SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

SEMINAR

Duboko učenje

Leo Pleše

Voditelj: *Siniša Šegvić*

Zagreb, svibanj, 2019.

Sadržaj

1. Uvod

Čovječanstvo je oduvijek tragalo za odgovorom što je bit ljudske inteligencije. Kako bismo se približili odgovoru na to pitanje, rodila se ideja da čovjek stvori dio te inteligencije. U tom slučaju, takvu bismo inteligenciju nazvali umjetnom inteligencijom nasuprot čovjekovoj prirodnoj inteligenciji.

Može se reći da je polje umjetne inteligencije danas zasebna, štoviše jedna od ključnih, grana znanosti i istraživačke djelatnosti sa širokom primjenom u samoj znanosti, ali i u svakodnevnom životu. U tom smislu, dovoljno je samo spomenuti npr. različite sustave za detekciju objekata na slikama ugrađene u kamerama, inteligentne strojeve koji donose različite odluke poput strojeva u medicini ili pak autonomna vozila – svi oni dijele jedno: umjetnu inteligenciju.

Već sada je vidljivo da bit umjetne inteligencije nije samo automatizacija npr. industrijske proizvodnje ili u bilo kojem smislu pomoći čovjeku ne bi li se neki rad obavio brže i kvalitetnije, već je to i sposobnost donošenja odluka. Primjerice, kod prepoznavanja slika to je donošenje konačne odluke o prepoznatom objektu, u medicinskoj primjeni donošenje dijagnoza bolesti pa sve do autonomnih vozila gdje je riječ i o prepoznavanju objekata i ljudi te pravovremenom obavljanju svih radnji. Za obavljanje tih inteligentnih radnji važne su dvije glavne stvari: baza znanja i sposobnost učenja.

Baza znanja jest temelj na kojemu se donosi svaka odluka pa je ona neizbježna. Inteligentni stroj najprije prikuplja informacije iz vanjskog svijeta kako bi upoznao situaciju u kojoj se nalazi i na temelju nje dalje reagirao. Nadalje, s vremenom treba naučiti kako reagirati na određenu situaciju za što treba sposobnost učenja. Specifično svojstvo učenja iz stečenog znanja zasnovano na prepoznavanju uzoraka u primljenim podacima naziva se strojnim učenjem.

Strojno učenje je također zasebno znanstveno polje koje proučava algoritme učenja temeljene na matematičkim modelima koji proizlaze iz polja vjerojatnosti i statistike. Strojno učenje ide i korak dalje od ekstrakcije uzoraka – prema grupiranju uzoraka u skupove uzoraka kojima se onda dodijeljuje određena reprezentacija. Taj je korak ključan prijelaz od strojnog prema dubokom učenju i naziva se reprezentacijskim učenjem. Međutim, taj je postupak stvaranja reprezentacije vrlo apstraktan zbog utjecaja varijacijskih faktora. Npr. na prepoznavanje objekta na slici mogu utjecati različiti vanjski vremenski uvjeti poput količine svjetlosti ili sam objekt može biti u nekom od svojih svojstava drugačiji od nekog drugog prije promatranog koji je prepoznat kao objekt iste kategorije. Upravo ovdje dolazi polje dubokog učenja koje pokušava razriješiti problem kompleksnosti apstrahiranja uzoraka u reprezentaciju objekta.

Duboko učenje je duboko jer kombinira više koncepata odnosno kriterija na temelju kojih sustavnim putem donosi odluku. Te koncepte primjenjuje u više razina gdje je svaka sljedeća razina sposobna prepoznati sve kompleksnije uzorke te na kraju u zadnjoj razini dobiti konačan odgovor tj. odluku. Na kraju, dubina dubokog učenja je u količini računanja potrebnog da se iz ulaza dobije izlaz unutar nekog modela te u

dubini grafa koji opisuje relacije među promatranim konceptima reprezentiranim u pojedinom slučaju.

2. Paradigme strojnog učenja

Danas prevladavaju tri glavna pristupa strojnom učenju: nadzirano (eng. supervised), nenadzirano (eng. unsupervised) i podržano (eng. reinforcement) učenje. Ta podjela zasniva se na tipu podataka i načinu na koji rade s ulaznim i izlaznim podacima.

2.1. Nadzirano učenje

Nadzirano učenje je nadzirano jer se temelji na poznatim izlaznim podacima odnosno podacima koji su već obilježeni da pripadaju određenoj kategoriji (eng. labeled data). Zahvaljujući tim obilježenim podacima, nadzirani model može stvoriti odnose između svojstava ulaznih podataka i kategorija kojima su oni obilježeni.

U nadziranom modelu posebno je važna podjela podataka na skup za učenje (eng. training set) i skup za testiranje (eng. testing set). Skup za učenje dio je raspoloživih podataka na temelju kojeg stroj uči, dok skup za testiranje je skup ostalih podataka različitih od onih na kojima je stroj učio koji se koristi kako bi se provjerila učinkovitost učenja odnosno učinkovitost stroja u zadanom zadatku.

Neke od najvažnijih primjena nadziranog modela učenja jesu regresija (eng. regression) i klasifikacija (eng. classification).

2.1.1. Regresija

Regresija kao nadzirani model svodi se na aproksimaciju funkcije koja mapira podatke iz promatranog ulaznog skupa na njihove izlazne vrijednosti. Jednom kad je model regresije formiran odnosno kada je funkcija određena, može se koristiti za predviđanje vrijednosti izlaza za nove neviđene ulazne podatke. Jednostavnije varijante regresije prikazuju odnos odnosno stvaraju funkciju ovisnosti dvaju određenih svojstava.

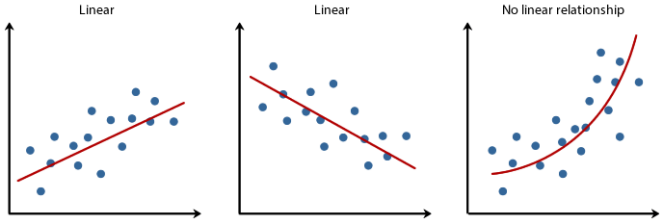
Dva su glavna tipa regresije – linearna i polinomna. Linearna regresija koristi linearnu funkciju kao funkciju aproksimacije, dok polinomna koristi za to polinomne funkcije višeg reda. Ilustracija obje vrste regresije je na slici 2.1.

2.1.1.1. Linearna regresija

Linearna regresija svodi se na pronalaženje linearne funkcije kojom se opisuje odnos između dvaju svojstava. Kao jedna od jednostavnijih tehnika koju pritom može koristiti je metodu najmanjih kvadrata (eng. OLS – Ordinary Least Squares) s ciljem minimizacije kvadratne pogreške između svake promatrane točke i točke na pravcu određenom linearnom funkcijom odnosno minimizacije varijance. Ta se tehnika također može protumačiti i kao pronalaženje funkcije gdje će vjerojatnost da se vrijednost promatranog podatka odgovara aproksimiranoj biti maksimalna, stoga se naziva i tehnikom procjenjivanja najveće izglednosti (eng MLE – Maximum Likelihood Estimation).

2.1.1.2 Polinomna regresija

Polinomna regresija svodi se na pronalaženje polinomne funkcije n-tog reda kojom se opisuje odnos između dvaju svojstava. Pritom je linearna regresija zapravo podslučaj polinomne za polinom prvog stupnja (n = 1).
U odabiranju polinoma koji najbolje opisuje odnos dvaju danih svojstava, važno je najprije naći polinom odgovarajućeg stupnja. Naime, ako je polinom preniskog stupnja, može doći do podnaučenosti (eng. underfitting) što znači nedovoljno dobro treniran model. S druge strane, ako je odabran polinom previsokog stupnja, javlja se problem prenaučenosti (eng. overfitting). To nikako nisu problemi specifični isključivo za polinomnu regresiju pa će o njima biti više riječi, no kada govorimo o polinomnoj regresiji, ti se problemi izravno manifestiraju odabirom neodgovarajućeg stupnja polinomne funkcije.
Također, uz odabir stupnja polinoma, važno je pažljivo odabrati i koeficijente/težine (eng. coefficients/weights) polinoma. Ti koeficijenti koji opisuju dana svojstva najčešće su linearni iako je sama funkcija bilo koja polinomna funkcija.



