i karakteristikama problema koji se rješava.

U procjeni performansi modela klasifikacije, matrica konfuzije pruža temelj za razumijevanje kako model pravi razliku između klasa. Kod binarne klasifikacije, gdje su klase pozitivna i negativna, matrica ima dimenzije 2x2 i klasifikuje predikcije modela u četiri osnovne kategorije (slika 8):

- TP (*True Positives*) predstavlja broj tačno pozitivnih klasifikacija (model tačno predviđa pozitivnu klasu),
- TN (*True Negatives*) predstavlja broj tačno negativnih klasifikacija (model tačno predviđa negativnu klasu),
- FP (*False Positives*) predstavlja broj pogrešno pozitivnih klasifikacija (model pogrešno predviđa negativnu klasu kao pozitivnu),
- FN (*False Negatives*) predstavlja broj pogrešno negativnih klasifikacija (model pogrešno predviđa pozitivnu klasu kao negativnu).

Korišćenjem matrice konfuzije mogu se izračunati ključne metrike performansi kao što su tačnost, preciznost, odziv i F1 ocjena, što pruža obuhvatniji uvid u sposobnost modela da tačno klasifikuje instance [11].



Slika 8: *Matrica konfuzije; Na horizontalnoj osi su stvarne vrijednosti koje model treba da predvidi, podijeljene na pozitivne i negativne klase. Vertikalna osa prikazuje predviđene vrijednosti koje je model generisao, takođe podijeljene na pozitivne i negativne.*

Najopštija metrika koja se često koristi kod klasifikacije, zbog svoje jednostavnosti i intuitivnosti, jeste **tačnost** (*Accuracy*), koja mjeri ukupan procenat tačno predviđenih instanci.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Međutim, u nebalansiranim skupovima podataka, tačnost može dati zavaravajuću sliku performansi, jer favorizuje dominantnu klasu. U takvim situacijama, preciznost (udio tačno pozitivnih predviđanja od ukupnih pozitivnih predviđanja) i odziv (sposobnost modela da identifikuje stvarno pozitivne instance) pružaju dublji uvid, a F1 ocjena efikasno balansira ove dvije metrike, posebno u scenarijima sa nebalansiranim klasama.

Preciznost (*Precision*) je mjera koja odražava koliko su predikcije modela tačne kada model predviđa određenu klasu. Drugim riječima, preciznost govori koliki je procenat ispravnih predikcija za određenu klasu u odnosu na sve predikcije koje model napravi za tu klasu. Matematički, za pozitivnu klasu, preciznost se izračunava kao odnos tačno pozitivnih predikcija (TP) i zbira tačno pozitivnih i lažno pozitivnih predikcija ($TP + FP$):

$$Precision = \frac{TP}{TP + FP}$$

Preciznost je posebno važna u situacijama u kojima je cijena lažno pozitivnih predikcija visoka. Na primjer, u medicinskoj dijagnostici, minimizacija broja pogrešnih dijagnoza bolesti je od ključnog značaja, dok kod detekcije prevara pogrešno označavanje transakcije kao sumnjive može dovesti do blokade računa ili transakcije, što narušava korisničko iskustvo.

Odziv (*Recall*), poznat i kao osjetljivost ili stopa pravih pozitivnih rezultata, mjeri sposobnost modela da identifikuje i tačno klasifikuje sve stvarne pozitivne instance unutar skupa podataka. Odziv izražava procenat stvarno pozitivnih predikcija u odnosu na ukupan broj pozitivnih slučajeva u podacima. Matematički, na osnovu matrice konfuzije, odziv se izračunava kao odnos tačno pozitivnih predikcija (TP) i zbira tačno pozitivnih i lažno negativnih predikcija ($TP + FN$):

$$Recall = \frac{TP}{TP + FN}$$

Odziv je posebno važan u situacijama gde je ključno ne propustiti nijedan pozitivan slučaj. Na primjer, sistemi za detekciju požara treba da imaju visok odziv, jer je važno da se svaki slučaj dima ili požara detektuje na vrijeme, čak i ako to znači da će povremeno aktivirati lažni alarm.

Dakle, dok se preciznost fokusira na minimizovanje lažno pozitivnih predikcija, odziv se, sa druge strane, fokusira na minimizovanje lažno negativnih predikcija. Visoka preciznost znači da je većina predikcija koje model označi kao pozitivne zaista pozitivna, dok visok odziv znači da model uspijeva prepoznati većinu stvarno pozitivnih slučajeva.

U idealnom scenariju, model bi trebalo da ima i visoku preciznost i visok odziv, odnosno da se minimizuju i lažno pozitivni i lažno negativni rezultati, ali u praksi često postoji kompromis između ove dvije metrike. Poboljšanje jedne od ovih metrika može dovesti do pogoršanja druge. Zbog toga se često koristi **F1 ocjena**, koja je harmonijska sredina preciznosti i odziva, i pruža balansiranu procjenu modela, posebno kod skupova podataka u kojima su nebalansirane klase. F1 ocjena omogućava uravnotežen pristup ocjenjivanja modela, uzimajući u obzir i koliko su predikcije tačne (preciznost) i koliko je model efikasan u otkrivanju svih stvarnih pozitivnih slučajeva (odziv) [12].

F1 ocjena se dobija kao:

$$F1 = 2\frac{Precision \cdot Recall}{Precision + Recall}$$

odnosno

$$F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

F1 ocjena može uzimati vrijednosti od 0 do 1, gdje 1 označava savršenu preciznost i odziv, a 0 označava najlošiji mogući rezultat.

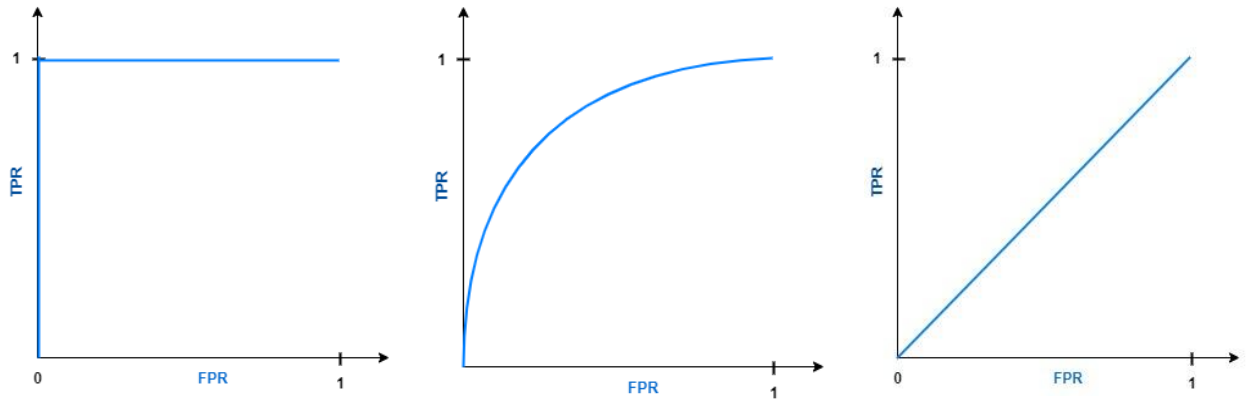
U situacijama gdje je važno razlikovati sposobnost modela da pravilno klasifikuje sve klase, a posebno u binarnoj klasifikaciji, koristi se **površina ispod *ROC* krive**, koja procjenjuje sposobnost modela da razlikuje između klase od interesa i ostalih klasa.

Površina ***AUC*** (*Area Under the Curve*) ispod ***ROC* krive** (*Receiver Operating Characteristic curve*) jedan je od ključnih alata za ocjenjivanje klasifikacionih modela, pružajući uvid u njihovu sposobnost razlikovanja između klasa pri različitim pragovima odlučivanja [13], [14], [15]. Naime, problem sa metrikama koje su zasnovane na matrici konfuzije ogleda se u tome što matrica konfuzije pruža informacije o učinku klasifikacije samo za jedan određeni prag odlučivanja. Na taj način se ne dobija potpuna slika o tome kako se model ponaša pri različitim pragovima. Za razliku od toga, kod *ROC* krive, za slučaj binarne klasifikacije, za različite pragove izračunavaju se stope tačnih pozitivnih rezultata (*TPR*), poznate i kao odziv, kao i stope lažnih pozitivnih rezultata (*FPR*). Zatim se formira *ROC* kriva koja nastaje povezivanjem tačaka čije koordinate na horizontalnoj i vertikalnoj osi predstavljaju odgovarajuće vrijednosti za *FPR* i *TPR*, respektivno. Vrijednosti *TPR* i *FPR* se definišu kao:

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

AUC predstavlja površinu ispod *ROC* krive, na taj način kvantifikujući ukupnu performansu *ROC* krive, te služi kao metrika koja odražava sposobnost modela da razlikuje pozitivne i negativne primjere, pri različitim pragovima. *AUC* može imati vrijednost između 0 i 1, gdje vrijednost 1 označava savršenu klasifikaciju (slika 9). Visoka vrijednost AUC ukazuje na to da klasifikator ima dobru performansu u razlikovanju između pozitivnih i negativnih primjera kroz većinu pragova odlučivanja, dok niska vrijednost ukazuje na lošu sposobnost razlikovanja.



Slika 9: *Primjeri ROC krive sa različitim AUC vrijednostima: lijevo AUC = 1, u sredini 0.5<AUC<1, desno AUC = 0.5*

Svaka od ovih metrika pruža jedinstvene uvide u performanse modela, i njihov odabir zavisi od specifičnih ciljeva projekta i prirode podataka koji se analiziraju.

2.6 Preprilagođavanje i potprilagođavanje

Preprilagođavanje (*Overfitting*) i potprilagođavanje (*Underfitting*) predstavljaju dva primarna izazova u procesu treniranja modela mašinskog učenja (slika 10), koji su usko povezani sa konceptima biasa i varijanse.

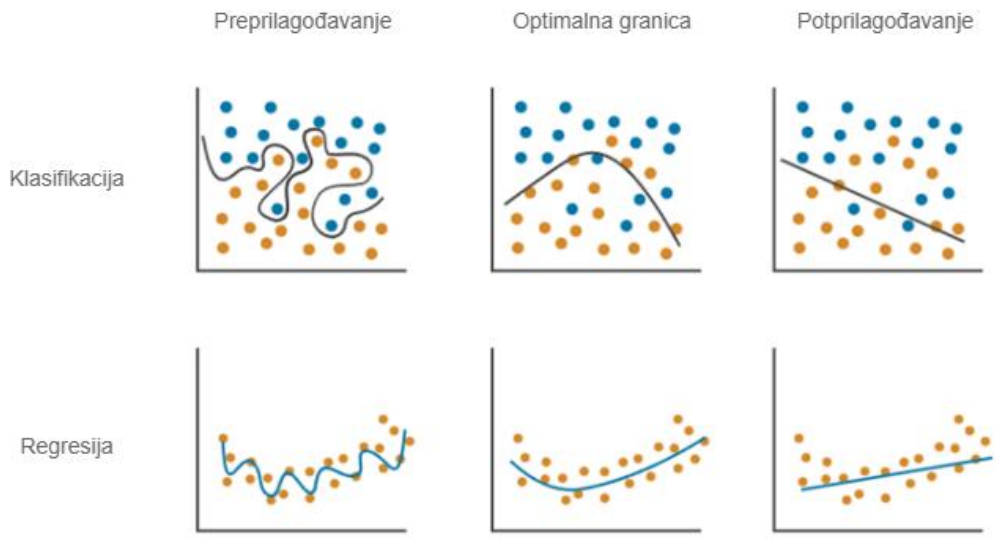
Bias predstavlja sistematsku pristrasnost modela, koja nastaje kada model ne može dovoljno dobro reprezentovati kompleksnost stvarnih podataka, često zbog njegove prevelike pojednostavljenosti. Visok bias vodi do potprilagođavanja, koje se dešava kada model nije dovoljno složen da nauči osnovne obrasce iz trening podataka, što rezultuje slabom tačnošću ili preciznošću kako na trening skupu, tako i na test skupu. Uzroci potprilagođavanja, pored nedovoljnog kapaciteta modela, uključuju i neadekvatnu količinu trening podataka, kao i nedovoljno treniranje.

S druge strane, varijansa ukazuje na promjenljivost modelovih performansi pri treniranju na različitim skupovima podataka. Visoka varijansa dovodi do preprilagođavanja, gdje model pokazuje odlične performanse na trening podacima, ali loše na neviđenim podacima, usljed pretjerane prilagođenosti specifičnostima trening seta. Ovo je posljedica tendencije modela da postane prekomjerno osjetljiv na specifičnosti i šum prisutan u trening setu, umjesto da nauči generalizovane obrasce. Takva situacija obično proizlazi iz prevelike složenosti modela,

manifestovane kroz obiman broj parametara, nedovoljne raznovrsnosti trening podataka, ili suviše dugačkog procesa treniranja, koji omogućava da model „zapamti" trening podatke, umjesto da nauči generalizovane obrazce.

Izazov u mašinskom učenju leži u pronalaženju prave ravnoteže između biasa i varijanse (problem poznat kao *Bias-Variance Tradeoff*), optimizujući model tako da efikasno uči iz podataka. Da bi se postigao ovaj balans, mogu se koristiti različite strategije. Na primjer, tehnike regularizacije smanjuju varijansu, bez značajnog povećanja pristrasnosti, dok tehnike poput ranog zaustavljanja treniranja sprječavaju prekomjerno učenje na trening setu. O ovim tehnikama biće više riječi u nastavku rada. Povećanje raznovrsnosti trening podataka i primjena unakrsne validacije dodatno poboljšavaju procjenu performansi modela i smanjuju rizik od preprilagođavanja.

Kako bias i varijansa igraju ključnu ulogu u određivanju da li će model biti podložan preprilagođavanju ili potprilagođavanju, cilj je minimizovati i jedno i drugo. To podrazumijeva pažljivo usklađivanje složenosti modela: odabir modela koji je dovoljno sofisticiran da identifikuje i nauči ključne obrasce iz podataka (čime se smanjuje bias), dok istovremeno ne postaje toliko specijalizovan da izgubi fleksibilnost i sposobnost prilagođavanja novim situacijama (čime se kontroliše varijansa).



Slika 10: *Primjer preprilagođavanja, optimalne klasifikacije, odnosno regresije, i potprilagođavanja, respektivno [16]*

3 Neuronske mreže i duboko učenje

3.1 Uvod

Neuronske mreže predstavljaju moćan model u mašinskom učenju, sposoban za prepoznavanje složenih obrazaca i donošenje odluka na osnovu podataka. Inspirisane strukturom i funkcijama ljudskog mozga, koriste mrežu međusobno povezanih jedinica - neurona, kako bi procesuirale informacije. Neuronske mreže predstavljaju fundamentalni alat modernog mašinskog učenja i imaju široku primjenu u rješavanju različitih problema, uključujući prepoznavanje slika, govora, predviđanje vremenskih serija i upravljanje autonomnim vozilima.

Koncept neuronskih mreža razvijen je iz potrebe da se razumiju principi po kojima ljudski mozak obrađuje informacije. U vještačkim neuronskim mrežama, ovaj princip realizuje se kroz težine pridružene vezama među neuronima, koje su analogne sinapsama u biološkom neuronu, i aktivacione funkcije, koje određuju nivo aktivacije neurona u mreži. Težine predstavljaju parametre koje mreža prilagođava tokom učenja kako bi optimizovala svoje performanse, dok aktivacione funkcije omogućavaju nelinearnost, što je ključno za prepoznavanje složenih obrazaca.

Ovaj pristup, koji uključuje prilagođavanje težina i primjenu aktivacionih funkcija za modelovanje složenih obrazaca u podacima, ključan je za razvoj mnogih savremenih algoritama dubokog učenja koji koriste višeslojne neuronske mreže za rješavanje zadataka, koji su ranije bili izvan dosega tradicionalnih algoritama. Iako vještačke neuronske mreže ne mogu u potpunosti imitirati kompleksnost ljudskog mozga, postigle su značajne uspjehe u brojnim oblastima, primjenom sličnih principa obrade informacija. Njihova fleksibilnost i sposobnost da se prilagode širokom spektru problema čine ih jednim od najvažnijih alata u današnjoj eri tehnoloških inovacija.

3.2 Osnovni elementi neuronskih mreža

Neuronske mreže sastoje se od neurona, osnovnih gradivnih jedinica koje simuliraju procesiranje informacija, nalik onom u biološkom mozgu. Svaki neuron u mreži prima ulazne signale, obrađuje ih, koristeći težine i aktivacionu funkciju, te proizvodi izlazni signal. Ovi procesi unutar neurona čine temelj neuronske mreže, omogućavajući modeliranje složenih funkcija i prepoznavanje obrazaca u podacima.

Ulazi u neuron predstavljaju podatke koji se obrađuju. Oni mogu biti sirovi podaci iz skupa podataka ili izlazi drugih neurona u mreži.

Težine u vještačkim neuronskim mrežama ključni su parametri koji određuju uticaj pojedinačnih ulaza na izlaz neurona. One se prilagođavaju tokom učenja, omogućavajući mreži da uskladi svoje predikcije s očekivanim ishodima. Ovaj proces prilagođavanja, vođen greškom između stvarnog i predviđenog izlaza, omogućava neuronskim mrežama da „uče" iz podataka i poboljšaju svoje performanse.

Aktivaciona funkcija je mehanizam koji omogućava neuronu da uvede nelinearnost pri izračunavanju svog izlaza, omogućavajući modeliranje složenih problema. Često korišćene aktivacione funkcije uključuju sigmoidnu, hiperbolički tangens i ReLU (*Rectified Linear Unit*), od kojih svaka ima specifične karakteristike i različite primjene unutar neuronskih mreža.

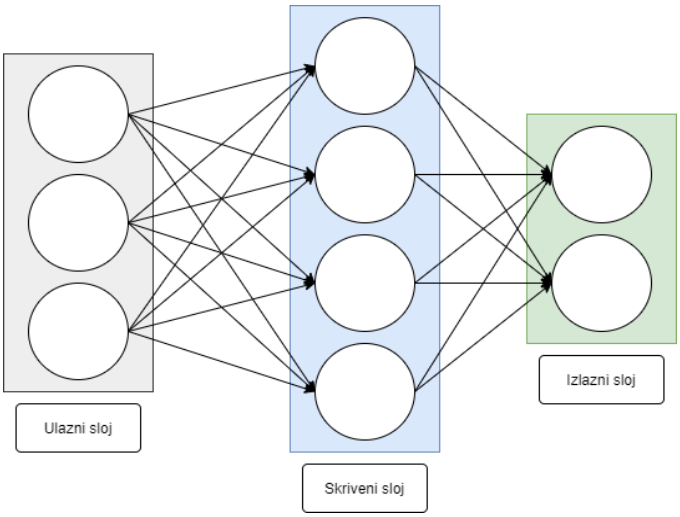
Izlaz neurona predstavlja konačnu vrijednost koju neuron generiše nakon primjene aktivacione funkcije na ukupan težinski sumirani ulaz. Ovaj izlaz se dalje može proslijediti kroz mrežu ili koristiti kao dio konačnog odgovora neuronske mreže.

3.3 Arhitektura i tipovi neuronskih mreža

Arhitektura neuronskih mreža osmišljena je kao složen sistem za obradu i analizu podataka, omogućavajući mreži da prepoznaje obrasce i donosi odluke na temelju ulaznih informacija. Višeslojna struktura omogućava dubinsku analizu i interpretaciju složenih podataka, što čini neuronske mreže pogodnim rješenjem za širok spektar izazovnih zadataka, uključujući klasifikaciju i regresiju.

Osnovna arhitektura neuronskih mreža sastoji se od tri ključna sloja: ulaznog sloja - koji prihvata podatke, jednog ili više skrivenih slojeva - koji vrše složene obrade i ekstrakciju karakteristika iz podataka, i izlaznog sloja - koji generiše krajnje predikcije na osnovu obrađenih informacija (slika 11).

3.3.1 Osnovna arhitektura neuronskih mreža



Slika 11: *Osnovna struktura neuronske mreže - sastavljena od ulaznog, skrivenog i izlaznog sloja*

Ulazni sloj predstavlja inicijalnu tačku u kojoj neuronska mreža prima sirove podatke. Broj neurona u ovom sloju usklađen je s dimenzijama ulaznog vektora, omogućavajući direktnu integraciju informacija u mrežu. Ovi neuroni ne vrše nikakve obrade, već samo prenose informacije dalje kroz mrežu.

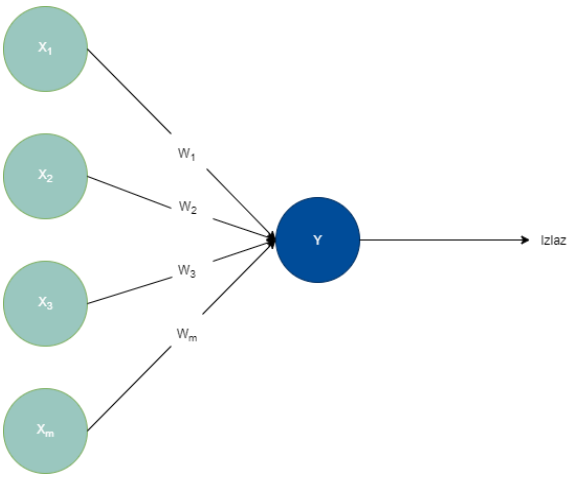
Skriveni slojevi predstavljaju srž neuronske mreže i ključni su za analizu i obradu informacija. Oni primjenjuju težine i aktivacione funkcije na ulazne podatke, kako bi izdvojili relevantne karakteristike i obrasce. Dinamika i kapacitet mreže, za modelovanje različitih nivoa složenosti, zavise od broja i konfiguracije skrivenih slojeva, što omogućava prilagođavanje mreže za specifične zadatke.

Izlazni sloj transformiše informacije, obrađene kroz skriveni sloj, u krajnje rezultate, odgovarajući specifičnim ciljevima zadatka, nezavisno od toga o kom se tipu predikcije radi.

3.3.2 Perceptron i napredne arhitekture neuronskih mreža

Jedna od najjednostavnijih i najranijih arhitektura neuronskih mreža je *feedforward* mreža, u kojoj informacije teku samo u jednom smjeru, od ulaznog ka izlaznom sloju, bez povratnih veza. Ova arhitektura predstavlja osnovu za mnoge složenije modele, pri čemu je njen osnovni gradivni blok *perceptron* (slika 12). Perceptron karakteriše jednostavna struktura, koja se sastoji od jednog ili više ulaza, neurona koji vrši proces obrade signala i jedinstvenog izlaza. Ključna

karakteristika inicijalne verzije perceptrona je binarna priroda, pri čemu izlaz može imati vrijednost 1 ili 0. Funkcija aktivacije perceptrona generiše izlaz 1 ako je suma ulaznih signala (težinski prilagođenih) pozitivna, dok za negativne vrijednosti sume izlaz postaje 0.



Slika 12: *Perceptron*

Matematički, perceptron može biti opisan formulom, gdje je izlaz neurona y definisan kao:

$$y = f\left(\sum_{i=1}^{m} w_i x_i + b\right)$$

gdje je f aktivaciona funkcija, w_i težina pridružena i-tom ulazu, x_i i-ti ulazni signal, a b bias.

Uprkos svojoj jednostavnosti, perceptron je imao ključnu ulogu u postavljanju temelja za složenije neuronske mreže, kao što su višeslojni perceptroni i druge napredne arhitekture. Njegovo osnovno ograničenje je nemogućnost rješavanja nelinearnih problema, što je dovelo do razvoja višeslojnih perceptrona (*Multilayer Perceptron - MLP*). Uvođenjem skrivenih slojeva između ulaznog i izlaznog sloja, kao i nelinearnih aktivacionih funkcija, višeslojni perceptroni omogućavaju modelovanje složenijih, nelinearnih odnosa. Ova sposobnost prepoznavanja i učenja složenih obrazaca u podacima značajno poboljšava performanse mreža u zadacima kao što su klasifikacija i regresija.

U oblasti vještačke inteligencije, raznolikost arhitektura neuronskih mreža igra ključnu ulogu u rješavanju širokog spektra izazova, specifičnih za različite primjene i domene. Arhitekture poput potpuno povezanih, konvolucionih, rekurentnih i drugih tipova neuronskih mreža, prilagođene su različitim tipovima podataka i problema, pri čemu koriste specifične tehnike učenja. Klasifikacija neuronskih mreža zasniva se na nekoliko ključnih aspekata, uključujući

arhitekturu mreže, način protoka informacija i specifične zadatke koje te mreže mogu rješavati. U nastavku slijedi kratak pregled nekih značajnijih tipova neuronskih mreža:

- Potpuno povezane neuronske mreže (*Fully Connected Neural Networks – FCNNs*) čine osnovu mnogih sistema mašinskog učenja. Karakteriše ih arhitektura u kojoj je svaki neuron iz jednog sloja povezan sa svim neuronima prethodnog i sljedećeg sloja, što omogućava njihovu široku primjenu. Ove mreže su temelj za razumijevanje osnovnih principa neuronskih mreža, pružajući čvrstu osnovu za razvoj složenijih modela.

- Autoenkoderi su specifičan tip neuronskih mreža dizajniran za smanjenje dimenzionalnosti i učenje kompaktnih reprezentacija podataka. Neke od najčešćih primjena autoenkodera uključuju kompresiju podataka, redukciju šuma, detekciju anomalija i generisanje podataka. Njihova sposobnost da nauče rekonstruisati ulazne podatke čini ih posebno pogodnim za otkrivanje anomalija, jer mogu identifikovati podatke koji značajno odstupaju od očekivanih obrazaca. Ova osobina ih čini ključnim u zadacima poput nadzora sistema, sigurnosti mreža i prevencije prevara. S obzirom na to da je fokus rada na autoenkoderima, naredno poglavlje će detaljno predstaviti njihovu strukturu i primjene.

- Konvolucione neuronske mreže (*Convolutional Neural Networks - CNNs*) su vrsta neuronskih mreža koje koriste konvolucione slojeve za automatsko izdvajanje karakteristika iz podataka. Nalaze široku primjenu u obradi videa, teksta, zvuka, a naročito slika, za šta se najčešće smatraju dominantnim tipom mreža. Ova sposobnost ih čini izuzetno efikasnim u zadacima vizuelne analize, kao što su prepoznavanje objekata i klasifikacija slika, zahvaljujući sposobnosti da nauče složene obrasce i teksture iz vizuelnih podataka.

- Rekurentne neuronske mreže (*Recurrent Neural Networks - RNNs*) posebno su prilagođene za obradu sekvencijalnih podataka, poput vremenskih serija i prirodnog jezika, omogućavajući modelu da zadrži informacije iz prethodnih koraka, kako bi bolje razumio trenutni kontekst. Zbog svoje sposobnosti da obrađuju informacije koje zavise od redosleda, često se koriste u zadacima kao što su: mašinsko prevođenje (npr. ranije verzije *Google Translate*), generisanje teksta, prepoznavanje govora (npr. glasovni asistenti poput *Siri* i *Google Assistant*), analiza vremenskih serija (npr. predviđanje cijena akcija ili vremenskih uslova), itd.

- Transformerske neuronske mreže (*Transformer Neural Networks - TNNs*) predstavljaju savremenu arhitekturu dubokog učenja, poznatu po efikasnom radu sa podacima koji imaju sekvencijalnu strukturu. Postale su standard u obradi prirodnog jezika, s mogućnošću razumijevanja kontekstualnih veza i generisanja teksta, donoseći revolucionarne promjene u oblastima kao što su mašinsko prevođenje i automatsko generisanje teksta (npr. *ChatGPT, LLaMA* itd.). Za razliku od rekurentnih neuronskih mreža, koje podatke obrađuju korak po korak, transformeri omogućavaju paralelnu obradu, čineći ih bržim i efikasnijim u radu sa dugim sekvencama. Ipak, *RNN*-ovi i dalje nalaze primjenu u specifičnim scenarijima, poput predikcija vremenskih serija, obrade podataka u realnom vremenu i rada sa manjim skupovima podataka, gdje su efikasniji i manje zahtjevni.

- Generativne suparničke mreže (*Generative Adversarial Networks - GANs*) predstavljaju arhitekturu neuronskih mreža koja se sastoji od dva modela, koji rade u međusobnoj konkurenciji: generativnog modela i diskriminatornog modela. Generativni model ima zadatak da stvori nove podatke, slične onima iz trening skupa, dok diskriminatorni model procjenjuje koliko su generisani podaci slični stvarnim. Ovaj proces iterativnog nadmetanja omogućava *GAN* mrežama da generišu izuzetno realistične podatke, kao što su slike (npr. generisanje lica ili pejzaža), zvuk (npr. sinteza govora) i video (npr. animacije ili *deepfake* sadržaji).

- Mreže dubokog učenja sa podrškom (*Deep Reinforcement Learning Networks*) predstavljaju spoj dubokog učenja i metoda učenja sa podrškom, omogućavajući modelima da se adaptiraju i optimizuju kroz direktnu interakciju sa okolinom, težeći postizanju specifičnih ciljeva uz maksimizovanje akumuliranih nagrada. Ovakvi modeli pronalaze svoju primjenu u širokom spektru naprednih aplikacija, uključujući razvoj sofisticiranih igračkih agenata, upravljanje autonomnim vozilima, te kreiranje efikasnih sistema za automatizovanu trgovinu. Njihova sposobnost da samostalno uče iz iskustva čini ih izuzetno moćnim alatima za rješavanje složenih problema, koji zahtijevaju dinamičko donošenje odluka.

3.4 Propagacija unaprijed

Centralni princip funkcionisanja neuronskih mreža jeste propagacija unaprijed (*Forward Propagation*), koja omogućava prenos informacija od ulaznog sloja prema izlaznom, prolazeći

kroz sve skrivene slojeve. Ovaj proces je ključan za izračunavanje izlaznih vrijednosti, na osnovu ulaznih podataka, koje se potom koriste za predikcije.

Tokom propagacije unaprijed, podaci se transformišu kroz niz koraka: težine pridružene svakom ulazu u neuron određuju značaj tih ulaza, odnosno njihov uticaj na izlaz, dok aktivacione funkcije uvode nelinearnost u model, što omogućava mreži da modeluje složene relacije u podacima i generiše izlazne vrijednosti.

Kada podaci prolaze kroz mrežu, aktivacija svakog neurona u sloju l zavisi od aktivacija neurona iz prethodnog sloja $l-1$. Ovaj proces može se formalno opisati sljedećom jednačinom [17]:

$$a_j^l = \sigma\left(\sum_k w_{jk}^l \cdot a_k^{l-1} + b_j^l\right)$$

gdje:

- a_j^l predstavlja aktivaciju j-tog neurona u sloju l, odnosno njegov izlazni signal
- σ predstavlja aktivacionu funkciju
- $\sum_k w_{jk}^l \cdot a_k^{l-1}$ predstavlja sumiranje po svim neuronima prethodnog sloja $(l-1)$ čiji izlaz predstavlja ulaz u j-ti neuron sloja l, pri čemu:
    - w_{jk}^l predstavlja težinu koja povezuje neuron k iz sloja $l-1$ sa neuronom j u sloju l
    - a_k^{l-1} predstavlja aktivaciju neurona k u sloju $l-1$, tj. izlaznu vrijednost neurona prethodnog sloja
- b_j^l predstavlja bias za j-ti neuron u sloju l

Kroz propagaciju unaprijed, podaci se kreću kroz mrežu, sloj po sloj, pri čemu svaki neuron obrađuje informacije na osnovu svojih ulaza, težina i biasa, primjenjujući aktivacionu funkciju kako bi izračunao izlaznu vrijednost. Ovaj proces se ponavlja kroz sve slojeve, sve dok se ne dobije konačni izlaz mreže, odnosno izlazi neurona iz poslednjeg sloja.