Slika 2.1: Linearna (lijevo i sredina) i polinomijalna regresija (desno)

2.1.2. Klasifikacija

Klasifikacija je tip nadziranog modela koji iz promatranih vrijednosti izvlači zaključke u vidu njihove pripadnosti određenoj kategoriji. Kategorija kod nadziranog učenja je izlaz modela koji znači oznaku koja će se dodijeliti podatku. Podaci iz skupa za učenje se označavaju čime se oni od tada mogu koristiti kao odgovori na kojih će se model dalje trenirati.

Mnoštvo je modela klasifikacije, a jedan od osnovnih je tzv. naivni Bayes.

2.1.2.1. Naivni Bayes

Model Bayesovog klasifikatora temelji se na uvjetnoj vjerojatnosti (Bayesovom teoremu). U kontekstu nadziranog učenja, ukoliko su poznata određena svojstva promatranog podatka, naivni Bayes predviđa kolika je vjerojatnost da će taj podatak biti klasificiran kao pripadnik jedne od određenih kategorija. Naziv „naivni" metoda je dobila jer ne uzima u obzir međuzavisnost pojedinih faktora koji utječu na konačnu odluku klasifikacije.

Tipičan primjer primjene ove metode je klasifikacija e-mail poruka na „spam" i „ham" („non-spam") poruke na temelju pojave određenih ključnih riječi koje signaliziraju da se radi o jednoj odnosno drugoj vrsti poruke. U tom smislu, prema Bayesovom teormu bi vrijedilo da je vjerojatnost da je poruka „spam" ukoliko sadrži odabranu riječ bila jednaka umnošku vjerojatnosti da je poruka „spam" i vjerojatnosti da poruka sadrži odabranu riječ ako se zna da je „spam" podijeljenom s vjerojatnosti da poruka sadrži tu riječ. Pritom se sama vjerojatnost da poruka sadrži odabranu riječ računa kao potpuna vjerojatnost. Formula je dana na slici 2.2.

$$p(spam|riječ) = \frac{p(spam)p(riječ|spam)}{p(riječ)}$$

Slika 2.2: Bayesov teorem na primjeru detekcije spam poruka s određenom riječi

Rezultat naivnog Bayesa je kategorija koja ima na kraju najveću uvjetnu vjerojatnost i koja se onda pridaje kao klasa promatranog podatka (procjena zvana maksimum aposterirori vjerojatnosti – eng. Maximum A Posteriori Probability). Za primjer e-mail poruka, ta je odluka očito binarna i daje onu od dvije klase koja ima veću vjerojatnost.

2.2. Nenadzirano učenje

Nenadzirano učenje, nasuprot nadziranom, je nenadzirano jer inicijalno stroj nema odgovora iz kojih bi učio već sam mora moći doći do razumijevanja podataka zasnivajući to isključivo na promatranjima. Kroz promatranja stroj počinje prepoznavati sličnosti među podacima.

Stoga, važna primjena nenadziranog učenja je grupiranje (eng. clustering). Ideja grupiranja je grupiranje podataka prema sličnosti. Pritom stroj sam određuje kriterije prema kojima vrši grupiranje i na temelju njih obilježava podatke. Upravo zbog toga jedna od najvažnijih primjena je u pronalaženju tzv. latentnih varijabli što su varijable koje predstavljaju određena svojstva ulaznih podataka za koje prije nije bilo poznato da postoje, a model nenadziranog učenja može ih uspješno razotkriti. Ta svojstva mogu u nekim slučajevima biti čak i važnija od onih za koje se prvotno smatralo da imaju glavni utjecaj.

Jedna od značajnijih tehnika grupiranja je algoritam K-srednjih vrijednosti.

2.2.1. K-means clustering

Algoritam K-srednjih vrijednosti (eng. K-means clustering) dijeli podatke u k-grupa (eng. cluster) podataka na temelju pronađene sličnosti uzoraka. Pritom se definiraju točke centroida koje predstavljaju centre svake od k-grupa. Ostale točke iz promatranog skupa podataka grupiraju se u onu od k-grupa čijem su centroidu najbliže. Dakle, model K-srednjih vrijednosti sam po sebi konvergira takvoj podjeli zasnivajući se na svojstvima samih podataka. Ilustracija takve podjele je na slici 2.3.

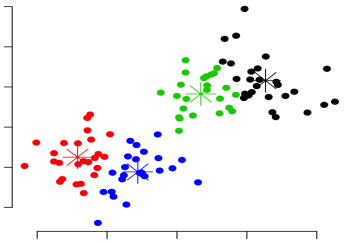
Sam algoritam nalaženja centroida konceptualno radi na sljedeći način. Najprije slučajno odabere K centroida i zatim svakoj od točaka podataka pridijeli grupu čijem je centroidu najbliža. Nakon toga računa se novi položaj centroida na temelju prosjeka točaka svake grupe. Cijeli postupak se ponavlja sve dok sve točke ne budu konstantno pridijeljivane istom centroidu. Na kraju, za tako dobiven model može se za bilo koju novu točku (podatak) odrediti kojoj od K grupa pripada.

U procesu grupiranja važno je odabrati prihvatljiv broj K. Taj broj nikako ne smije biti premalen jer se konvergencija neće postići, a dovoljno je da bude određene vrijednosti iznad koje se kvadratna pogreška stabilizira odnosno kada se udaljenosti svake točke od centroida stabiliziraju.

Također, važna je i inicijalna pozicija centroida. U tom smislu, zbog slučajnog izbora položaja centroida može doći do problema lokalnih minimuma koji sprječavaju sam

algoritam da dođe do ispravne konačne podjele u grupe. Radi izbjegavanja tog problema, potrebno je provesti algoritam više puta.

Na kraju, važno je napomenuti da jedan od glavnih izazova ovog modela jest opisati značenje grupa pronađenih algoritmom. Dakle, algoritam provodi samo grupiranje, a nakon toga je potrebno provesti analizu značenja.



Slika 2.3: Ilustracija K-means algoritma za k = 4

2.3. Podržano učenje

Podržano učenje model je učenja, ali i zasebno polje unutar strojnog učenja koje se bavi pitanjem maksimizacije sume nagrada u pojedinoj situaciji. Situacija u kojoj se agent nalazi određena je njegovim stanjem i skupom akcija koje može poduzeti kako bi prešao u sljedeće stanje. Na svom putu od početka do cilja za svako stanje agent istražuje stanja kojima može prolaziti i akcije koje pritom može poduzeti kako bi na tom putu maksimizirao nagradu.

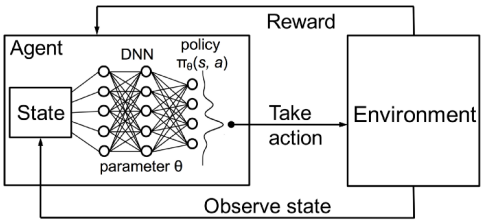
Jedna od značajnih implementacija podržanog učenja je model Q-učenja.

2.3.1. Q-učenje

Q-učenje (eng. Q-learning) određeno je skupom stanja okruženja (s), skupom mogućih akcija u tim stanjima (a) i vrijednostima za svaki par stanje-akcija (Q-vrijednosti) koje su inicijalno postavljene na 0. Akcije u svakom stanju mogu dati nagradu koja pozitivna, negativna ili jednaka nuli. Ako je npr. za dano stanje-akcija Q-vrijednost pozitivna, to može povećati Q-vrijednost prošlih stanja-akcija kojima je agent već prošao. Pri izračunu Q-vrijednosti u obzir se uzima i faktor koji uzima u obzir udaljenost dvaju promatranih stanja. Ilustracija ovog algoritma je na slici 2.4.

Problem Q-učenja je istraživanje svih mogućih stanja. Na prvi pogled, naivni pristup bi bio za svako stanje odabrati akciju koja daje najveću Q-vrijednost, no na taj način cijeli prostor mogućnosti ne bi bio istražen. Stoga se uvodi tzv. epsilon vrijednost vjerojatnosti: ako je slučajno odabrani broj manji od epsilon, akcija se odabire slučajno.

Q-učenje se može opisati i tzv. Markovljevim procesom odlučivanja koji upravo definira stanja (trenutno i prošlo), funkcije prijelaze iz prethodnog u trenutno stanje danom akcijom te funkciju nagrade (ekvivalent Q-vrijednosti) za prijelaz pojedinom akcijom iz prethodnog u trenutno stanje. Također, često se upotrebljava i pojam dinamičkog programiranja u kontekstu Q-učenja koje podrazumijeva rješavanje problema na način da se rješavaju i zatim pamte rješenja potproblema kako bi se ubuduće isti problem mogao riješio koristeći prije dobivene spoznaje o poznatim mogućnostima prijelaza iz jednog u drugo stanje.



Slika 2.4: Ilustracija algoritma Q-učenje

3. Duboko učenje

3.1. Funkcije gubitka

Funkcije gubitka (eng. loss/cost function) u dubokom učenju imaju važnu ulogu u problemima predviđanja gdje je bitna procjena kvalitete modela podataka odnosno procjene točnosti s kojom algoritam predviđa vrijednosti iz danog skupa podataka. Stoga se one koriste, prije svega, kao indikator učinkovitosti modela pri radu na poboljšanju danog algoritma. Ovdje će se opisati dvije osnovne funkcije gubitka (regresije i klasifikacije).

3.1.1. Kvadratna pogreška

Kvadratna pogreška (eng. MSE – Mean Square Error, L2 Loss) jedna je osnovnih funkcija gubitka regresije koja daje prosjek kvadratnih pogreški odnosno odstupanja predviđenih od stvarnih vrijednosti. Ona mjeri očekivanje promatrane slučajne varijable uzimajući isključivo kvadrat iznosa pogreška, ali ne i smjera (formula na slici 3.1.). Budući da se radi o kvadratu, predviđanja dalje od stvarnih vrijednosti (iznimke – eng. outliers) imaju znatno veću kvadratnu pogrešku odnosno ta predviđanja glavnim dijelom doprinose ukupnoj kvadratnoj pogreški. Slično, kvadratnu pogreška također karakterizira „stabilan“ gradijent koji se postupno povećava idući od minimuma funkcije gubitka dalje u bilo koju stranu čime ukazuje da se pogreška povećava na navedeni način. To je upravo npr. razlika između MSE i MAE (apsolutna pogreška – eng. Mean Absolute Error, L1 Loss, definirana kao prosjek apsolutne vrijednosti razlika predviđene i stvarne vrijednosti) jer ova druga ima gradijent jednak za sve vrijednosti manje, odnosno sve veće od minimuma.

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \bar{y_i})^2}{n}$$

Slika 3.1: Formula za izračun MSE

3.1.2. Negativna log-izglednost

Negativna log-izglednost (eng. NLL - Negative Log Likelihood) jedna je osnovnih funkcija gubitaka klasifikacije. Mjera je odstupanja predviđene vjerojatnosti od stvarne oznake kategorije u koju se stvarna vrijednost klasificira odnosno mjera točnosti modela klasifikatora koji daje vjerojatnost za svaku klasu (formula na slici 3.2.). Može se reći da ova funkcija uključuje ideju pouzdanosti vjerojatnosti i, u tom smislu, ova funkcija penalizira veliku pouzdanost u vrijednost daleko od stvarne vrijednosti. Prema tome, cilj koji se ovom funkcijom može postići jest dobiti relativno veliku vjerojatnost (log-izglednost) točne klasifikacije podataka s relativno malom pogreškom. Ovdje leži i razlog zašto je ova log funkcija negativna – radi interpretacije: kako bi veća predviđena vjerojatnost imala manji log gubitak. Iz izraza za log-izglednost vidljivo je da osigurano i sljedeće: za vrijednost jednaku jedan samo je lijevi dio izraza je različit od nule, a za stvarnu vrijednosti jednaku nula samo desni dio. Na taj način osigurano je da funkcija radi isključivo s predviđenom vjerojatnosti za točnu klasu.

$$NLL = -(y_i \log(\overline{y_i}) + (1 - y_i)\log(1 - \overline{y_i}))$$

Slika 3.2: Formula za izračun NLL

3.2. Stohastički gradijentni spust

Gradijentni spust (eng. GD – Gradient Descent) metoda je izračuna gradijenta ciljane funkcije s ciljem pronalaženja optimalnog postava parametara za dani problem pritom koristeći neku od funkcija gubitka. Konceptualno radi na sljedeći način: najprije se slučajno odabere vrijednost za koju se mjeri odstupanje od stvarne vrijednosti, zatim se izračuna pogreška koristeći danu funkciju gubitka, te zatim prijeđe na sljedeću vrijednost. Ukoliko sljedeća vrijednost ima manje odstupanje, postupak se ponavlja sve dok algoritam ne dođe do vrijednosti koja ima pogrešku veću od pogreške u prethodnoj točci, što znači da je algoritam došao do minimuma pogreške što i jest bio cilj gradijentnog spusta. Naziv „stohastički“ ova je implementacija gradijentog spusta dobila zbog slučajnog odabira uzoraka iz skupa za učenje i slučajnog poretka uzoraka vrijednosti u uzetom skupu.

Jedan od glavnih problema ove metode jest problem lokalnih minimuma (eng. local minima problem). Kako je odabira inicijalne pozicije na krivulji pogreške slučajan, funkcija može doći do minimuma idući u jednom smjeru, ali to ne treba nužno biti globalni, već samo lokalni minimum dane funkcije. U tom slučaju, optimalna točka minimalne pogreška nikad ne bi bila pronađena čime cilj stohastičkog gradijentnog spusta kao optimizacijske funkcije ne bi bio ispunjen, već bi pronađena greška bila u toj suboptimalnoj ravnotežnoj točki.

Naposljetku, korisno je napomenuti da postoje različite tehnike ubrzavanja samog algoritma SGD koje se zasnivaju na brzom i učinkovitom računanju gradijenta. Jedna od ključnih tehnika je automatsko diferenciranje (eng. AD - automatic differentiation). Primjerice kod neuronskih mreža se koristi mod reverznog diferenciranja što znači da se prolazak kroz mrežu vrši od kraja prema početku što je općenito bolji odabir od prolaska od početka prema kraju. Razlog tomu je činjenica da većina neuronskih mreža ima znatno manje izlaza od ulaza, stoga broj parcijalnih derivacija koje treba izračunati okvirno odgovara broju izlaza iz mreže. Pri tom izračunu se koristi lančano pravilo u kojemu se redom množe parcijalne derivacije ovisnosti trenutnog izlaza o prethodnim ulazima sve dok se od izlaznog sloja ne dođe do ulaznog sloja neurona. To će biti ilustrirano kod jedne od najvažnijih primjena ove metode – algoritma propagacije greške unatrag.

3.3. Potpuno povezan model

Potpuno povezan model (perceptron) model je duboke neuronske mreže sačinjen od niza potpuno povezanih slojeva neurona. Osnova na kojoj je nastao bili su linearni diskriminativni modeli (eng. LTU – Linear Threshold Unit) koje najprije treba promotriti.