Nakon generisanja izlaza, mreža procjenjuje grešku u odnosu na stvarne, ciljne vrijednosti. Ova greška služi kao osnova za optimizaciju težina tokom procesa obučavanja, čime mreža iterativno poboljšava svoje performanse.

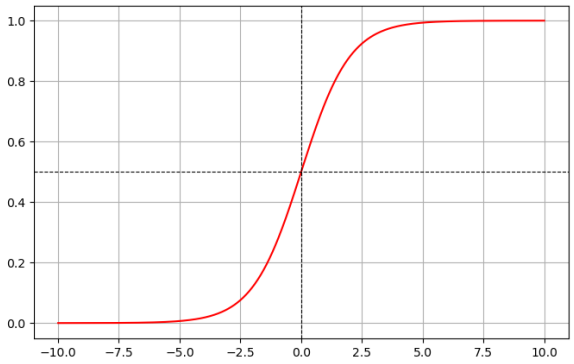
3.5 Aktivacione funkcije

Aktivacione funkcije su jedan od ključnih elemenata neuronskih mreža, jer omogućavaju uvođenje nelinearnosti u model. Na ovaj način, neuronske mreže stiču sposobnost da modeliraju kompleksne obrasce i veze u podacima, koje linearni modeli ne mogu efikasno obraditi. Aktivacione funkcije se primjenjuju na izračunate težinske sume ulaznih signala neurona, određujući tako izlazni signal neurona.

Neke od najčešće korišćenih aktivacionih funkcija date su u nastavku.

- Sigmoidna funkcija (slika 13) ograničava izlaz na opseg između 0 i 1, što je čini korisnom u binarnoj klasifikaciji. Svojstvo sigmoidne funkcije, da izlaz ograničava unutar ovog opsega, čini je dobrim izborom za situacije gdje izlaz treba interpretirati kao vjerovatnoću. Matematički, sigmoidna funkcija, za ulaz x, definiše se kao:

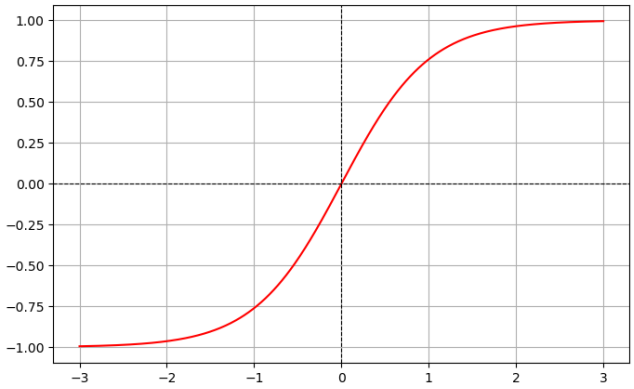
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



Slika 13: *Sigmoidna aktivaciona funkcija*

- Hiperbolički tangens (*tanh*) je funkcija koja izlaze transformiše na opseg od -1 do 1, čime se postiže centriranje izlaza oko nule (slika 14). Zbog svoje sposobnosti da efikasno distribuira aktivacije unutar mreže, ova funkcija se često koristi u skrivenim slojevima, pružajući bolje performanse u mnogim situacijama, posebno kada je poželjno da izlazi budu simetrični u odnosu na nulu. Matematički, *tanh* funkcija se definiše formulom:

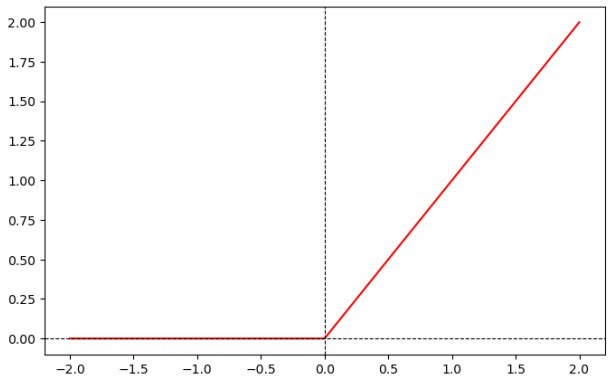
$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$



Slika 14: *Hiperbolički tangens funkcija*

- ReLU (*Rectified Linear Unit*) funkcija vrši aktivaciju neurona samo kada primaju pozitivne ulazne vrijednosti, dok negativne postavlja na nulu (slika 15), čime se doprinosi smanjenju računarske složenosti. *ReLU* je posebno popularna u obradi i analizi vizuelnih podataka kroz konvolucione neuronske mreže, jer ubrzava proces učenja i pomaže u rješavanju problema nestajućih gradijenata (o čemu će biti više riječi u nastavku rada), čime doprinosi boljem prepoznavanju i klasifikaciji objekata na slikama. Zahvaljujući jednostavnosti i empirijskom uspjehu, *ReLU* je podrazumijevani izbor u velikom broju modela. Matematički, *ReLU* funkcija se definiše formulom:

$$\sigma(x) = max(0, x)$$



Slika 15: *ReLU funkcija*

Izbor odgovarajuće aktivacione funkcije može imati značajan uticaj na performanse neuronske mreže. Pravilan izbor funkcije zavisi od specifičnosti problema i prirode podataka koji se obrađuju.

3.6 Obučavanje neuronskih mreža algoritmom propagacije unazad i algoritmi optimizacije

Propagacija unazad (*Backpropagation*) predstavlja osnovni algoritam za obučavanje neuronskih mreža, pri čemu se težine i biasi u neuronskoj mreži iterativno prilagođavaju, kako bi se smanjila razlika između stvarnog i predviđenog izlaza, koja se mjeri primjenom neke odabrane funkcije gubitka [18], [19].

Proces obuke, za svaki dio skupa podataka za obučavanje (*batch*), teče kroz dvije faze. Prva faza je, ranije pomenuta, propagacija unaprijed, u kojoj se podaci propagiraju kroz slojeve sve do izlaza, gdje se računa vrijednost funkcije gubitka (greška), odnosno razlika između dobijenog i stvarnog izlaza. Nakon što se dobije greška, vrši se druga faza, odnosno tzv. *backward pass*, u kojoj se greška prosljeđuje unazad, sloj po sloj, kako bi se odredilo koliko je svaki parametar mreže (težine i biasi) uticao na tu grešku. Drugim riječima, u toku propagacije unazad, računaju se parcijalni izvodi funkcije gubitka u odnosu na težine i biase. Tako dobijeni parcijalni izvodi se dalje koriste za modifikaciju težina i biasa u mreži, s ciljem minimizacije greške. Način na koji se ti parcijalni izvodi koriste za modifikaciju parametara, zavisi od odabranog algoritma optimizacije. Cijeli proces obučavanja obavlja se dok vrijednost funkcije gubitka ne bude dovoljno mala, ili dok ne istekne zadati broj epoha obuke.

Kada je riječ o algoritmima optimizacije, koji se koriste za modifikaciju parametara mreže na osnovu gradijenta, osnovna tehnika je metod gradijentnog spusta [20]. Osim njega, često se koriste i brojni drugi algoritmi, npr. stohastički gradijentni spust, RMSprop, Adam, itd. U nastavku će biti opisan metod gradijentnog spusta, kao osnovni među njima, kao i Adam - napredniji algoritam koji je korišćen u radu.

3.6.1 Metod gradijentnog spusta

Metod gradijentnog spusta koristi se za pronalaženje minimuma neke funkcije. U svakoj iteraciji najprije se izračunava gradijent funkcije, tj. vektor parcijalnih izvoda po svim promjenljivim, u trenutnoj tački. Kako gradijent pokazuje smjer najvećeg porasta vrijednosti funkcije, da bi se vrijednost funkcije smanjila, treba pomjerati parametre u suprotnom smjeru, odnosno u smjeru suprotnom od gradijenta. Na taj način, pri svakom koraku „silazi se niz padinu" funkcije, nastojeći da se dođe do minimuma. U slučaju složenijih funkcija, sa većim brojem lokalnih minimuma, nije uvijek garantovano da će se ovim metodom pronaći globalni minimum. Međutim, u praksi se, u velikom broju slučajeva, pokazalo da ovakvi lokalni

minimumi često daju dovoljno dobre rezultate, a neke naprednije tehnike optimizacije posjeduju mehanizme za prevazilaženje ovakvih problema.

Za ilustraciju ovog metoda, može se posmatrati prost primjer funkcije jedne promjenljive. Neka je data funkcija $f(x)$ i početna vrijednost parametra $x_0 = 0$:

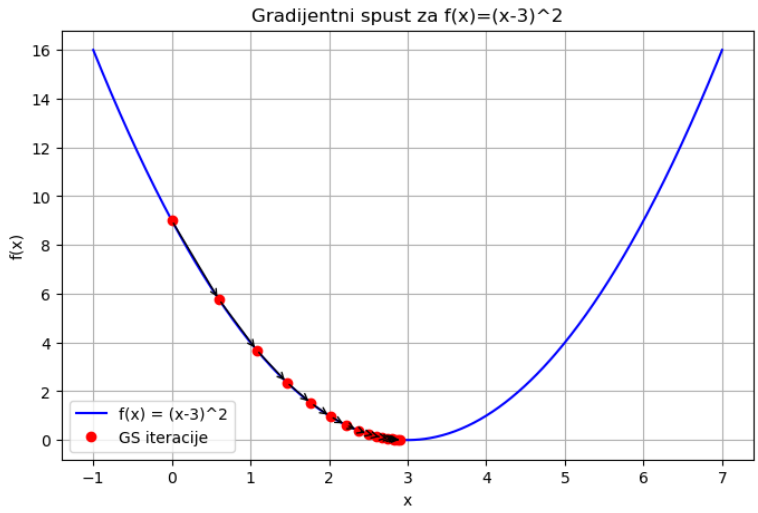
$$f(x) = (x-3)^2$$

Algoritam ažurira vrijednost za x u i-tom koraku kao (slika 16):

$$x_i = x_{i-1} - \eta f'(x_{i-1})$$

gdje je sa η označena stopa učenja (*Learning Rate*), koja predstavlja jedan od parametara učenja koji se zadaju unaprijed. Stopa učenja služi za regulisanje veličine koraka pomjeranja parametra x, u smjeru suprotnom od gradijenta. Premala vrijednost stope učenja može dovesti do spore konvergencije, dok prevelika može uzrokovati da dođe do preskakanja minimuma, te je pravilan izbor stope učenja od velike važnosti. Algoritam se ponavlja dok se ne obradi zadati broj epoha, ili dok se ne dostigne minimum funkcije, što se detektuje kad izvod u x_i postane jednak 0, ili približno 0. U složenijim primjenama, naročito u višedimenzionalnim slučajevima, vrše se i provjere da li je norma gradijenta postala manja od nekog praga, da li je tačnost na skupu za validaciju zadovoljavajuća, itd.

Iako je u gore navedenom primjeru data funkcija sa jednom promjenljivom, u opštem slučaju se postupa analogno, tako što se računa vektor parcijalnih izvoda funkcije po svim promjenljivim i svaka od njih se ažurira koristeći odgovarajuću vrijednost parcijalnog izvoda.



Slika 16: *Primjer algoritma gradijentnog spusta*

3.6.2 Adam optimizator

Adam je napredniji optimizator koji na drugačiji način vrši ažuriranje parametara mreže – težina i biasa. Naime, svakom parametru mreže, u Adam algoritmu, pridružene su dvije pomoćne promjenljive, koje se obično inicijalizuju na 0: m, koja se naziva prvi moment i v, koja se naziva drugi moment [21]. Osim pomoćnih promjenljivih, koriste se sljedeći hiperparametri: η koji predstavlja stopu učenja, ε koji predstavlja malu vrijednost, koja se obično koristi da se izbjegne dijeljenje sa nulom, kao i β_1 i β_2, odnosno tzv. faktori „zaboravljanja".

Nakon što se propagacijom unazad dobiju parcijalni izvodi funkcije greške po svakom parametru mreže, vrši se ažuriranje prvog i drugog momenta – na primjer, za neku težinu w_i, vrijednosti prvog i drugog momenta u k-toj iteraciji se računaju kao:

$$m_{k,i} = \beta_1 m_{k-1,i} + (1-\beta_1) g_{k,i}$$

$$v_{k,i} = \beta_2 v_{k-1,i} + (1-\beta_2)\left(g_{k,i}\right)^2$$

gdje je $g_{k,i}$ vrijednost parcijalnog izvoda funkcije greške po težini w_i, u k-toj iteraciji.

Na isti način, vrši se ažuriranje odgovarajućih vrijednosti prvog i drugog momenta za biase. Najčešće vrijednosti za faktore „zaboravljanja" su $\beta_1 = 0.9$ i $\beta_2 = 0.999$. Naziv ovih konstanti potiče od toga što ukazuju na to u kojoj mjeri stara vrijednost momenta utiče na njegovu novu vrijednost. Što je vrijednost ovih konstanti veća, to stare vrijednosti za momente imaju veću težinu, odnosno sporije se „zaboravljaju".

Kako se momenti obično inicijalizuju na 0, a β_1 i β_2 na vrijednosti blizu 1, postoji tendencija da u prvim iteracijama momenti imaju vrijednost blizu nule. Zbog toga se, nakon prethodno opisanog ažuriranja, vrši sljedeća korekcija:

$$\hat{m}_{k,i} = \frac{m_{k,i}}{1-\beta_1^k}$$

$$\hat{v}_{k,i} = \frac{v_{k,i}}{1-\beta_2^k}$$

Finalno, svaki parametar mreže se ažurira na osnovu odgovarajućih korigovanih momenata. Na primjer, za težinu w_i, u k-toj iteraciji:

$$w_k = w_{k-1} - \eta \frac{\hat{m}_{k,i}}{\sqrt{\hat{v}_{k,i}} + \varepsilon}$$

Analogno se ažuriraju i biasi, a kako je Adam optimizator generalno manje osjetljiv na izbor početnih hiperparametara, za stopu učenja se najčešće uzima vrijednost $\eta = 0.001$. Adam generalno brže konvergira od običnog gradijentnog spusta, pogotovo na kompleksnijim strukturama, kao što su duboke neuronske mreže.

3.7 Duboko učenje

Za razliku od tradicionalnih metoda mašinskog učenja, koje se oslanjaju na ručno odabrane karakteristike (*features*) i relativno jednostavnije modele za obradu podataka, duboko učenje uvodi koncept dubokih neuronskih mreža. Ove mreže, koje se sastoje od višestrukih slojeva, ili „dubina", sposobne su za automatsko izvlačenje i učenje visoko apstraktnih karakteristika iz velikog obima sirovih podataka, čime se eliminiše potreba za eksplicitnim inženjeringom karakteristika.

Jedna od ključnih razlika između dubokog učenja i tradicionalnih pristupa leži u ovoj sposobnosti mreža da samostalno otkrivaju korisne reprezentacije podataka, što omogućava dubokim modelima da efikasno rješavaju složene zadatke poput prepoznavanja slika, generisanja teksta, razumijevanja prirodnog jezika, itd. Ova samostalnost u izvlačenju karakteristika omogućava dubokim modelima da uče složene obrasce u podacima koje tradicionalni modeli, ograničeni predefinisanim karakteristikama, možda ne bi mogli prepoznati.

Međutim, duboko učenje sa sobom nosi i različite izazove, uključujući potrebu za velikim količinama podataka i značajnim računarskim resursima za obučavanje modela. Složenost dubokih neuronskih mreža i njihova potreba za specijalizovanom hardverskom podrškom, čine duboko učenje resursno zahtjevnijim, u poređenju sa tradicionalnim pristupima.

Osim toga, dok duboko učenje može pružiti značajne rezultate u mnogim aplikacijama, načini na koje model donosi odluke često su nejasni, što može biti ograničenje u kontekstima gdje je potreban visok stepen transparentnosti i razumijevanja modela, kao što je to, na primjer, u slučaju medicinskih i pravnih sistema.

Uprkos ovim izazovima, duboko učenje je transformisalo mnoge aspekte vještačke inteligencije, omogućavajući razvoj sofisticiranih aplikacija, koje su bile nezamislive sa tradicionalnim tehnikama. Njegova sposobnost da se prilagodi i uči iz kompleksnih i velikih skupova podataka, čini ga nezamjenljivim alatom u modernoj tehnološkoj eri, dok tradicionalno mašinsko učenje ostaje vrijedan pristup za aplikacije gdje je dostupnost podataka ograničena, potrebna brza implementacija, ili je interpretabilnost modela od ključnog značaja. Izbor između

dubokog i tradicionalnog mašinskog učenja zavisi od specifičnih potreba projekta, dostupnosti podataka i infrastrukture, kao i od ciljeva i ograničenja zadatka koji se rješava.

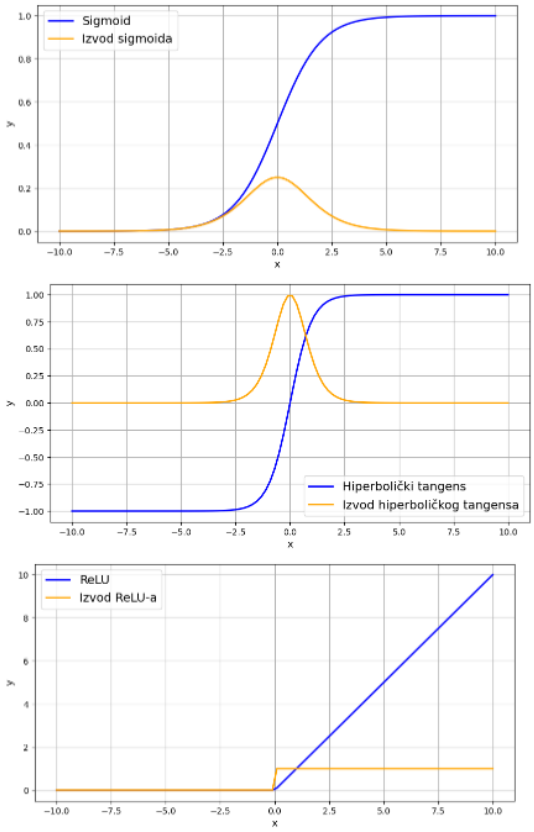
3.7.1 Problemi nestajućih i eksplodirajućih gradijenata

U procesu dubokog učenja, tokom obučavanja kompleksnih neuronskih mreža, dva značajna izazova su problem nestajućih i problem eksplodirajućih gradijenata [4], [22].

Problem nestajućih gradijenata nastaje kada vrijednosti gradijenata postanu previše male tokom propagacije unazad, što efektivno smanjuje ili potpuno zaustavlja ažuriranje težina u ranijim slojevima neuronske mreže. Ovaj fenomen je posebno izražen u dubokim mrežama, gdje gradijenti, koji se izračunavaju tokom propagacije unazad, često opadaju eksponencijalno kako informacija putuje kroz slojeve, posebno kada se koriste određene aktivacione funkcije, poput sigmoidne i hiperboličkog tangensa. Ove funkcije, kada se ulazne vrijednosti približe svojim ekstremnim granicama, imaju izvode koji teže nuli (slika 17), što uzrokuje da gradijenti postanu vrlo mali tokom propagacije unazad. Kao rezultat, slojevi bliži ulaznom sloju uče veoma sporo ili čak uopšte ne napreduju, što značajno otežava obučavanje mreže.

Za razliku od njih, aktivaciona funkcija *ReLU,* na primjer, pokazala se efikasnom u ublažavanju problema nestajućih gradijenata, zbog svoje linearne prirode za pozitivne ulaze. Ovo linearno ponašanje za pozitivne ulaze funkcije osigurava da gradijenti, za vrijeme propagacije unazad, ostaju veliki i stabilni, što omogućava efikasnije i konzistentnije ažuriranje težina.

Suprotan problem, problem eksplodirajućih gradijenata, događa se kada vrijednosti gradijenata eksponencijalno rastu tokom propagacije unazad, što može dovesti do nestabilnosti i divergencije u procesu obučavanja. Ovaj fenomen može uzrokovati prekomjerno velika ažuriranja težina, čineći mrežu nestabilnom. Problem eksplodirajućih gradijenata je obično lakše rješiv od problema nestajućih gradijenata, a jedna od najčešće primijenjenih tehnika, za rješavanje ovog problema, jeste ograničenje gradijenta (*Gradient clipping*).



Slika 17: *Prikaz funkcija sigmoida, hiperboličkog tangensa i ReLU sa njihovim izvodima.*

Ograničenje gradijenata (*Gradient Clipping)* je metoda koja ograničava, odnosno „siječe", vrijednosti gradijenata na definisanu maksimalnu vrijednost, ako gradijenti premaše tu granicu. Ovo osigurava da gradijenti, bez obzira na njihovu veličinu, ne premaše prag koji bi mogao uzrokovati nestabilnost u procesu ažuriranja težina. Tehnika se može primijeniti na različite načine, ali dva najčešća pristupa su ograničenje norme (*Norm Clipping*) i ograničenje vrijednosti (*Value Clipping*) [23].

- Ograničenje norme je metoda koja ograničava gradijente tako što postavlja njihovu L2 normu na definisanu maksimalnu vrijednost. L2 norma vektora je matematička mjera njegove dužine u Euklidskom prostoru i definisana je kao kvadratni korijen sume kvadrata svih komponenti vektora. Za vektor $g = [g_1, g_2, \dots, g_n]$, L2 norma je:

$$\|g\|_2 = \sqrt{g_1^2 + g_2^2 + \cdots + g_n^2}$$

  Ako ukupna norma gradijenata premaši ovu vrijednost, sve komponente vektora se skaliraju proporcionalno, tako da ukupna norma bude jednaka maksimalnoj dozvoljenoj vrijednosti. Ovo osigurava da se pravac gradijenta očuva, dok se njegova norma smanjuje, kako bi se spriječila nestabilnost.

- Ograničenje vrijednosti je metoda kod koje se svaka dimenzija gradijenta pojedinačno provjerava, i ako njena vrijednost premaši definisane granice (maksimalnu i minimalnu), ograničava se na te granice. Za razliku od ograničenja norme, ova metoda ne očuvava pravac gradijenta, ali je efikasna u sprječavanju izuzetno velikih promjena težina, koje mogu dovesti do divergencije.

3.7.2 Tehnike za poboljšanje performansi modela

Za poboljšanje performansi modela, posebno zarad sprječavanja prekomjernog prilagođavanja podacima, koriste se različite tehnike regularizacije. Ove tehnike pomažu u smanjenju složenosti modela i poboljšavaju njegovu sposobnost generalizacije na neviđenim podacima, dodavanjem kaznenog faktora funkciji gubitka, stabilizacijom obuke, povećanjem raznolikosti skupa za obučavanje, itd. Neke od najpoznatijih tehnika su [24]:

- L1 regularizacija – tehnika koja dodaje sumu apsolutnih vrijednosti težina funkciji gubitka:

$$L_{reg} = L + \lambda \sum_{i=1}^{N} |w_i|$$

gdje je L osnovna funkcija gubitka, λ regularizacioni parametar, koji kontroliše snagu regularizacije, w_i predstavlja težine modela, a N je ukupan broj težina u modelu. Veća vrijednost λ znači jaču regularizaciju, jer se više „kažnjava" složenost modela.

L1 regularizacija može dovesti do rijetkih modela (*sparse models*), pri čemu neke težine postaju nula, što eliminiše nepotrebne ili manje važne karakteristike. Na ovaj način, ne samo da smanjuje složenost modela, već i pomaže u selekciji karakteristika koje najviše doprinose performansama modela.

- L2 regularizacija – tehnika koja dodaje sumu kvadratnih vrijednosti težina funkciji gubitka:

$$L_{reg} = L + \lambda \sum_{i=1}^{N} {w_i}^2$$

gdje su L, λ, w_i i N definisani kao kod L1 regularizacije.

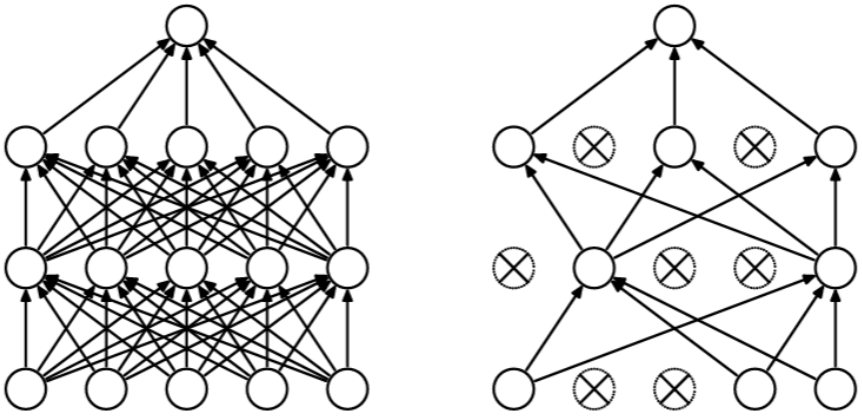
Za razliku od L1 regularizacije, koja linearno kažnjava težine i može dovesti do rjeđih modela postavljanjem manjih težina na nulu, L2 regularizacija strožije kažnjava veće težine (jer kazneni faktor sadrži kvadrate težina), ali ne eliminiše manje težine, te ne dovodi do rjeđih modela. Umjesto toga, L2 regularizacija smanjuje veličinu svih težina, „gurajući" ih prema manjim vrijednostima. Ovaj pristup pomaže u generalizaciji modela tako što smanjuje uticaj manje relevantnih karakteristika i stabilizuje obuku modela. Takođe, L2 regularizacija smanjuje veličinu gradijenata tokom propagacije unazad, što je posebno korisno u prevenciji eksplodirajućih gradijenata.

Regularizacija, bilo kroz L1 ili L2, predstavlja moćan alat za poboljšanje generalizacije modela i za smanjenje rizika od prekomjernog prilagođavanja trening podacima. Dok L1 regularizacija pomaže u selekciji važnih karakteristika, L2 regularizacija smanjuje veličinu težina i povećava stabilnost obuke. Oba pristupa pomažu modelu da postane robusniji i otporniji na šum u podacima.

- *Dropout* - još jedan oblik regularizacije specifičan za neuronske mreže, koji funkcioniše tako što nasumično isključuje određeni procenat neurona tokom faze obuke (slika 18), koristeći predefinisanu stopu *dropout*-a [25]. Cilj ove tehnike je sprječavanje prekomjerne zavisnosti modela od specifičnih neurona, čime se poboljšava sposobnost mreže da uči generalizovane obrasce i distribuira proces učenja kroz mrežu.

  Nasumično isključivanje neurona se realizuje tako što se izlazi od odabranih neurona postavljaju na nulu, čime se ti neuroni efektivno isključuju iz procesa obuke tokom jednog koraka. Kako bi se optimizovale performanse modela, vrši se podešavanje vrijednosti hiperparametra, koji se naziva stopa *dropout*-a. Može imati vrijednosti u opsegu između 0 i 1, a najčešće korišćene vrijednosti kreću se između 0.2 i 0.5, što znači da se nasumično isključuje između 20% i 50% neurona odgovarajućeg sloja, tokom svakog koraka obuke.

  Jedna od ključnih karakteristika ove tehnike je to da se tokom faze testiranja svi neuroni sloja aktiviraju, ali se njihovi izlazi skaliraju, na način što se množe sa faktorom $(1 - p)$, gdje je p stopa *dropout*-a, koja je korišćena tokom obuke. Ovaj proces skaliranja osigurava konzistentnost između obuke i evaluacije modela, jer je očekivana vrijednost sume izlaza u sloju ista, tokom obije faze. *Dropout* se pokazao izuzetno efikasnim u smanjenju prekomjernog prilagođavanja, naročito u zadacima kao što su prepoznavanje slika, gdje mreže mogu brzo „zapamtiti" podatke.



Slika 18: *Standardna neuronska mreža – lijevo; mreža nakon primjene dropout-a – desno; [25]*

- Normalizacija *batch*-a (*Batch Normalization*) je tehnika koja se koristi za stabilizaciju i ubrzavanje procesa obučavanja neuronske mreže, koja se ogleda u dodavanju posebnih slojeva za normalizaciju mreži. Osnovni cilj ovakvog sloja za normalizaciju *batch*-a je da se, prilikom propagacije svakog *batch*-a kroz mrežu u toku obučavanja, izvrši normalizacija vrijednosti, koje su izlazi neurona prethodnog sloja.

  U prvom koraku ovog procesa, za konkretan *batch* i neki konkretan neuron mreže, iz sloja koji prethodi sloju za normalizaciju, vrijednosti na nivou *batch*-a koje izlaze iz tog neurona se normalizuju na način da imaju srednju vrijednost 0 i standardnu devijaciju 1.

  Dakle, ako se sa $x_i^{(k)}$ označi izlaz iz k-tog neurona tog sloja za i-ti *sample* iz *batch*-a, u sloju za normalizaciju se vrši sljedeća transformacija vrijednosti $x_i^{(k)}$ [26]:

$$\hat{x}_i^{(k)} = \frac{x_i^{(k)} - \mu_B^{(k)}}{\sqrt{\left(\sigma_B^{(k)}\right)^2 + \epsilon}}$$

  gdje je $\mu_B^{(k)}$ srednja vrijednost izlaza $x_i^{(k)}$ iz *k*-tog neurona tog sloja, u trenutnom batch-u B veličine m, dok je $\sigma_B^{(k)}$ standardna devijacija ovih vrijednosti tj.:

$$\mu_B^{(k)} = \frac{1}{m}\sum_{i=1}^{m} x_i^{(k)}$$

$$\sigma_B^{(k)} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(x_i^{(k)} - \mu_B^{(k)}\right)^2}$$

Vrijednost ϵ je mali broj, tipično 10^{-5} ili 10^{-7}, dodat da bi se izbjeglo dijeljenje sa nulom.

Pored ove normalizacije vrijednosti, sljedeći korak koji se radi je uvođenje skaliranja (faktora $\gamma^{(k)}$) i pomjeranja (faktora $\beta^{(k)}$) za neuron, tako da se dobijene vrijednosti $\hat{x}_i^{(k)}$ dalje transformišu kao:

$$y_i^{(k)} = \gamma^{(k)} \cdot \hat{x}_i^{(k)} + \beta^{(k)}$$

Faktori $\gamma^{(k)}$ i $\beta^{(k)}$ su parametri sloja za normalizaciju *batch*-a koje mreža uči, pri čemu svaki neuron prethodng sloja ima odgovarajuće zasebne parametre. Ovi parametri daju mreži mogućnost da „vrati" distribuciju vrijednosti na optimalan nivo, jer ponekad najbolji oblici distribucija nisu oni sa srednjom vrijednošću 0 i standardnom devijacijom 1, već nešto drugo što se nauči tokom treniranja. Parametar $\gamma^{(k)}$ omogućava skaliranje standardizovanih vrijednosti, odnosno povećanje ili smanjenje raspona vrijednosti, dok parametar $\beta^{(k)}$ omogućava da se vrijednosti pomjere ka drugoj srednjoj vrijednosti, za slučaj da je to korisno za performanse mreže, iz perspektive određenog neurona. Na ovaj način se zadržava fleksibilnost mreže da naknadno prilagodi distribuciju podataka. Konačno, ovako dobijene vrijednosti $y_i^{(k)}$ se prosljeđuju narednom sloju mreže, odnosno predstavljaju izlaz sloja za normalizaciju.