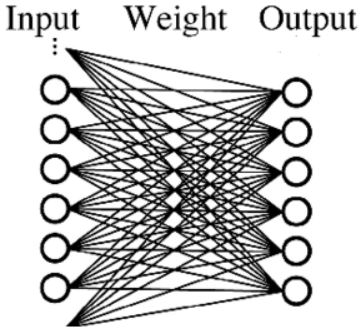
Linearni diskriminativni modeli zasnivaju se na biološkom znanju o neuronima. Prva činjenica je da svaki ulaz ima određenu težinu što je analogija jačini neuronske veze. Stoga se izlaz modelira kao suma produkata ulaza i odgovarajućih težina. Izlaz je

pritom ustvari definiran pomoću step-funkcije kao funkcije aktivacije (objašnjene u 3.5.) ovog modela koja se na kraju primijeni na dobivenu sumu: izlaz je 1 ako je dobivena suma veća ili jednaka nuli, odnosno 0 ako je manja. Time je dobiven relativno jednostavan, ali realističan model biološkog neurona.

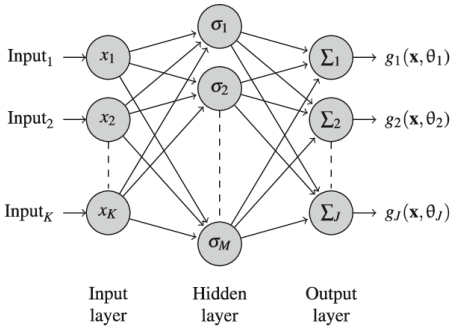
Sada se može uočiti: povezivanjem više linearnih diskriminativnih modela svim mogućim vezama u jedan sloj dobiva se potpuno povezan sloj (skica modela na slici 3.3.) modela neurona čiji izlaz može biti ulaz za sljedeći potpuno povezan sloj i tako tvoriti potpuno povezan model. Takav je sloj odnosno model važan jer je on taj koji omogućuje učenje.

Glavni izazov ovakvog modela je namještavanje težina (koeficijenata funkcije) kojima se množe ulazi svakog sloja s ciljem optimizacije učinkovitosti modela.

Nadalje, potpuno povezan model, kako je spomenuto, može imati više potpuno povezanih slojeva čime se dobiva višerazinski potpuno povezan model/perceptron (eng. MLP – multi-layer perceptron) odnosno duboka unaprijedna neuronska mreža (eng. deep feedforward neural network) koja osim sloja ulaza i sloja izlaza ima i jedan ili više tzv. skrivenih slojeva (skica na slici 3.4.). Upravo ti skriveni slojevi omogućuju ekstrakciju kompleksnijih svojstava promatranih podataka. Na primjeru ekstrakciju svojstava slike: svaki sljedeći skriveni sloj višerazinskog potpuno povezanog modela omogućuje prepoznavanje sve kompleksnijih uzoraka iz slike – prvi sloj može npr. prepoznavati samo rubove, sljedeći već geometrijske uzorke sa slike, a neki od zadnjih čak i objekte iz stvarnog svijeta sadržane na slici. Dakle, ovakve neuronske mreže koje sadrže skrivene slojeve nazivamo dubokima. Broj slojeva u tom ulančanom nizu potpuno povezanih slojeva predstavlja dubinu takve mreže, dok broj paralelnih jedinica unutar jednog sloja njenu širinu.



Slika 3.3: Ilustracija potpuno povezanog modela



Slika 3.4: Ilustracija MLP modela s jednim skrivenim slojem

3.4. Algoritam propagacije greške unatrag

Algoritam propagacije greške unatrag (eng. backpropagation) matematički je alat primjenjiv za računanje gradijenta funkcije skalarnih argumenata čija je jedna od važnih primjena već spomenuta tehnika stohastičkog gradijentnog spusta koji ovaj algoritam koristi za minimizaciju greške dane funkcijom aktivacije u svrhu učenja modela. Općenitije, ovaj algoritam radi ne samo za višerazinske potpuno povezane modele, već i druge funkcije (za koje je derivacija definirana).

Usmjerivši se na MLP modele, konkretan zadatak dan ovom algoritmu na rješavanje je računanje gradijenta funkcije gubitka s obzirom na njene parametre. Pritom se koristi lančano pravilo za računanje derivacije kompozitne funkcije kao umnoška parcijalnih derivacija funkcije redom kompozicije.

Konceptualno, algoritam radi na sljedeći način: najprije izračuna grešku za trenutni čvor, zatim izračuna doprinos svakog neurona iz prijašnjeg skrivenog sloja toj greški, te propagira tu pogrešku unatrag pri reverznom prolazu, te na kraju prilagodi težinu radi smanjenja pogreške.

Najveća prednost ovog algoritma leži u činjenici da pojedine podizraze umnožaka parcijalnih derivacija neće trebati računati više puta zahvaljujući rekurzivnoj prirodi problema. Kada ne bi bilo tako, kompleksnost izračuna gradijenta bila bi eksponencijalnog reda. S druge strane, ako je memorija ograničena, onda je nužno višestruko računanje pojedinih podizraza.

3.5. Funkcije aktivacije

Aktivacijske funkcije su funkcije transformacije sume izlaza iz čvora neuronske mreže. Aktivacija je definirana kao suma skalarnog produkta težina i ulaza te pristranosti. Pritom je potrebno definirati pristranost (eng. bias) – to je izraz koji se dodaje i koji utječe isključivo na izlazne, ali ne i na ulazne vrijednosti, na način da pomiče funkcije lijevo ili desno kako bi bolje pristajala podacima. Može se uspostaviti analogija s pragom perceptrona gdje je pristranost bila vrijednost potrebna da bi izlaz bio 1, dok je ovdje u neuronskim mrežama ta vrijednost važna radi prilagođavanja prema ulazu.

Jedna od takvih funkcija je spomenuta step-funkcija kao primjer jednostavne funkcije koja služi za dobivanje konačnog izlaza 1 odnosno 0 za vrijednost težinske sume veće odnosno manje od nule redom. Međutim, problem sa step-funkcijom je što račun gradijenta nije koristan, te pritom ima problem u nuli gdje je derivacija nedefinirana. Stoga ćemo se osvrnuti na dvije važne aktivacijske funkcije – ReLU i softmax (na primjeru sigmoidalne) – prikazane na slici 3.6.

3.5.1. ReLU funkcija

Relu funkcija (eng. rectified linear unit) odnosno funkcija jedinične rampe aktivacijska je funkcija koja za negativne ulaze daje nulu, a za pozitivne uvijek nenegativnu vrijednost. Jedini problem je u nuli gdje nema definiranu derivaciju, no za to postoji npr. softplus funkcija koja logaritamski aproksimira derivaciju rampe (Heavisideovu step funkciju).

Najveća prednost relu funkcije očito je njena jednostavnost.

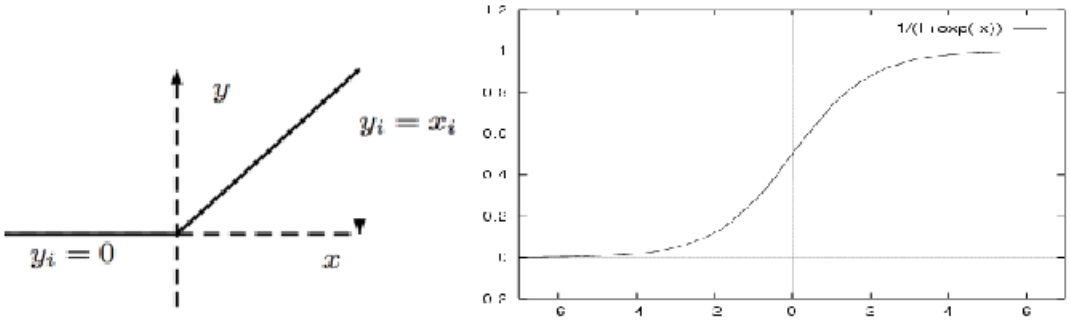
3.5.2. Softmax funkcija

Softmax funkcija prevodi pojedine težine neuronske mreže u normaliziranu vrijednost vjerojatnosti (pozitivan realan broj između 0 i 1). Na taj način softmax daje vjerojatnosti za svaki izlaz koje izravno odgovaraju oznaci kategorije vrijednosti u problemu klasifikacije. Na kraju, odabrana će kategorija biti ona s najvećom vjerojatnosti.