Kad je u pitanju faza testiranja, kako se često testira po 1 ulaz, ne postoji *batch*, pa samim tim ni odgovarajuća srednja vrijednost i standardna devijacija koje se koriste za normalizaciju vrijednosti. Umjesto toga, tokom obučavanja, računa se eksponencijalni pokretni prosjek za srednju vrijednost $\mu_r^{(k)}$ i varijansu $\sigma^2{}_r^{(k)}$, koji akumulira dobijene vrijednosti iz *batch*-eva, uz korišćenje faktora momentuma α. Vrijednost ovog faktora određuje koliko se vrijednosti iz starijih *batch*-eva „zaboravljaju", odnosno kolika se važnost daje kasnijim *batch*-evima. Tako izračunate vrijednosti koriste se za normalizaciju u fazi testiranja:

$$\mu_r^{(k)} = \alpha\mu_r^{(k)} + (1-\alpha)\mu_B^{(k)}$$

$$\sigma^2{}_r^{(k)} = \alpha\sigma^2{}_r^{(k)} + (1-\alpha)\sigma^2{}_B^{(k)}$$

Osim normalizacije pomoću ovih eksponencijalnih pokretnih prosjeka, vrši se i skaliranje i pomjeranje tako dobijenih vrijednosti pomoću naučenih faktora $\gamma^{(k)}$ i $\beta^{(k)}$.

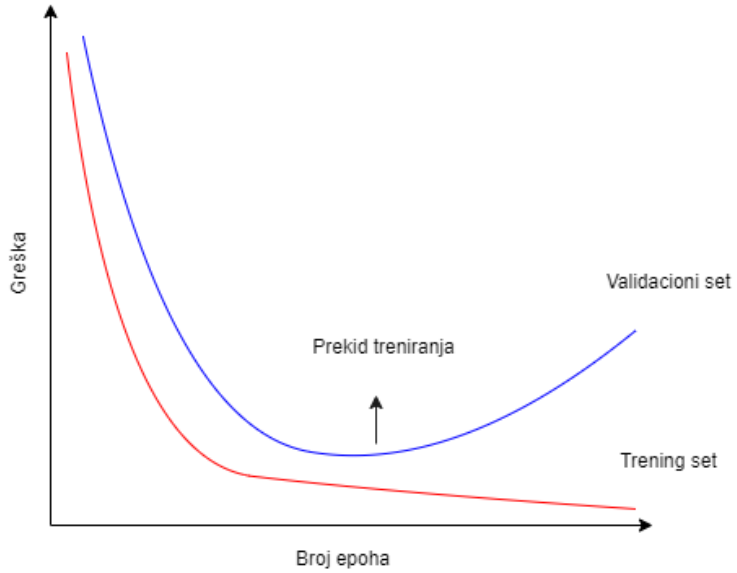
Tehnika normalizacije *batch*-a pokazala se kao veoma značajna tehnika u obučavanju dubokih mreža. Normalizacija se najčešće primjenjuje prije primjene aktivacionih funkcija, čime se stabilizuje i ubrzava konvergencija, što doprinosi bržem treniranju. Ova tehnika je značajna i za smanjenje šuma i ekstremnih vrijednosti, a obezbjeđuje se i manja osjetljivost na izbor inicijalnih parametara. Osim toga, značajan je i uticaj na smanjenje problema poput nestajućih i eksplodirajućih gradijenata.

- Augmentacija podataka je tehnika koja povećava raznolikost skupa za obučavanje, primjenom različitih transformacija na postojeće podatke, bez potrebe za prikupljanjem novih. Ove transformacije mogu uključivati rotaciju, promjenu veličine, dodavanje šuma, promjenu osvjetljenja na slikama, itd. Cilj je simulirati različite scenarije koje model može sresti u stvarnom svijetu, čime se poboljšava njegova sposobnost generalizacije.

  Ova tehnika je posebno korisna u oblastima poput računarske vizije i obrade prirodnog jezika, gdje je važno da modeli prepoznaju obrasce u širokom spektru prikaza i konteksta. Augmentacija smanjuje rizik od prekomjernog prilagođavanja, čineći model robusnijim i sposobnijim za rad sa neviđenim podacima.

- Rano zaustavljanje (*Early Stopping*) je jednostavna, ali efikasna tehnika regularizacije, koja prati performanse modela na validacionom skupu podataka tokom treniranja i prekida obuku kada se detektuje da performanse na validacionom skupu počinju da opadaju, iako se na skupu za obučavanje i dalje poboljšavaju (slika 19). Na taj način se sprječava to da model postane previše prilagođen specifičnostima skupa za obučavanje. Najčešće se koristi na način takav da se unaprijed definiše broj epoha k (tzv. parametar *patience*), tako da, nakon što obučavanje modela prođe kroz k uzastopnih epoha na kojima performanse na validacionom skupu nisu poboljšane, proces obučavanja se obustavlja i zadržavaju se parametri modela sa najboljim performansama na validacionom skupu.

  Ova tehnika eliminiše potrebu za ručnim podešavanjem broja epoha i automatski prepoznaje trenutak u kojem se performanse pogoršavaju, prekidajući obuku modela u optimalnom trenutku i zadržavajući model koji najbolje generalizuje.



Slika 19: *Prikaz tehnike ranog zaustavljanja – trenutak prekida obuke kada greška na validacionom skupu počinje da raste, a greška na trening skupu opada.*

Primjena različitih tehnika regularizacije i stabilizacije, poput L1/L2 regularizacije, *dropout*-a, augmentacije podataka i ranog zaustavljanja, pomaže modelima da postignu dobre performanse i smanje rizik od prekomjernog prilagođavanja. Takođe, tehnike poput normalizacije *batch*-a mogu doprinijeti boljoj stabilnosti tokom treniranja, smanjujući probleme poput eksplodirajućih ili nestajućih gradijenata. Pravilna kombinacija ovih metoda omogućava izgradnju robusnih modela koji se efikasno prilagođavaju različitim zadacima.

3.7.3 Autoenkoderi

Autoenkoderi su specijalizovani tipovi neuronskih mreža, dizajnirani za efikasno transformisanje ulaznih podataka u kompaktan latentni prostor, a zatim za rekonstrukciju originalnih ulaznih podataka, sa što manjim gubitkom informacija. Arhitektura autoenkodera sastoji se od dva glavna dijela: enkodera i dekodera [27], [28], [29].

Enkoder vrši transformaciju ulaznog vektora $\boldsymbol{x} \in R^D$ u latentni vektor $\boldsymbol{z} \in R^d$, gdje je $d < D$, efikasno kompresujući podatke u manje dimenzionalni prostor. Ova operacija se matematički može izraziti kao $\boldsymbol{z} = f(\boldsymbol{x})$, gdje je $f: R^D \rightarrow R^d$ funkcija enkodera, koja primjenjuje niz linearnih transformacija na ulazne podatke, u kombinaciji sa nelinearnim aktivacionim funkcijama. Ova kombinacija omogućava mreži da modeluje složenije nelinearne odnose u podacima.

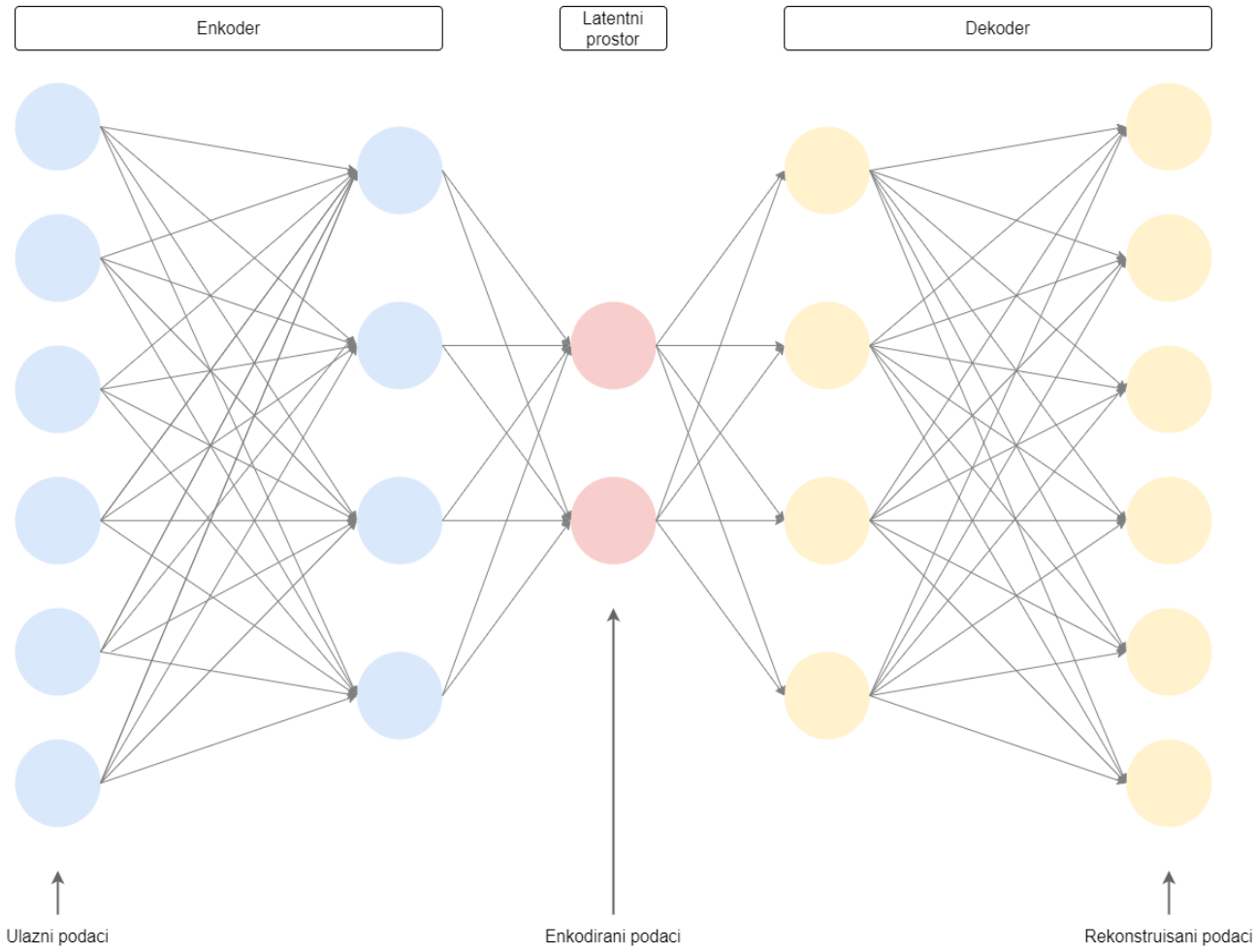
Dekoder zatim pokušava da rekonstruiše originalni ulazni vektor $\boldsymbol{x}$ na osnovu latentnog vektora $\boldsymbol{z}$, što rezultuje rekonstruisanim vektorom $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{x}} \in R^D$. Ovaj proces može se izraziti kao $\hat{\boldsymbol{x}} = g(\boldsymbol{z})$, gdje je $g: R^d \rightarrow R^D$ funkcija dekodera. Cilj je minimizovati razliku između $\boldsymbol{x}$ i $\hat{\boldsymbol{x}}$, što se često postiže minimizacijom funkcije gubitka, kao što je srednja kvadratna greška (*MSE*):

$$L(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \frac{1}{D}\sum_{i=1}^{D}(x_i - \hat{x}_i)^2$$

gdje D predstavlja dimenziju ulaznog vektora $\boldsymbol{x}$, x_i originalne vrijednosti iz ulaznog vektora, a $\hat{x}_i$ rekonstruisane vrijednosti.

Jedna od ključnih karakteristika autoenkodera jeste latentni prostor, ili takozvani *bottleneck* sloj, koji se nalazi između enkodera i dekodera (slika 20). On ograničava količinu informacija koje mogu proći kroz mrežu, podstičući model da identifikuje i sačuva najvažnije karakteristike ulaznih podataka, za njihovu efikasnu rekonstrukciju. Tipično je dizajniran sa manjim brojem neurona u odnosu na ulazni i izlazni sloj, čime se omogućava kompresija podataka, uz zadržavanje ključnih informacija.

Pravilno dimenzionisan latentni prostor pomaže autoenkoderu da nauči generalizovane reprezentacije podataka, smanjujući rizik od prekomjernog prilagođavanja. Premali latentni prostor može dovesti do gubitka važnih informacija, dok preveliki može smanjiti efikasnost kompresije i povećati složenost modela. Balansiranje između kompresije i rekonstrukcije omogućava modelu da dobro filtrira šum i nauči generalizovane obrasce, umjesto da ih samo „zapamti".



Slika 20: *Arhitektura autoenkodera*

Pored pomenutog standardnog autoenkodera, postoji nekoliko ključnih tipova autoenkodera, čiji će kratak pregled biti dat u nastavku [30].

- **Autoenkoderi za uklanjanje šuma (*Denoising Autoencoders*)** predstavljaju varijantu autoenkodera koji se treniraju s ciljem da efikasno ignorišu šum u ulaznim podacima. Ulazni podaci se namjerno korumpiraju dodavanjem Gausovog, ili nekog drugog tipa šuma, postavljanjem dijela podataka iz ulaza na vrijednost 0, ili sličnim metodama, a autoenkoder se trenira da rekonstruiše originalne, nekorumpirane podatke. Na ovaj način, model se forsira da nauči suštinske osobine podataka i da ignoriše nebitne varijacije, što poboljšava njegovu robusnost i dovodi do učenja korisnije reprezentacije ulaza u latentnom prostoru.

- **Variacioni autoenkoderi (*Variational Autoencoders*)** razlikuju se od klasičnih autoenkodera po tome što svaki ulaz mapiraju na raspodjelu u latentnom prostoru, umjesto na jednu „fiksnu“ tačku [31]. Konkretno, enkoder proizvodi vektor srednjih

vrijednosti $\boldsymbol{\mu}$ i vektor standardnih devijacija $\boldsymbol{\sigma}$, pri čemu svaka komponenta tih vektora odgovara jednoj dimenziji latentnog prostora (dakle, za svaki neuron/dimenziju postoji po jedno μ_i i σ_i). Iz tako dobijene višedimenzionalne Gausove raspodjele uzorkuje se latentni vektor, koji dekoder zatim rekonstruiše u izlazne podatke. Na taj način model, pored same rekonstrukcije, uči i generisanje novih primjera nasumičnim uzorkovanjem iz naučene raspodjele. Primjene ovih autoenkodera uključuju generisanje slika i teksta, detekciju anomalija, itd.

- **Sekvencijalni autoenkoderi (*Sequential Autoencoders*)** prilagođeni su za rad sa sekvencijalnim podacima, poput vremenskih serija ili teksta. U ovim modelima, kako bi se efikasno obradili i generisali takvi podaci, enkoderi i dekoderi imaju arhitekturu rekurentnih neuronskih mreža ili transformerskih modela. Sekvencijalni autoenkoderi se često primjenjuju u oblastima poput obrade prirodnog jezika, gdje pomažu u generisanju i rekonstrukciji sekvenci teksta.

- **Konvolucioni autoenkoderi (*Convolutional Autoencoders*)** koriste konvolucione slojeve kako u enkoderskom, tako i u dekoderskom dijelu mreže, što ih čini posebno pogodnim za obradu slika i drugih vizuelnih podataka. Konvoluciona arhitektura omogućava efikasno izdvajanje i rekonstrukciju prostorno-organizovanih karakteristika slika, pri čemu se koristi manji broj parametara u poređenju sa klasičnim, potpuno povezanim mrežama. Kao rezultat toga, kompleksnost modela se smanjuje, a ključne informacije se zadržavaju, osiguravajući kvalitetnu obradu vizuelnih sadržaja.

- **Rijetki autoenkoderi (*Sparse Autoencoders*)** predstavljaju specifičnu vrstu autoenkodera, koja primjenjuje tehniku regularizacije, kako bi se obezbijedila rijetkost u aktivacijama skrivenih neurona. Umjesto fokusiranja na težine, kao kod standardnih tehnika regularizacije, rijetki autoenkoderi penalizuju prekomjerne aktivacije u mreži, čime se model podstiče da u svakom trenutku aktivira samo mali broj neurona. Ova karakteristika omogućava modelu da se fokusira na najvažnije osobine ulaznih podataka, što poboljšava njegovu sposobnost generalizacije.

- **Kontraktivni autoenkoderi (*Contractive Autoencoders*)** primjenjuju dodatnu regularizaciju u funkciji gubitka, kako bi osigurali da male promjene u ulaznim podacima ne utiču značajno na latentni prostor. Ovo se postiže dodavanjem regularizacionog člana, koji kažnjava velike promjene u latentnom prostoru u odnosu na male promjene u

ulaznim podacima. Na taj način, model postaje robusniji i uči stabilne karakteristike koje dosljedno predstavljaju ulaz, čak i u prisustvu šuma ili varijacija, odnosno model se bolje fokusira na suštinske informacije u podacima [32].

Generalno, autoenkoderi efikasno smanjuju dimenzionalnost podataka kompresujući ulazne informacije u manji latentni prostor. Ovaj proces smanjuje broj varijabli koje treba obraditi, olakšavajući vizualizaciju složenih skupova podataka i poboljšavajući interpretaciju ključnih karakteristika. Osim toga, smanjenje dimenzionalnosti smanjuje računske zahtjeve i ubrzava algoritme mašinskog učenja. Ovo je posebno korisno u oblastima kao što su bioinformatika i genetika, gdje rad sa visokodimenzionalnim podacima može biti zahtjevan, a interpretacija je značajno olakšana prikazivanjem podataka u sažetom, često dvodimenzionalnom prostoru. Na taj način, autoenkoderi izdvajaju ključne karakteristike iz složenih podataka, što pojednostavljuje otkrivanje obrazaca [33].

Izuzetno su korisni i u zadacima detekcije anomalija. Trenirani na „normalnim" podacima, autoenkoderi uče da rekonstruišu tipične primjere iz određenog domena. Zbog toga im je, često, veoma teško precizno rekonstruisati podatke koji odstupaju od onoga što su naučili kao standard, što ih čini izuzetno efikasnim za detekciju anomalija. Ova sposobnost je naročito značajna u finansijskoj industriji, gdje je rano prepoznavanje prevara od ključnog značaja, ali i u analizi industrijskih senzorskih podataka, gdje identifikacija neuobičajenih događaja može spriječiti ozbiljne kvarove. Autoenkoderi su donijeli značajan napredak u pogledu efikasnosti i preciznosti detekcije anomalija, jer mogu automatski prepoznati složene obrasce u različitim tipovima podataka. Njihova otpornost na šum i sposobnost učenja kvalitetnih reprezentacija naročito dolaze do izražaja pri analizi zvučnih signala industrijskih mašina, poput ventilatora, pumpi i motora [34]. Kada se pojave neuobičajeni ili „nenormalni" zvukovi, pomoću autoenkodera moguće ih je pouzdano detektovati, što omogućava pravovremeno održavanje i poboljšava sigurnost i efikasnost proizvodnih procesa. Na taj način, ovi modeli doprinose boljem kvalitetu proizvodnje i prevenciji potencijalno skupih i rizičnih kvarova.

Osim navedenog, važno je napomenuti da se primjena autoenkodera proteže na širok spektar oblasti, uključujući medicinsku dijagnostiku, personalizaciju preporuka u e-komercu, optimizaciju u telekomunikacijama, efikasno upravljanje potrošnjom energije u energetici, kao i mnoge druge. Ovo raznoliko korišćenje demonstrira izuzetnu svestranost autoenkodera u različitim primjenama, od analize složenih podataka, do interakcije sa realnim svijetom. Međutim, iako autoenkoderi pružaju značajne mogućnosti, njihova efikasna primjena zahtijeva

pažljiv odabir arhitekture i podešavanje hiperparametara, kako bi se spriječilo memorisanje ulaznih podataka, a osiguralo učenje suštinskih i korisnih reprezentacija. U narednom dijelu detaljno će biti objašnjeni pristupi implementaciji modela i metodologija korišćena za rješavanje ovih izazova, u okviru specifične primjene.

4 Metodologija i implementacija modela

4.1 Uvod

U savremenim industrijskim okruženjima, ključno je održavati visok nivo produktivnosti, sigurnosti i efikasnosti mašina. Anomalije u radu mašina mogu dovesti do prekida proizvodnje, štete na opremi i sigurnosnih incidenata, te je njihova detekcija od velikog značaja za smanjenje ovih rizika i optimizaciju troškova održavanja. Primjenom metoda vještačke inteligencije i mašinskog učenja na analizu zvuka mašina, moguće je precizno detektovati anomalije razlikujući uobičajene operativne zvuke od neobičnih akustičnih obrazaca koji, ukazuju na potencijalne probleme. Autoenkoderi, kao napredni modeli dubokog učenja, pokazali su se posebno efikasni u obradi i rekonstrukciji kompleksnih zvučnih signala, omogućavajući precizno identifikovanje odstupanja od uobičajenog zvučnog profila mašine.

Iako se većina industrijskih postrojenja još uvijek oslanja na konvencionalne metode detekcije, poput periodičnih fizičkih inspekcija i različitih senzorskih sistema, te metode često imaju svoja ograničenja. Fizičke inspekcije zahtijevaju znatan ljudski trud i često detektuju probleme samo kada postanu ozbiljni, što dovodi do neplaniranih prekida i visokih troškova popravki. Senzorske inspekcije, iako pružaju zadovoljavajuće rezultate, često zahtijevaju kompleksne instalacije i visoke troškove, ograničavajući njihovu primjenu u mnogim industrijskim scenarijima. Stoga, postoji izražena potreba za razvojem naprednijih, efikasnijih i ekonomičnijih rješenja za detekciju anomalija.

Autoenkoderi, specijalizovani za obradu i analizu kompleksnih podataka poput zvučnih signala, omogućavaju identifikaciju suptilnih promjena u operativnom ponašanju mašina, koje tradicionalne metode često ne prepoznaju. Korišćenjem ovih modela omogućava se ranije otkrivanje problema, što doprinosi proaktivnim intervencijama i značajno smanjuje rizik od ozbiljnijih kvarova ili zastoja u proizvodnji.

Fokus ovog istraživanja je na razvoju efikasnog modela za detekciju anomalija u radu industrijskih mašina korišćenjem autoenkodera, s posebnim akcentom na primjenu u okviru MIMII skupa podataka.

U nastavku teksta dat je detaljniji pregled istraživanja, uključujući opis skupa podataka, proces ekstrakcije karakteristika, pregled arhitekture modela, proces treniranja i metodologiju testiranja.

4.2 MIMII skup podataka

MIMII (*Malfunctioning Industrial Machine Investigation and Inspection*) skup podataka je specijalizovana kolekcija zvučnih snimaka, dizajnirana za istraživanje i razvoj metoda za detekciju anomalija u industrijskim mašinama. Ovaj skup sadrži snimke rada četiri različite vrste industrijskih mašina: ventila, pumpi, ventilatora i kliznih šina, u normalnim i abnormalnim stanjima (tabela 1).

Ventili u ovom skupu su solenoidni, što znači da rade na principu elektromagnetnih impulsa koji omogućavaju njihovo naizmjenično otvaranje i zatvaranje u okviru industrijskog procesa. Pumpe su vodene i namenjene su cirkulaciji, tj. ispumpavanju vode iz rezervoara i njenom vraćanju nazad. Ventilatori obezbjeđuju neprekidni protok gasa ili vazduha u industrijskim postrojenjima. Klizne šine predstavljaju linearne klizne sisteme koji se sastoje od pokretne platforme i baze, omogućavajući precizno linearno kretanje.

Skup uključuje po sedam pojedinačnih mašina za svaku vrstu, pri čemu svaka mašina može biti različitog modela. Ukupno je prikupljeno 26 092 normalna zvučna segmenta za sve mašine, kao i 6 065 segmenata sa anomalijama poput kontaminacije, curenja, rotacione neravnoteže, oštećenja šina, itd., pri čemu treba napomenuti da su trenutno javno dostupni signali samo za po četiri mašine iz svake vrste (označene sa Model ID 00, 02, 04 i 06).

Snimanje je vršeno korišćenjem mikrofonskog niza TAMAGO-03, koji se sastoji od osam nezavisnih mikrofonskih jedinica, omogućavajući višekanalnu prostornu analizu zvučnih obrazaca. Zvučni segmenti su snimani u trajanju od 10 sekundi, pri čemu je mikrofon za ventile postavljen na udaljenosti od 10 centimetara, a za ostale mašine na 50 centimetara. Svaki segment obuhvata osam odvojenih audio kanala, čime se osigurava detaljna analiza akustičnih karakteristika.

Zvučni snimci rada mašina su digitalizovani kao 16-bitni signali sa uzorkovanjem od 16 kHz i snimljeni su u reverberacionom okruženju, nakon čega je dodavan šum različite energije, koji je odvojeno sniman istom aparaturom u više fabrika. Za simuliranje različitih odnosa energije zvuka mašina i šuma, skup podataka uključuje snimke sa različitim SNR (*Signal-to-Noise Ratio*) vrijednostima od -6 dB, 0 dB i 6 dB, što omogućava procjenu sposobnosti modela da precizno detektuje i klasifikuje zvučne signale pri različitim intenzitetima zvučnog okruženja.

Zvuk svake vrste mašine snimljen je u odvojenim sesijama, što omogućava precizniju izolaciju i analizu njihovih specifičnih zvučnih karakteristika. Korišćenje MIMII skupa podataka omogućava razvoj i validaciju modela autoenkodera za automatsko prepoznavanje odstupanja u radu mašina, što značajno doprinosi povećanju pouzdanosti i efikasnosti procesa održavanja u industrijskim sistemima.

Tip mašine	Model ID	Broj segmenata normalnog zvuka	Broj segmenata abnormalnog zvuka
Ventil	00	991	119
	01	869	120
	02	708	120
	03	963	120
	04	1000	120
	05	999	400
	06	992	120
Pumpa	00	1006	143
	01	1003	116
	02	1005	111
	03	706	113
	04	702	100
	05	1008	248
	06	1036	102
Ventilator	00	1011	407
	01	1034	407
	02	1016	359
	03	1012	358
	04	1033	348
	05	1109	349
	06	1015	361
Klizne šine	00	1068	356
	01	1068	178
	02	1068	267
	03	1068	178
	04	534	178
	05	534	178
	06	534	89
Ukupno		**26092**	**6065**

Tabela 1: *Sadržaj MIMII skupa podataka [1]*

4.3 Mel spektrogram i ekstrakcija karakteristika

Za potrebe ovog rada, u procesu ekstrakcije karakteristika korišćen je Mel spektrogram. Mel spektrogram predstavlja jednu od važnijih reprezentacija vremensko-frekvencijskih karakteristika zvučnih signala, sa primjenama u automatskom prepoznavanju govora, sintezi zvuka, analizi akustičkih scena, analizi zvuka mašina, itd. Ključne komponente Mel

spektrograma su Mel skala, koja predstavlja proizvod transformacije linearne frekvencijske skale u frekvencijsku skalu zasnovanu na ljudskoj percepciji, i banka Mel filtera, koja prikuplja energiju iz frekvencijskih opsega prema rasporedu definisanom Mel skalom.

Proces transformacije linearnog STFT spektrograma u Mel spektrogram koristi linearni spektrogram kao osnovu, koji predstavlja vremensko-frekvencijsku reprezentaciju signala, pri čemu su frekvencije na osi predstavljene kao linearno raspoređeni frekvencijski opsezi (*bins*). Broj *bins*, odnosno opsega frekvencija, zavisi od veličine okvira STFT-a, a svaki *bin* sadrži energiju signala u određenom linearnom opsegu. Iako ovaj spektrogram pruža detaljnu analizu frekvencija zahvaljujući visokoj rezoluciji, on nije prilagođen ljudskoj percepciji, jer ljudsko uho ne opaža sve frekvencijske opsege jednako – posebno u višim frekvencijama, gdje se osjetljivost na razlike smanjuje.

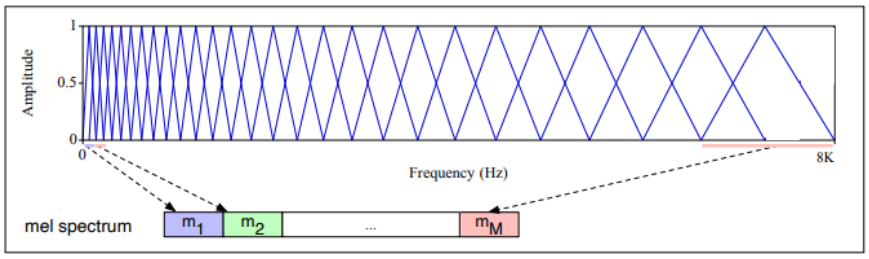
Da bi se spektrogram prilagodio ljudskoj percepciji, koristi se Mel skala, koja određuje kako frekvencije iz linearne skale treba rasporediti tako da odražavaju ljudsku percepciju – niže frekvencije, koje ljudi jasnije razlikuju, su šire razmaknute, čime dobijaju više prostora na frekvencijskoj osi, dok su više frekvencije kompresovane. Ova nelinearnost omogućava naglašavanje nižih frekvencija, koje su ključne za razumijevanje govora i tonaliteta, dok se višim frekvencijama, čije razlike ljudsko uho slabije razlikuje, pridaje manji značaj.

Ova transformacija motivisana je empirijskim istraživanjima osetljivosti ljudskog sluha na različite frekvencijske opsege, a matematički je definisana kao [35]:

$$mel(f) = 1127 \cdot \ln\left(1 + \frac{f}{700}\right)$$

gde je f linearna frekvencija u Hz.

Na osnovu Mel skale određuju se središnje frekvencije i opsezi Mel filtera, koji se pozicioniraju tako da pokrivaju specifične frekvencijske opsege. Ovi filteri su najčešće trouglastog oblika i logaritamski su pozicionirani (slika 21), čime omogućavaju visoku rezoluciju u nižim frekvencijama, dok više frekvencije bivaju kompresovane. Svaki od njih prikuplja energiju iz svog opsega, pri čemu se opsezi međusobno preklapaju radi kontinuiteta u analizi signala.

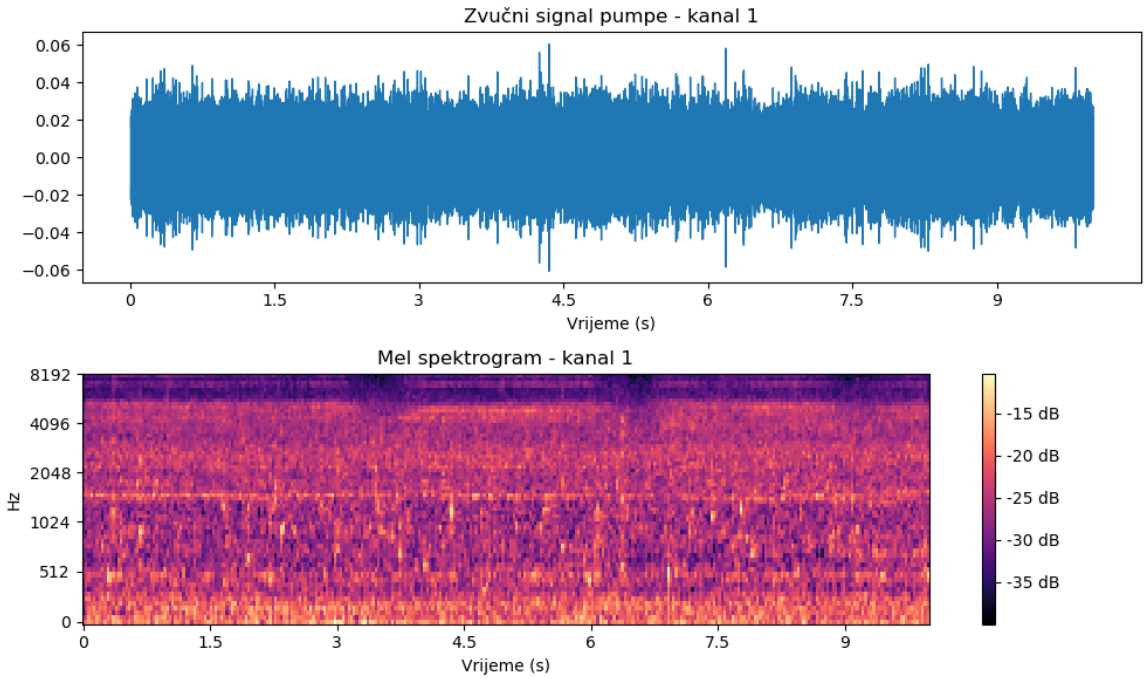


Slika 21: *Mel filter banka (Davis i Mermelstein, 1980). Svaki trouglasti filter, raspoređen logaritamski duž Mel skale, prikuplja energiju iz određenog frekvencijskog opsega [35]*

Unutar svakog trouglastog filtera, obuhvaćeni *bins* nemaju jednaku težinu – energija bliža odgovarajućoj središnjoj frekvenciji filtera ima veću važnost i proporcionalno veću težinu u izračunavanju ukupne energije filtera, dok ivični *bins* imaju manji doprinos. Ovaj proces omogućava da se frekvencijski sadržaj spektra sažme u broj vrijednosti koji odgovara broju Mel filtera. Rezultat primjene Mel filter banke na linearni spektrogram je Mel spektrogram – vremensko-frekvencijska reprezentacija signala, ali sa značajno smanjenim brojem frekvencijskih tačaka (*bins*), koji sada odgovaraju broju filtera (npr. 64 umjesto 1024).