Konkretan primjer softmax funkcije je sigmoidalna logistička funkcija (formuka na slici 3.5.). Ona je dobila ime prema svom oblika slova sigma i ona se često koristi kao aktivacijska funkcija (kao i softmax općenito) umjesto jednostavne step-funkcije budući da je ona glatka funkcija koja u svakoj točki ima definiranu derivaciju i daje željenu vjerojatnosti.

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

Slika 3.5: Sigmoidalna funkcija



Slika 3.6: ReLU (lijevo) i sigmoidalna (desno) aktivacijska funkcija

3.6. Sprečavanje problema prenaučenosti

Problem prenaučenosti (eng. overfitting) jedan je od najvećih problema kod učenja modela. Problem se sastoji u tome što model počinje previše odgovarati konkretnim primjerima na kojima je treniran čime više ne odgovara generalnim uzorcima već samo tim specifičnima iz skupa za učenje. Zbog toga važno je promotriti neke od tehnika koje efektivno ublažavaju taj problem. Generalno se ovakve strategije koje pokušavaju učiniti model učinkovitijim ne samo na uzorke za učenje nego i nove ulaze podrazumijevaju pod pojmom regularizacije.

3.6.1. Rano zaustavljanje

Rano zaustavljanje (eng. early stopping) regularizacijska je forma koja promatra performanse učenja modela kroz vrijeme. Glavna ideja jest zaustaviti učenje u trenutku kad poboljšavanje učenja modela počinje imati manji utjecaj od greške koja se javlja pri učenju.

3.6.2. Dropout

Dropout tehnika je relativno jednostavna, ali učinkovita tehnika izbjegavanja prenaučenosti. Glavna ideja jest uzeti u obzir određen dio (npr. 50%) neurona slučajnim odabirom u svakom koraku treniranja, a ostale neurone zanemariti. Na taj način se postiže ravnomjernija raspodjela učenja po neuronima odnosno u velikoj mjeri se sprječava da određeni neuroni na sebe preuzimaju većinski dio učenja.

4. Konvolucijske neuronske mreže

Duboke unaprijedne neuronske mreže odnosno višerazinski perceptroni (MLP) jesu umjetne neuronske mreže koje propagiraju informaciju kroz aproksimativnu funkciju od ulaza dajući izlaz. Uz njih, postoje i povratne neuronske mreže koje za razliku od unaprijednih imaju povratne veze (eng. RNN – recurrent neural networks) što znači da izlazi sloja neurona mogu kasnije utjecati na njihov ulaz odnosno na cjelokupni proces učenja imajući tako svojstvo memorije.

Konvolucijske neuronske mreže (eng. CNNs – convolutional neural networks) vrsta su dubokih unaprijednih neuronskih mreža s posebnom primjenom u procesiranju podataka organiziranih u 1-D, 2-D i 3-D strukture. Primjer takvih jednodimenzionalnih podataka su kontinuirano uzorkovanje nakon određenog vremenskog intervala, dok je primjer dvodimenzionalnih podataka obrada podataka slike predstavljene kao 2-D rastersko polje piksela (slikovnih elemenata).

Inspiraciju za konvolucijske mreže može se pronaći u biologiji ljudskog vizualnog korteksa. Polja receptora za vid su grupe neurona zadužene za percepciju dijela vidnog polja, dok se susjedna polja dijelom preklapaju. Tako dobiveni poduzorci šalju svoj izlaz u daljnje slojeve neurona koji tada stvaraju sliku sa sve kompleksnijim svojstvima poput linija, oblika i cijelih objekata naposljetku, te uz to sve prisutne su boje.
Može se reći da su slojevi koji otkrivaju kompleksne uzorke analogni konvolucijskim slojevima (eng. convolutional layers) konvolucijske neuronske mreže, svojstva poput linija i oblika odgovaraju značajkama koje se detektiraju uporabom različitih filtera/jezgara (eng. filter/kernel), a sam opisani proces analogan je operaciji konvolucije (eng. convolution) u konvolucijskim neuronskim mrežama.

Važno je odmah naglasiti da je operacija konvolucije i procesno i resursno zahtjevan za CPU, GPU i RAM. Stoga su, uz navedeno, u konvoluciji važni su i slojevi sažimanja (eng. pooling layers) koji koriste jezgre kako bi ekstrahirali najvažnije informacije iz svakog dijela slike na temelju određenog kriterija i tako stvorili pojednostavljenu reprezentaciju slike koja se dalje može tretirati kao ulaz za potpuno povezan sloj.

Kako bi provođenje konvolucije bilo efikasno, kritičan je izbor tzv. hiperparametara specifičnih za konvoluciju poput vrsta i dimenzija jezgara te broju slojeva za sažimanje. Također, važan je i učinkovit raspored samih slojeva unutar topologije modela konvolucijske mreže. Primjer takvog jednog jednostavnijeg rasporeda slojeva mogao bi biti: konvolucijski sloj, sloj za sažimanje, potpuno povezan sloj, sloj za dropout te izlazni sloj kroz npr. sigmoidalnu ili ReLU funkciju.

Opišimo još operaciju konvolucije koristeći matematičku definiciju. Operacija konvolucije definira se kao funkcija s(t) gdje t može biti vrijeme npr. kod uzimanja uzoraka vrijednosti nekog podatka kroz vrijeme ili pak polje podataka. Ta je funkcija jednaka konvoluciji nad višedimenzionalnim poljem ulaza x i višedimenzionalnim poljem jezgara w. Za slučaj slike oba polja bit će dvodimenzionalna. Sam izlaz te funkcije konvolucije nazive se mapom značajki (eng. feature map).

4.1. Ključna svojstva operacije konvolucije

4.1.1. Rijetke interakcije

Za razliku od potpuno povezanog modela, konvolucijske mreže nemaju sve interakcije tj. nisu svi izlazi povezani sa svim ulazima – odakle potječe ime svojstva: rijetke interakcije (eng. sparse interactions). To je postignuto zahvaljujući tome što su veličine jezgara manje od veličine ulaznog polja. To znači da se konvoluiranjem jezgre preko slike kao rezultat dobiva skup podataka manje veličine i s manje parametara čime cjelokupni proces postaje manje resursno zahtjevan. Također, konvolucija postaje i manje računski zahtjevna jer umjesto dosadašnjih m x n parametara (za m ulaza i n izlaza), za ograničen broj izravnih konekcija k koje izlaz može imati (k je optimalno nekoliko redova veličine manji od m), potrebno je samo k x n parametara čime je složenost procesa izravno smanjena. Općenito, broj izravnih konekcija je smanjen, no time nije umanjena neizravna povezanost neurona iz dubljih s neuronima iz plićih slojeva što znači da su sva važna svojstva ulaza sačuvana.