Mel spektrogram ne samo da omogućava efikasniju analizu zvučnih signala, već reflektuje i način na koji ljudi percipiraju zvuk. Ova metodologija prvi put je predstavljena u radu Davisa i Mermelsteina (1980) [36], gdje je prikazana njena primjena u analizi govora, uključujući vizualizaciju Mel filter banke, koja jasno ilustruje način na koji energija iz različitih frekvencijskih opsega doprinosi konačnom Mel spektru.

Što se tiče procesa ekstrakcije karakteristika iz audio signala za potrebe ovog rada, on započinje korišćenjem *Librosa* biblioteke za učitavanje audio signala, koja sadrži širok spektar funkcionalnosti za efikasnu obradu multikanalnih signala. Nakon učitavanja, određeni kanali se selektivno obrađuju pomoću funkcije *librosa.feature.melspectrogram*. Ova funkcija interno koristi kratkotrajnu Furijeovu transformaciju (STFT) sa mogućnošću zadavanja različitih parametara. U ovom radu, korišćena je veličina okvira od 1024 uzorka i korak preklapanja od 512 uzoraka. U istom koraku, primjenjuje se Mel filter banka sa 64 filtera, čime se konvertuje frekvencijski spektar dobijen iz STFT-a u Mel spektrogram. Mel spektrogram se dalje skalira u logaritamsku skalu (dB) kako bi se dinamički raspon prilagodio perceptivnim osobinama ljudskog sluha, čime se dobijaju amplitude koje bolje reflektuju ljudsku percepciju intenziteta zvuka. Rezultat je Mel spektrogram (slika 22) koji pruža detaljnu vremensko-frekvencijsku reprezentaciju zvučnih karakteristika, usklađenu s ljudskom percepcijom, što omogućava preciznije analize u aplikacijama mašinskog učenja.



Slika 22: *Primjer zvučnog signala pumpe i odgovarajućeg Mel spektrograma*

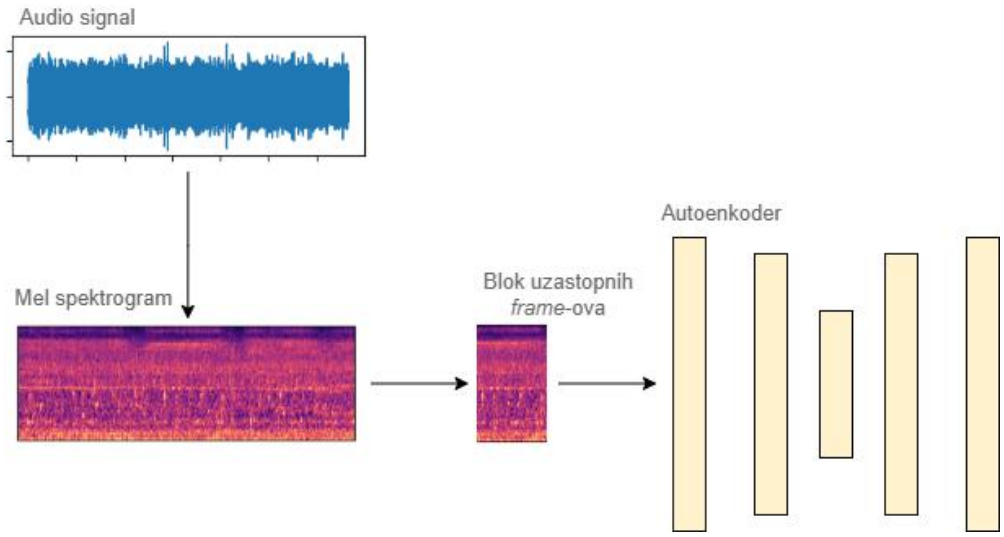
4.4 Arhitektura autoenkodera

U sklopu simulacija testirane su različite arhitekture autoenkodera, a izdvojene su sljedeće koje su pokazale značajnije rezultate:

	Prva arhitektura	Druga arhitektura
Ulazni sloj	(*)	(*)
Enkoder – sloj 1	64 neurona	128 neurona
Enkoder – sloj 2	64 neurona	96 neurona
Bottleneck sloj	8 neurona	32 neurona
Dekoder – sloj 1	64 neurona	96 neurona
Dekoder – sloj 2	64 neurona	128 neurona
Izlazni sloj	(*)	(*)

Tabela 2: *Arhitekture autoenkodera korišćene u simulacijama*

Prva arhitektura korišćena je i u originalnom radu. Ulazni i izlazni sloj (*) se mijenjaju zavisno od simulacije, odnosno, u zavisnosti od toga iz koliko uzastopnih *frame*-ova se koriste karakteristike, iz koliko kanala se istovremeno uzimaju karakteristike, itd. (slika 23)



Slika 23: *Prikaz toka podataka koji se prosljeđuju arhitekturama autoenkodera predstavljenim u tabeli 2*

4.5 Proces treniranja i evaluacija modela

Skupovi podataka za treniranje, validaciju i testiranje organizovani su tako da obezbijede balansiranost i reprezentativnost, osiguravajući da model ima robustne i pouzdane performanse u različitim scenarijima i uslovima rada. Svi fajlovi iz skupa podataka sa abnormalnim uslovima rezervisani su za testiranje. Kako bi testiranje bilo balansirano, jednak broj fajlova sa normalnim radnim uslovima takođe je izdvojen za testiranje. Preostali normalni fajlovi korišćeni su za treniranje modela. Ovo omogućava precizno ocjenjivanje performansi modela u realnim uslovima, sa jasno odvojenim podacima za testiranje anomalija.

Da bi se osigurala dobra generalizacija na neviđenim podacima, 10% podataka iz trening skupa odvojeno je za validaciju, dok je model treniran na preostalih 90% podataka. Ovo omogućava procjenu performansi modela na neviđenim podacima tokom svake epohe treniranja.

Optimizacija modela vršena je korišćenjem Adam optimizatora, sa podrazumijevanim parametrima (predstavljenim u poglavlju 3.6.2), koji se ističe svojom efikasnošću u konvergenciji. Poboljšanje performansi i efikasnosti treniranja postignuto je primjenom tehnika kao što su adaptivna stopa učenja, rano zaustavljanje i normalizacija *batch*-a. Adaptivna stopa učenja, korišćena u simulacijama, automatski prilagođava stopu učenja množenjem sa faktorom 0.2 nakon pet epoha bez poboljšanja performansi na validacionom skupu, s minimalnom stopom učenja postavljenom na 0.00001. Tehnika ranog zaustavljanja definiše kriterijum za prekid treniranja modela nakon što se ne zabilježi poboljšanje performansi tokom deset uzastopnih epoha, pri čemu se model vraća na težine koje su postigle najbolje rezultate. Dodatno,

normalizacija *batch*-a je primijenjena između svih susjednih slojeva enkodera i dekodera, osim nakon ulaznog i izlaznog sloja, kako bi se normalizovali izlazi iz unutrašnjih slojeva, ubrzalo učenje i postigla stabilnija distribucija podataka kroz modele.

Po završetku treniranja, model je testiran na balansiranom setu normalnih i abnormalnih podataka, kako bi se verifikovala njegova sposobnost precizne detekcije anomalija. Evaluacija modela sprovedena je korišćenjem metrike ROC AUC, koja kvantifikuje sposobnost modela da razlikuje između normalnih i abnormalnih uslova, kako je ranije opisano u radu.

5 Eksperimentalni rezultati

Eksperimentalni rezultati prikazani u ovom poglavlju demonstriraju značajna poboljšanja u performansama autoenkodera na MIMII skupu podataka u odnosu na originalne modele [1]. Modifikacije modela uključivale su primjenu adaptivne stope učenja, povećanje broja epoha treniranja, formiranje ulaza primjenom različitog broja frejmova i ekstrakciju signala iz više kanala, što je omogućilo bolju adaptaciju modela na varijacije u šumovima prisutnim u podacima. Testiranjem različitih veličina *batch*-a, odabrana je vrijednost 512, dok eksperimenti sa dropout slojevima nisu rezultovali dodatnim poboljšanjem performansi.

Rezultati iz originalnog rada prikazani su u tabeli 3, dok su poboljšani rezultati modela, iz prve simulacije u ovom radu, prikazani u tabeli 4, a dobijeni su povećanim maksimalnim brojem epoha sa 50 na 100, u kombinaciji sa tehnikom ranog zaustavljanja i primjenom adaptivne stope učenja. Ova promjena nije samo povećala preciznost modela u identifikaciji nepravilnosti, već je i spriječila preprilagođavanje, osiguravajući bolju generalizaciju na neviđenim podacima.

Rezultati u tabeli 4 dobijeni su primjenom iste arhitekture kao u referentnom radu, odnosno prve arhitekture predstavljene u tabeli 2, pri čemu su ulazni i izlazni sloj sačinjeni od po 320 neurona (5 *frame*-ova po 64 Mel koeficijenta).

VENTIL			
Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.68	0.55	0.62
02	0.66	0.59	0.57
04	0.64	0.65	0.50
06	0.70	0.66	0.53

KLIZNA ŠINA			
Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.99	0.99	0.93
02	0.93	0.79	0.74
04	0.88	0.78	0.61
06	0.71	0.56	0.52

PUMPA			
Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.84	0.65	0.58
02	0.45	0.46	0.52
04	0.99	0.95	0.93
06	0.94	0.76	0.61

VENTILATOR			
Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.75	0.63	0.57
02	0.99	0.83	0.68
04	0.92	0.75	0.57
06	0.99	0.97	0.83

Tabela 3: *Izvorni AUC rezultati za različite mašine i nivoe šuma iz referentnog rada*

VENTIL			
Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.86	0.76	0.63
02	0.77	0.68	0.59
04	0.85	0.76	0.69
06	0.80	0.72	0.57

KLIZNA ŠINA			
Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.99	0.99	0.98
02	0.96	0.87	0.79
04	0.95	0.93	0.86
06	0.92	0.81	0.64

PUMPA			
Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.88	0.75	0.71
02	0.68	0.63	0.55
04	1.0	0.99	0.95
06	0.96	0.86	0.74

VENTILATOR			
Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.84	0.60	0.57
02	0.99	0.93	0.74
04	0.94	0.84	0.57
06	0.99	0.98	0.83

Tabela 4: *AUC rezultati dobijeni u prvoj simulaciji*

U tabeli 5, prikazani su rezultati, koji su za razliku od prethodnih rezultata, dobijeni primjenom većeg maksimalnog broja epoha, tj. 300, i primjenom veće mrežne arhitekture (tabela 2, druga arhitektura), pri čemu su ulazni i izlazni sloj sačinjeni od po 576 neurona (9 *frame*-ova).

Dobijeni rezultati pokazuju znatno poboljšanje performansi u detekciji anomalija za ventile i klizne šine. Ovo poboljšanje može sugerisati na to da veća arhitektura bolje obuhvata složenost zvučnih obrazaca ovih mašina, omogućavajući efikasniju analizu suptilnih varijacija koje mogu ukazivati na probleme. S druge strane, mašine kao što su pumpe i ventilatori nisu pokazale slična poboljšanja, što može ukazivati na to da su njihovi zvučni obrasci manje osjetljivi na promjene u arhitekturi i broju *frame*-ova, potencijalno zbog manje složenih zvučnih karakteristika u odnosu na ventile i klizne šine.

VENTIL

Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.90	0.82	0.68
02	0.91	0.80	0.70
04	0.93	0.88	0.71
06	0.85	0.75	0.61

KLIZNA ŠINA

Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.99	0.99	0.99
02	0.98	0.90	0.82
04	0.99	0.99	0.94
06	0.95	0.94	0.74

PUMPA

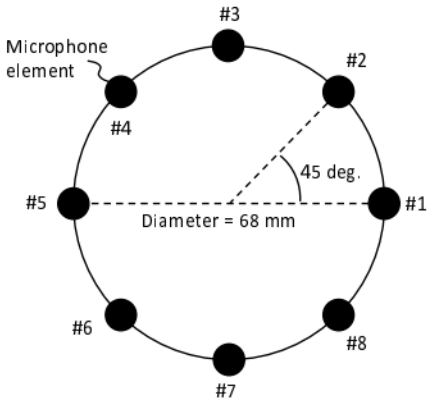
Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.90	0.61	0.66
02	0.63	0.54	0.44
04	1.0	0.99	0.95
06	0.91	0.77	0.69

VENTILATOR

Input SNR / Model ID	6 dB	0 dB	-6 dB
00	0.80	0.53	0.50
02	0.99	0.92	0.66
04	0.92	0.79	0.56
06	0.99	0.98	0.79

Tabela 5: *AUC rezultati dobijeni u drugoj simulaciji, primenjujući povećanu arhitekturu i povećan broj epoha*

U tabeli 6, prikazani su rezultati dobijeni sa maksimalnih 400 epoha treniranja, korišćenjem još veće mrežne arhitekture i podataka iz četiri audio kanala (#1, #3, #5, #7), pri čemu je svaki raspoređen pod uglom od 90 stepeni u odnosu na prethodni (slika 24). Ulazni sloj sastoji se od 1280 neurona (za svaki kanal po 5 *frame*-ova, sa po 64 Mel koeficijenta), dok su dva međusloja enkodera sačinjena od 128 i 96 neurona, a *bottleneck* sloj od 32 neurona, isto kao u prethodnom slučaju.



Slika 24: *Kružni niz mikrofona korišćen za formiranje skupa podataka [1]*

Na osnovu rezultata prikazanih u tabeli 6, može se zaključiti da korišćenje većeg broja kanala i većeg ulaznog vektora značajno poboljšava sposobnost modela da detektuje anomalije. Izbor audio kanala napravljen je na opisan način, kako bi se omogućilo modelu da uhvati širi spektar varijacija u zvučnim obrazcima, što rezultuje boljom analizom zvučnih signala. Ovo doprinosi većoj efikasnosti u detekciji anomalija, jer model može da procesuira više relevantnih informacija o akustičkom okruženju.