4.1.2. Dijeljenje parametara

Za razliku od potpuno povezanog modela gdje se svaki element polja težina (w) koristi samo jednom pri izračunu dotičnog izlaza, kod konvolucijskih se mreža pojedine težine dijele između više jedinica odnosno stvara se samo jedan skup parametara zajedički za sve elemente ulaznog polja. To se postiže i konvolucijom jezgre po svim elementima ulaza čime se svaki element polja jezgre primjenjuje na svaki element polja ulaza. Pritom su jedino problematični rubni elementi gdje se informacije djelomično gube – stoga se koristi tehnika nadopunjavanja o kojoj će biti riječ u kasnijem primjeru.

4.1.3. Ekvivarijantnost reprezentacije

Svojstvo ekvivarijantnosti reprezentacije govori da se izlaz mijenja na isti način kao i ulaz. Prema definiciji f(x) je ekvivarijantna s g(x) ako vrijedi: f(g(x)) = g(f(x)) gdje f predstavlja konvoluciju, a g translaciju. Na primjeru slike, ako se objekt u ulaznoj slici pomakne za određeni iznos, za isti taj iznos će se pomaknuti njegova reprezentacija u mapi značajki.

4.2. Sažimanje

Sažimanje (eng. pooling) je funkcija koja izlaz na određenoj lokaciji zamjenjuje statističkom vrijednosti izlaza s lokacija iz njegovog susjedstva. U tom procesu glavnu ulogu imaju jezgre koje su polja određene veličine koje prelaze preko cijelog ulaznog polja s određenim korakom (eng. stride). Pritom različiti kriteriji mogu biti korišteni poput sažimanja maksimalnom vrijednosti, srednjom vrijednosti, težinskim usrednjavanjem koje se zasniva na udaljenosti od centralnog piksela. Jedna od najčešće korištenih tehnika je sažimanje maksimumom (eng. max pooling).

Jedno od značajnih svojstava koje sažimanje daje pri prepoznavanju uzoraka je invarijantnost na translaciju što je na primjeru slike invarijantnost na položaj objekta

na slici. To okvirno znači da ako se ulaz translatira za relativno mali iznos, izlaz funkcije sažimanja se ne mijenja. To svojstvo može biti korisno ako točna pozicija svojstva na slici nije važna, već samo njegova prisutnost – primjer tomu je prepoznavanje lica na slici. U drugim slučajevima, lokacija može biti važna – primjer je kut koji je obilježen spajanjem dvaju bridova jednog objekta.

Pri sažimanju važno je primijetiti jednu negativnu pojavu – rubni elementi polja podataka npr. slike manje su zastupljeni u reprezentaciji dobivenoj sažimanjem. Općenito, za veličinu jezgre k i ulaz veličine m, izlaz je veličine m-k+1. Kako bi izlaz bio jednako velik kao i ulaz, potrebno je primijeniti tehniku nadopune s k-1 nula na rubovima slike (eng. zero padding). Na taj način spriječeno je smanjivanje reprezentacije s dubinom sloja u mreži. Jedini nedostatak takvog rješenja je da rubni elementi ne utječu u jednakoj mjeri kao elementi u sredini na izlaznu reprezentaciju.

U svrhu ilustracije postupka sažimanja, dan je primjer prepoznavanja rukopisa u 4.3.

4.3. Primjer konvolucije

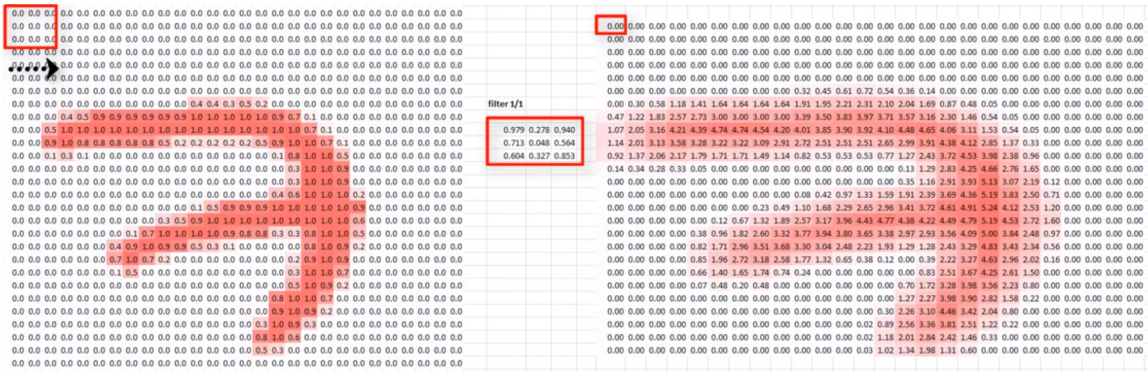
4.3.1. Prepoznavanje rukom pisanih znamenki

Primjer rukom pisanih znamenki iz MNIST baze podataka ovdje će nam poslužiti prvenstveno za ilustraciju koncepta konvolucije i sažimanja. MNIST baza sadrži skup slika s rukom pisanim znamenkama i kao takva predstavlja dobar izvor podataka za treniranje i testiranje modela koji provodi klasifikaciju u jednu od deset kategorija (znamenke 0-9).

Za sljedeći primjer definirat ćemo: ulazno polje vrijednosti veličine 28x28 piksela, polje jezgre veličine 3x3 koje će se koristiti za konvoluciju, filter sažimanja veličine 2x2 i koraka 2 te kao tehniku sažimanja odabrati sažimanje maksimumom. U ovom primjeru konkretno uzet ćemo jedan rukopis broja 7 (slika 4.1.).

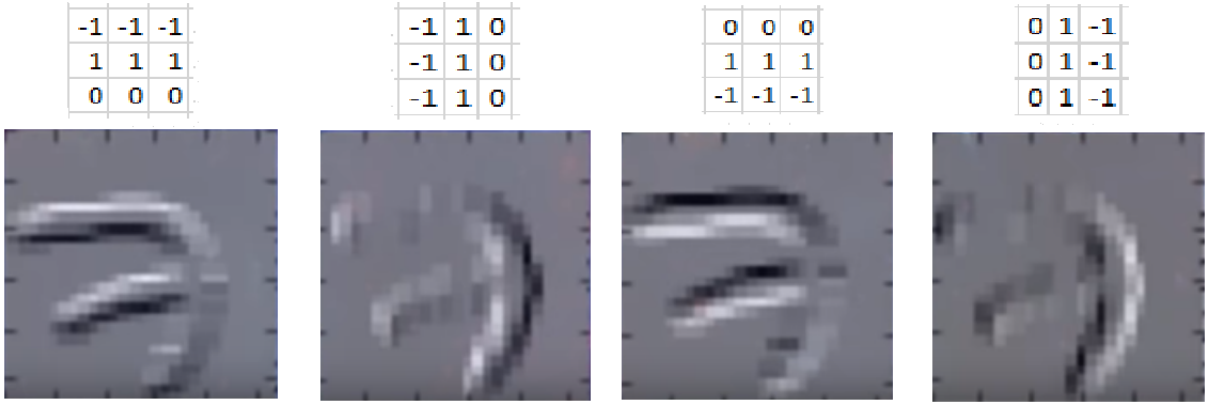
Slika 4.1: Rukom pisan broj 7 iz MNIST baze podataka

Najprije, ulaznu sliku koja se sastoji od 28x28 piksela određenih vrijednosti potrebno je konvoluirati određenom jezgrom kako bi se dobila njena reprezentacija. Odabrana je jezgra (filter) veličine 3x3. Postupak je sljedeći: jezgra veličine 3x3 prolazi svim 3x3 regijama ulazne slike i za svaku od tih regija radi skalarni produkt jezgre i trenutne regije ulazne slike čime dobiva skalarnu vrijednost koja se upisuje na odgovarajuću poziciju izlaza. Na kraju se kao rezultat dobiva slika manje dimenzije: umjesto 28x28 sada je 26x26 gdje je 26=28-3+1, odnosno općenito n=m-k+1 (n je broj izlaza, m je broj ulaza, k je dimenzija jezgre). Nadalje, važno je naglasiti kako smo sada promotrili samo primjenu jedne jezgre u jednom konvolucijskom sloju jedan put. Ovaj postupak se dalje primjenom novih filtera može ponoviti i tako dobiti kompleksniju reprezentaciju slike (ilustracija na slici 4.2.).



Slika 4.2: Konvolucija nad danim slikovnim ulazom (lijevo) koristeći filter (sredina) daje izlaz konvolucije (desno)

Dani filter u primjeru konvolucije bila je matrica popunjena slučajnim vrijednostima. Međutim, u procesu konvolucije su vrijednosti polja jezgre ključni za svojstvo koje prepoznaju. Korisno je vizualizirati jezgre koje ekstrahiraju jednostavno svojstvo poput rubova. Stoga na slici je dan primjer četiri takve jezgre koje prepoznaju rubove, konkretno: gornji horizontalni, lijevi vertikalni, donji horizontalni i desni vertikalni rub. Pritom na slici bijela boja predstavlja područje koje pojedini filter raspoznaje. Može se uočiti da filteri koji raspoznaju horizontalne rubove imaju tako posloženu i jezgru koja „1“ ima tamo gdje su bijeli rubovi čime se to svojstvo ekstrahira (slika 4.3.).

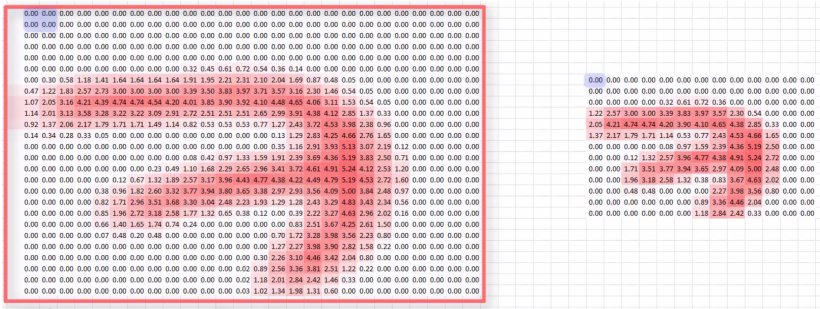


Slika 4.3: Primjeri jezgara konvolucije za obrise znamenki

Sljedeće, promotrimo kako se iz konvolucijom dobivene slike primjenom sažimanja dobiva konačna reprezentacija slike u ovakvom jednom ciklusu konvolucije i sažimanja (ilustracija na slici 4.4).

Nakon konvolucije reprezentacija dane ulazne slika imala je 26x26 piksela. Sada primijenimo tehniku sažimanja maksimumom pomoću filtera sažimanja veličine 2x2 i koraka 2. Sljedećim redom prolazimo po rezultatu konvolucije: uzmemo prvu regiju veličine koja odgovara veličini filtera (2x2 piksela), pronađemo maksimum od tih elemenata i zapišemo ga kao prvi element rezultantnog polja, zatim se pomaknemo za korak (2 piksela) udesno i ponovimo postupak. Postupak se ponavlja sve do kraja trenutnog reda, a zatim se prelazi za onoliko redova dolje koliki je korak (u primjeru 2 reda dolje). Tada se isti postupak ponavlja sve do kraja tog reda odnosno iterativno do kraja polja po visini. Na kraju se dobiva polje veličine 13x13 jer je i po širini i po visini 2 puta smanjeno. Promatrajući unutar jednog reda, svaka 2 susjedna piksela

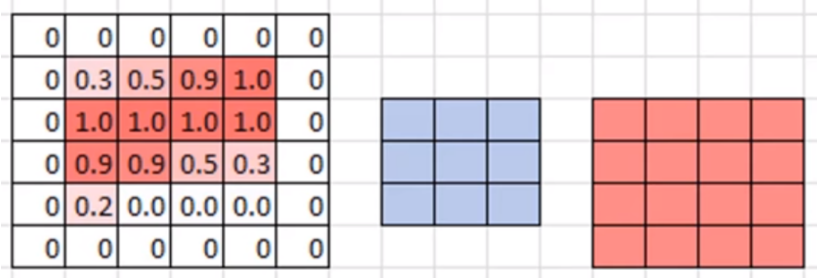
uzimana su točno jednom jer je korak bio jednak širini filtera, te je slično vrijedilo za piksele unutar jednog stupca gdje je korak bio jednak visini filtera.



Slika 4.4: Primjeri sažimanja maksimumom (plavo jedna regija za sažimanje)

Na kraju, treba još napomenuti da smo već u prvom koraku konvolucijom iz ulazne slike 28x28 dobili sliku 26x26 primijenivši dani filter. Međutim, ukoliko je kritično očuvati dubinu reprezentacije s potrebnim svojstvima i osigurati da izlaz bude iste veličine kao i ulaz, korisno je primijeniti nadopunu nulama. Ilustrirajmo to na primjeru.

Na slici 4.5. prikazana je glavna ideja nadopune nulama u vidu dimenzija ulaznog i izlaznog polja. Originalni ulaz slike ima dimenzije 4x4 piksela, primijenjena jezgra konvolucije veličine je 3x3, te prema dosad pokazanom može se zaključiti da će izlaz biti dimenzija nxn gdje je n=m-k+1 – konkretno n=4-3+1=2. Međutim, ako inicijalno dodamo „obrub“ nula oko ulaznog polja, veličina ulaznog polja postat će 6x6 čime će širina odnosno visina izlaznog polja biti n=6-3+1=4. Time smo iz ulaza dimenzije 4x4 dobili izlaz dimenzije 4x4 što se željelo postići. Općenito, važno je napomenuti da je ponekad potrebno dodati i više od jednog sloja „obruba“ nulama kako bi se zadržala reprezentacija tako da je broj „obruba“ nulama još jedan dodatan hiperparametar konvolucije koji je potrebno namjestiti s ciljem optimizacije same konvolucije.



Slika 4.5: Primjer nadopune nulama – ulaz (lijevo), filter (sredina), izlaz (desno)

Konačno, stavimo konvoluciju u kontekst konvolucijske neuronske mreže. Obično imamo nekoliko konvolucijskih slojeva u kojima se koriste različite jezgre za prepoznavanje različitih uzoraka različite kompleksnosti, te na izlazu aktivacijske funkcije poput ReLU. Iza toga slijedi sloj za sažimanje kojim se dobije umanjena reprezentacija slike sa svim potrebnim svojstvima zadržanim, nakon čega je još korisno dodati sloj za sprečavanje prenaučenosti poput „dropout“ sloja te zatim sloj koji će dobivenu 2-D reprezentaciju prevesti u 1-D reprezentaciju (eng. flatten layer). Tako dobiveno polje je konačno ulaz za skrivene slojeve tipične za potpuno povezane modele odgovorne za učenje modela na čiji se izlaz na kraju ponovno može primijeniti neka od tehnika sprečavanja prenaučenosti i naposljetku dobiti

željeni rezultat klasifikacije koristeći aktivacijsku funkciju (npr. softmax) za klasifikaciju ulaza u jednu od 10 kategorija (znamenki 0-9).

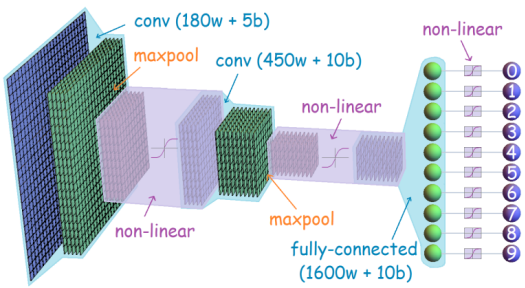
5. Zaključak

Konvolucijske neuronske mreže imaju važnu primjenu u prepoznavanju objekata na slikama, klasifikaciji slika, ali i mnogim drugim područjima poput analize prirodnih jezika, klasifikaciji rečenica, strojnom prevođenju, sustavima za preporučivanje, analizi sentimenata i mnoštvu drugih. Niz je primjena koje su, na prvi pogled, međusobno potpuno različite, no zajedničko im je da podaci o njima sadrže određena svojstva koje je primjenom konvolucijskih neuronskih mreža moguće učinkovito prepoznati i zatim analizirati tako dobivenu reprezentaciju.