VENTIL			
Input SNR / Model ID	**6 dB**	**0 dB**	**-6 dB**
00	0.89	0.73	0.62
02	0.90	0.83	0.72
04	0.92	0.85	0.73
06	0.84	0.70	0.56

KLIZNA ŠINA			
Input SNR / Model ID	**6 dB**	**0 dB**	**-6 dB**
00	0.99	0.99	0.98
02	0.99	0.94	0.85
04	0.99	0.99	0.87
06	0.99	0.95	0.73

PUMPA			
Input SNR / Model ID	**6 dB**	**0 dB**	**-6 dB**
00	0.95	0.72	0.70
02	0.77	0.71	0.60
04	1.0	1.0	0.99
06	0.98	0.87	0.69

VENTILATOR			
Input SNR / Model ID	**6 dB**	**0 dB**	**-6 dB**
00	0.91	0.67	0.54
02	0.99	0.96	0.88
04	0.96	0.90	0.69
06	0.99	0.99	0.91

Tabela 6: *AUC rezultati dobijeni u trećoj simulaciji, korišćenjem četiri audio kanala*

U nastavku je dat pregled AUC rezultata, dobijenih kroz gore navedene simulacije, primijenjenih na zvučne podatke svake mašine zasebno. Simulacije su označene kao S0, S1, S2 i S3, pri čemu:

- **S0** predstavlja originalne rezultate iz rada na kojem se bazira istraživanje, dobijene kroz 50 epoha (tabela 3). Ovi rezultati služe kao referentna tačka.
- **S1** predstavlja rezultate koji su dobijeni korišćenjem iste mrežne arhitekture, ali sa povećanim brojem epoha, tj. 100, kao i primjenom gore pomenutih tehnika (tabela 4)
- **S2** predstavlja rezultate koji su, za razliku od prethodne simulacije, dobijeni primjenom veće mrežne arhitekture i većeg broja epoha, tj 300. (tabela 5)
- **S3** predstavlja rezultate koji su, za razliku od prethodne simulacije, dobijeni korišćenjem još veće mrežne arhitekture, usljed korišćenja podataka iz četiri audio kanala, uz veći broj epoha - 400 (tabela 6)

Rezultati su prikazani za tri nivoa odnosa energije signala i šuma (SNR: 6 dB, 0 dB, -6 dB), omogućavajući analizu performansi modela u različitim uslovima pozadinske buke. Svi rezultati, koji nisu referentni, su prikazani kao relativna poboljšanja (ili pogoršanja) u odnosu na S0, kako bi se jasnije istakla efikasnost svake simulacije.

VENTIL												
Input SNR / **Model ID**	**6 dB**				**0 dB**				**-6 dB**			
	S0	**S1**	**S2**	**S3**	**S0**	**S1**	**S2**	**S3**	**S0**	**S1**	**S2**	**S3**
00	**0.68**	+0.18	+0.22	+0.21	**0.55**	+0.21	+0.27	+0.18	**0.62**	+0.01	+0.06	-
02	**0.66**	+0.11	+0.25	+0.24	**0.59**	+0.09	+0.21	+0.24	**0.57**	+0.02	+0.13	+0.15
04	**0.64**	+0.21	+0.29	+0.28	**0.65**	+0.11	+0.23	+0.20	**0.50**	+0.19	+0.21	+0.23
06	**0.70**	+0.10	+0.15	+0.14	**0.66**	+0.06	+0.09	+0.04	**0.53**	+0.04	+0.08	+0.03

Tabela 7: *AUC rezultati za ventil dobijeni kroz različite simulacije i nivoe šuma (SNR)*

KLIZNA ŠINA												
Input SNR / **Model ID**	**6 dB**				**0 dB**				**-6 dB**			
	S0	**S1**	**S2**	**S3**	**S0**	**S1**	**S2**	**S3**	**S0**	**S1**	**S2**	**S3**
00	**0.99**	-	-	-	**0.99**	-.	-	-	**0.93**	+0.05	+0.06	+0.05
02	**0.93**	+0.03	+0.05	+0.06	**0.79**	+0.08	+0.11	+0.15	**0.74**	+0.05	+0.08	+0.11
04	**0.88**	+0.07	+0.11	+0.11	**0.78**	+0.15	+0.21	+0.21	**0.61**	+0.25	+0.33	+0.26
06	**0.71**	+0.21	+0.24	+0.28	**0.56**	+0.25	+0.38	+0.39	**0.52**	+0.12	+0.22	+0.21

Tabela 8: *AUC rezultati za kliznu šinu dobijeni kroz različite simulacije i nivoe šuma (SNR)*

PUMPA												
Input SNR / **Model ID**	**6 dB**				**0 dB**				**-6 dB**			
	S0	**S1**	**S2**	**S3**	**S0**	**S1**	**S2**	**S3**	**S0**	**S1**	**S2**	**S3**
00	**0.84**	+0.04	+0.06	+0.11	**0.65**	+0.10	-0.04	+0.07	**0.58**	+0.13	+0.08	+0.12
02	**0.45**	+0.23	+0.18	+0.32	**0.46**	+0.17	+0.08	+0.25	**0.52**	+0.03	-0.08	+0.08
04	**0.99**	-	-	+0.01	**0.95**	+0.04	+0.04	+0.05	**0.93**	+0.02	+0.02	+0.06
06	**0.94**	+0.02	-0.03	+0.04	**0.76**	+0.10	+0.01	+0.11	**0.61**	+0.13	+0.08	+0.08

Tabela 9: *AUC rezultati za pumpu dobijeni kroz različite simulacije i nivoe šuma (SNR)*

VENTILATOR												
Input SNR / **Model ID**	**6 dB**				**0 dB**				**-6 dB**			
	S0	**S1**	**S2**	**S3**	**S0**	**S1**	**S2**	**S3**	**S0**	**S1**	**S2**	**S3**
00	**0.75**	+0.09	+0.05	+0.16	**0.63**	-0.03	-0.10	+0.04	**0.57**	-	-0.07	-0.03
02	**0.99**	-	-	-	**0.83**	+0.10	+0.09	+0.13	**0.68**	+0.06	-0.02	+0.20
04	**0.92**	+0.02	-	+0.04	**0.75**	+0.09	+0.04	+0.15	**0.57**	-	-0.01	+0.12
06	**0.99**	-	-	-	**0.97**	+0.01	+0.01	+0.02	**0.83**	-	-0.04	+0.08

Tabela 10: *AUC rezultati za ventilator dobijeni kroz različite simulacije i nivoe šuma (SNR)*

Dakle, na osnovu tabela 7 – 10 može se vidjeti da su se rezultati značajno poboljšali u odnosu na rezultate referentne simulacije (S0). Pojedini slučajevi označeni su simbolom „-”, što ukazuje na to da je u tim scenarijima dobijen rezultat približno jednak rezultatu referentne simulacije (S0).

Dodavanje tehnika ranog zaustavljanja, normalizacije *batch*-a i adaptivne stope učenja, kao i povećanje broja epoha sa 50 na 100, u S1, već donosi značajna poboljšanja AUC vrijednosti u većini slučajeva. Dalje povećanje složenosti mreže i epoha, kao i uvođenje višekanalnog ulaza (u S3), rezultuje dodatnim poboljšanjima, koja se mogu uočiti na svim SNR (6 dB, 0 dB, –6 dB), iako u manjem broju slučaja rezultati neke simulacije nisu bolje od odgovarajućih rezultata iz prethodne simulacije, ili su poboljšanja marginalnija kada su performanse prethodne simulacije već visoke.

U slučaju SNR = 6 dB, modeli su u S1 već postigli relativno dobre AUC vrijednosti. Iako prelazak na S2 i S3 uglavnom donosi dodatna poboljšanja, povećanja su manja nego što su bila između S0 i S1, jer su performanse već dosegle visok nivo u S1.

Iako nivo SNR = -6 dB predstavlja najizazovniji scenario za detekciju, pojedini modeli pokazali su značajna poboljšanja rezultata i u ovim uslovima. To ukazuje na to da su modeli postali robusniji prema bučnim ulaznim podacima u odnosu na ranije, jednostavnije verzije

modela. Složeniji modeli, S2 i S3, pokazuju izuzetnu otpornost na visoke nivoe šuma – jednu od ključnih karakteristika za primjenu u realnim scenarijima. Posebno kod kliznih šina i ventila, pri SNR od –6 dB, naprednije arhitekture dostižu veće AUC vrijednosti, značajno nadmašujući tako performanse jednostavnijih modela u ekstremno bučnim uslovima rada. Ovo ukazuje na to da optimizovana arhitektura modela i tehnike obrade signala mogu efikasno da izdvoje relevantne informacije iz signala čak i u prisustvu znatnog šuma, što znači da su u stanju da „nauče" šablone neophodne za pouzdanu detekciju i u vrlo nepovoljnim akustičkim uslovima.

6 Zaključak

Precizna detekcija anomalija u industrijskim mašinama od ključnog je značaja za osiguranje pouzdanosti, efikasnosti i sigurnosti u savremenim industrijskim okruženjima. Ovaj rad istražuje primjenu autoenkodera za detekciju anomalija koristeći MIMII skup podataka, fokusirajući se na četiri vrste mašina: ventile, pumpe, ventilatore i klizne šine. Kroz modifikacije arhitekture modela i optimizaciju parametara treniranja, poput povećanja broja epoha, dinamičkog podešavanja stope učenja, primjene tehnike ranog zaustavljanja, varijacije broja *frame*-ova koji se koriste na ulazu, kao i korišćenja više audio kanala, postignuta su značajna poboljšanja performansi modela.

Rezultati istraživanja ukazuju na značajan potencijal autoenkodera u detekciji anomalija, pri čemu su veće arhitekture mreža i dodatni audio kanali omogućili bolju detekciju, posebno u slučaju kliznih šina, gdje je postignuto poboljšanje performansi do 70% u odnosu na referentne rezultate (za model 06 pri SNR = 0 dB). Štaviše, složenije arhitekture (S2 i S3) pokazale su izuzetnu otpornost na visoke nivoe šuma, uključujući izuzetno nepovoljne uslove SNR = -6 dB. Kod kliznih šina i ventila, ovi modeli su značajno nadmašili performanse jednostavnijih verzija, što ukazuje na njihovu sposobnost da izdvoje relevantne šablone iz signala čak i u uslovima visokog nivoa šuma. Ovakvi rezultati sugerišu da optimizovana arhitektura modela može omogućiti pouzdanu detekciju u realnim industrijskim scenarijima.

Međutim, metodološko ograničenje može biti u samom pristupu korišćenja autoenkodera. Iako su autoenkoderi moćni u detekciji anomalija, oni zavise od kvaliteta i količine trening podataka. Nedostatak raznovrsnosti u trening podacima može dovesti do toga da model ne prepoznaje sve vrste anomalija.

Dalji pravci u istraživanju uključuju testiranje modela na drugim skupovima podataka ili u realnim industrijskim okruženjima kako bi se potvrdila njegova sposobnost generalizacije i prilagodljivost u različitim uslovima. Buduća istraživanja takođe mogu obuhvatiti integraciju više izvora podataka, poput vibracionih i temperaturnih podataka, koji bi mogli doprinijeti poboljšanju tačnosti modela. Na primjer, kombinacija akustičkih signala s vibracionim podacima

mogla bi omogućiti modelima da identifikuju širi spektar anomalija, poboljšavajući njihovu ukupnu efikasnost.

Osim toga, optimizacija memorijskih i računarskih zahtjeva modela mogla bi omogućiti njihovu implementaciju u realnom vremenu i na industrijskim uređajima, što bi dovelo do bržih i efikasnijih odgovora na potencijalne probleme, smanjujući rizike i troškove povezane s kvarovima mašina.

Ovim istraživanjem ne samo da su unaprijeđene trenutne performanse modela, već je pružen značajan doprinos razumijevanju uticaja arhitekture i konfiguracije ulaznih podataka na sposobnost detekcije anomalija. Predstavljeni zaključci i smjernice otvaraju mogućnosti za dalja istraživanja i praktičnu primjenu ovih metoda u realnim industrijskim okruženjima.

Literatura

[1] H. Purohit, R. Tanabe, K. Ichige, T. Endo, Y. Nikaido, K. Suefusa, and Y. Kawaguchi, "MIMII dataset: Sound dataset for malfunctioning industrial machine investigation and inspection," *arXiv preprint* arXiv:1909.09347, Sep. 2019. [Online]. Available: https://arxiv.org/abs/1909.09347

[2] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950, doi: 10.1093/mind/LIX.236.433.

[3] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM Journal of Research and Development*, vol. 44, pp. 206–226, 1959, doi: 10.1147/rd.441.0206.

[4] A. Burkov, *The Hundred-Page Machine Learning Book*. [S.l.]: Andriy Burkov, 2019.

[5] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York: Springer, 2009.

[6] I. H. Sarker, *Machine Learning: Algorithms, Real World Applications and Research Directions*. Singapore: Springer Nature Singapore Pte Ltd., 2021.

[7] DataCamp, "Normalization vs Standardization," [Online]. Available: https://www.datacamp.com/tutorial/normalization-vs-standardization

[8] T.-T. Wong, "Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation," *Pattern Recognition*, vol. 48, no. 9, pp. 2839–2846, Sep. 2015, doi: 10.1016/j.patcog.2015.03.009.

[9] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*. New York: Springer, 2013, ISBN: 978-1461471370.

[10] M. Hossin and M. N. Sulaiman, "A review on evaluation metrics for data classification evaluations," *International Journal of Data Mining & Knowledge Management Process*, vol. 5, no. 2, pp. 1–11, 2015, doi: 10.5121/ijdkp.2015.5201.

[11] T. Fawcett, "ROC graphs: Notes and practical considerations for researchers," 2004.

[12] Y. Sasaki, *The Truth of the F-Measure*. School of Computer Science, University of Manchester, Oct. 2007.

[13] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997, doi: 10.1016/S0031-3203(96)00142-2.

[14] T. Fawcett, "Introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006, doi: 10.1016/j.patrec.2005.10.010.

[15] D. J. Hand and R. J. Till, "A simple generalisation of the area under the ROC curve for multiple class classification problems," *Machine Learning*, vol. 45, no. 2, pp. 171–186, 2001.

[16] MathWorks, "What Is Overfitting?," [Online]. Available: https://www.mathworks.com/discovery/overfitting.html

[17] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015. [Online]. Available: http://neuralnetworksanddeeplearning.com/

[18] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986, doi: 10.1038/323533a0.

[19] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Proceedings of the International 1989 Joint Conference on Neural Networks*, 1989.

[20] J. Zhang, "Gradient Descent based Optimization Algorithms for Deep Learning Models Training," Information Fusion and Mining Laboratory, Tech. Rep., Feb. 2019; revised Mar. 2019.

[21] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, May 2015.

[22] R. Grosse, "Lecture 15: Exploding and vanishing gradients," University of Toronto Computer Science, 2017.

[23] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed. O'Reilly Media, 2019.

[24] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[26] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France, 2015, pp. 448–456.

[27] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," *arXiv preprint* arXiv:2003.05991, 2020. [Online]. Available: https://arxiv.org/abs/2003.05991

[28] U. Michelucci, "An introduction to autoencoders," *arXiv preprint* arXiv:2201.03898, Jan. 2022. [Online]. Available: https://arxiv.org/abs/2201.03898

[29] C. C. Aggarwal, *Neural Networks and Deep Learning: A Textbook*. Cham: Springer, 2018, doi: 10.1007/978-3-319-94463-0.

[30] K. Berahmand, F. Daneshfar, E. S. Salehi, Y. Li, and Y. Xu, "Autoencoders and their applications in machine learning: A survey," *Artificial Intelligence Review*, vol. 57, no. 28, 2024, doi: 10.1007/s10462-023-10662-6.

[31] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, Banff, AB, Canada, Apr. 2014.

[32] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 833–840. [Online]. Available: https://icml.cc/2011/papers/455_icmlpaper.pdf

[33] B. Janakiramaiah, G. Kalyani, S. Narayana, and T. B. M. Krishna, "Reducing dimensionality of data using autoencoders," in *Smart Intelligent Computing and Applications*, S. Satapathy, V. Bhateja, J. Mohanty, and S. Udgata, Eds. Singapore: Springer, 2020, vol. 160, pp. 57–66

[34] S. Ahmad, K. Styp-Rekowski, S. Nedelkoski, and O. Kao, "Autoencoder-based condition monitoring and anomaly detection method for rotating machines," *arXiv preprint* arXiv:2101.11539, 2021. [Online]. Available: https://ar5iv.org/abs/2101.11539

[35] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language*

Models, 3rd ed., draft of January 12, 2025, [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/

[36] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357–366, Aug. 1980, doi: 10.1109/TASSP.1980.1163142.