Konkretno, prepoznavanje objekata na slici prirodan je primjer jedne reprezentacije – objekti su sami po sebi jedna kompleksna reprezentacija. Stoga je krajnji cilj doći do te visoke razine apstrakcije što je moguće kroz niz slojeva jedinica od kojih je svaka zadužena za analizu ulaznih podataka po drugačijem kriteriju. Osim kombinacije različitih slojeva, bitan je i njihov prirodan poredak – od onih koji prepoznaju jednostavne uzorke pa sve do onih koji prepoznaju relativno kompleksne.

Zanimljivo je spomenuti kako su konvolucijske mreže kroz povijest bile pokretač značajnih istraživanja u polju istraživanja ljudskog mozga i dubokom učenju. Zahvaljujući njima, neuronske mreže općenito postale su popularno polje istraživanja. Također, konvolucijske su mreže jedne od prvih iskoristile moćan alat – algoritam propagacije greške unatrag (eng. backprop) čime je omogućeno brže i efikasnije računanje odnosno procesiranje podataka i učenje modela. U tom pogledu, konvolucijske su mreže u usporedbi s potpuno povezanim modelima bile u vodstvu. Jedne od prvih primjena bile su upravo u prepoznavanju rukopisa poput one koja je rezultirala MNIST bazom rukom pisanih znamenki koja je obrađena i u ovom radu kao primjer jedne od prvih primjena na čijem je primjeru vidljiva ideja i koncept konvolucijskih mreža (slika 5.1.). Sve je to konvolucijske mreže dovelo do današnje popularnosti u komercijalnom smislu, ali i sve intenzivnijih istraživanja na tom polju.

Danas nam predstoje veliki izazovi u ovom području. Najprije, možemo se odlučiti usavršavati postojeće modele konvolucijskih neuronskih mreža i pronalaziti optimalne postave hiperparametara što je jedan od najvećih izazova pri konstrukciji bilo kojeg modela dubokih neuronskih mreža, ali posebno kod konvolucijskih koje imaju i mnoštvo drugih hiperparametara. Na taj način, možemo poboljšati efikasnost modela u postojećim primjenama. S druge strane, možemo pronaći novo područje primjene konvolucijskih mreža čime bi one dobile još jedan vid primjene. U svakom slučaju, izazov je osmisliti efikasnu topologiju mreže za specifičnu primjenu i namjestiti hiperparametre na odgovarajući način.



Slika 5.1.: Skica topologije konvolucijske mreže za prepoznavanje znamenki

6. Literatura

[1] Ian Goodfellow, Yoshua Bengio, Aaron Courville. Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org/

[2] Yann LeCunn, Corinna Cortes, Christopher J.C. Burges, *The MNIST database of handwritten digits*, http://yann.lecun.com/exdb/mnist/, pristup: 14.4.2019.

[3] Massachusetts Institute of Technology, Prof. Patrick Henry Winston. *Artifical Intelligence* (MIT Course Number 6.034), https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-034-artificial-intelligence-fall-2010/lecture-videos/, pristup: 14.4.2019.

[4] Massachusetts Institute of Technology, Prof. Tomi Jaakkola. *Machine Learning* (MIT Course Number 6.867), https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-867-machine-learning-fall-2006/, pristup: 14.4.2019.

[5] DeepLizard. *Deep Learning And Machine Learning Intro*, 22.11.2017. http://deeplizard.com/learn/video/gZmobeGL0Yg, pristup: 14.4.2019.

[6] Complex Projective 4-Space. *Deep Learning with the Analytical Engine*, 6.2.2016., https://cp4space.wordpress.com/2016/02/06/deep-learning-with-the-analytical-engine/, pristup: 14.4.2019.

[7] Laerd Statistics. *Linear Regression Analysis Using SPSS Statistics*, https://statistics.laerd.com/spss-tutorials/linear-regression-using-spss-statistics.php, pristup: 14.4.2019.

[8] STHDA. *Partitioning cluster analysis: Quick start guide - Unsupervised Machine Learning*, http://www.sthda.com/english/wiki/print.php?id=236, pristup: 14.4.2019.

[9] Hongzi Mao, Mohammad Alizadeh, Ishai Menache, Srikanth Kandula, Massachusetts Institute of Technology, Microsoft Research. Resource Management with Deep Reinforcement Learning. HotNets, 10.11.2016. http://people.csail.mit.edu/hongzi/content/publications/DeepRM-HotNets16.pdf

[10] Kota Ando, Shinya Takamaeda-Yamazaki, Masayuki Ikebe, Tetsuya Asai, Masato Motomura. A Multithreaded CGRA for Convolutional Neural Network Processing. Scientific Research Publishing, 2017. https://www.researchgate.net/publication/318025612_A_Multithreaded_CGRA_for_Convolutional_Neural_Network_Processing

[11] Anish Singh Walia, Towards Data Science. *Activation functions and its types – Which is better?*, https://towardsdatascience.com/activation-functions-and-its-types-which-is-better-a9a5310cc8f, pristup: 14.4.2019.

[12] Pedro Antonio Gutierrez, Cesar Hervas Martinez, Manuel Lozano. Designing multilayer perceptrons using a Guided Saw-tooth Evolutionary Programming Algorithm. Soft Computing, 2010. https://www.researchgate.net/publication/220176288_Designing_multilayer_perceptrons_using_a_Guided_Saw-tooth_Evolutionary_Programming_Algorithm

7. Sažetak

Ovaj rad rezultat je istraživanja osnovnih koncepata dubokog učenja polazeći od osnova strojnog učenja i umjetnih neuronskih mreža na kojima ono počiva. Kao prvo, obrađuju se glavne paradigme strojnog učenja i značajni algoritmi vezani uz njih: nadzirano učenje (primjeri: regresija - linearna i polinomijalna, klasifikacija - naivni Bayes), nenadzirano učenje (primjer: grupiranje – K-means) i podržano učenje (primjer: Q-učenje). Nakon toga, obrađuju se koncepti dubokog učenja: funkcije gubitka (primjeri: kvadratna pogreška (MSE), negativna log izglednost (NLL)), stohastički gradijentni spust (SGD), potpuno povezani model i višerazinski potpuno povezan model (MLP), treniranje duboke neuronske mreže algoritmom SGD pomoću izračuna gradijenta algoritmom propagacije greške unatrag, funkcije aktivacije (ReLU, softmax na primjeru sigmoidalne funkcije) te tehnike sprečavanja problema prenaučenosti (primjeri: rano zaustavljanje, dropout). Na kraju, poseban je naglasak stavljen na konvolucijske duboke neuronske mreže posebno s primjenom u računalnom vidu na primjeru prepozavanja rukom pisanih znakova koristeći primjer iz MNIST baze podataka koja sadrži rukopise znamenki. Pritom su obrađeni najvažniji pojmovi poput konvolucije i sažimanja s ciljem dobivanja općeg uvida u način funkcioniranja konvolucijskih neuronskih mreža.

Ključne riječi: duboko učenje, strojno učenje, neuronske mreže, konvolucijske neuronske mreže, nadzirano učenje, nenadzirano učenje, podržano učenje, regresija, klasifikacija, grupiranje, funkcije gubitka, stohastički gradijentni spust, potpuno povezan model, problem prenaučenosti, prepoznavanje rukopisa, konvolucija, sažimanje.