

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 5393

**PREPOZNAVANJE RUKOPISA
STROJNIM UČENJEM I PRIMJENA U
IDENTIFIKACIJI POPUNJENIH POLJA NA
UPLATNICI**

Josip Mrđen

Zagreb, svibanj 2018.

Zahvaljujem se svima koji su bili uz mene tijekom mog školovanja,
posebice svojim roditeljima koji su mi pružili neizmjernu ljubav i podršku u
svakom trenutku mog života

Zahvaljujem se mentoru, prof. dr. sc. Domagoju Jakoboviću, na pomoći,
pruženom znanju i savjetima za vrijeme svog mentorstva.

SADRŽAJ

1.	Uvod.....	1
2.	Metode klasifikacija slika	2
2.1.	Stroj potpornih vektora.....	2
2.2.	Algoritam K – najbližih susjeda.....	3
2.3.	Umjetne neuronske mreže.....	4
2.3.1.	Model neurona	4
2.3.2.	Model neuronske mreže	6
2.3.3.	Učenje neuronske mreže – algoritam Backpropagation	7
3.	Konvolucijske neuronske mreže.....	11
3.1.	Struktura konvolucijske neuronske mreže	11
3.2.	Slojevi konvolucijske neuronske mreže	12
3.2.1.	Konvolucijski sloj	12
3.2.2.	Sloj sažimanja	13
3.2.3.	Potpuno povezani sloj	14
3.2.4.	Slaganje slojeva konvolucijske neuronske mreže i primjeri	15
4.	Identifikacija popunjenih polja na uplatnici i rezultati	19
4.1.	Skup podataka.....	20
4.2.	Korištene tehnologije	21
4.3.	Treniranje konvolucijske neuronske mreže.....	22
4.4.	Pretprocesiranje uplatnice	29
4.4.1.	Detekcija uplatnice na površini	30
4.4.2.	Detektiranje polja na uplatnici.....	32
4.4.3.	Detekcija linija, riječi i slova iz polja uplatnice.....	33

4.5.	Prepoznavanje teksta na uplatnici	34
5.	Zaključak	37
6.	Literatura	38
7.	Sažetak	39
8.	Abstract	40

1. Uvod

Cilj završnog rada je prepoznavanje teksta (slova i brojeva) u svrhu ekstrakcije podataka iz rukom pisanih uplatnica. Kako bi se takav zadatak ostvario, potrebno je domensko znanje iz računalnog vida, discipline koja se bavi razumijevanjem, ekstrakcijom i klasifikacijom slika, slikovnih segmenata i multimedijских podataka.

Računalni vid se razvojem strojnog učenja u zadnjih nekoliko godina znatno popularizirao te je sve više alata koji se bave analizom informacija dobivenih iz slika. Među raznim granama računalnog vida, prepoznavanje teksta igra bitnu ulogu kod prijenosa podataka u digitalni oblik. Digitalizacijom svijeta nastala je potreba da se sva tekstualna informacija drži zapisana u računalima te bazama podataka, no to je zamoran posao i iziskuje napor kod utipkavanja. Zbog toga postoje metode koje skeniranjem mogu iščitati rukom pisanu informaciju i unijeti je u računalno. U ovom završnom radu za raspoznavanje teksta korištene su konvolucijske neuronske mreže, pogodne upravo za analizu slike.

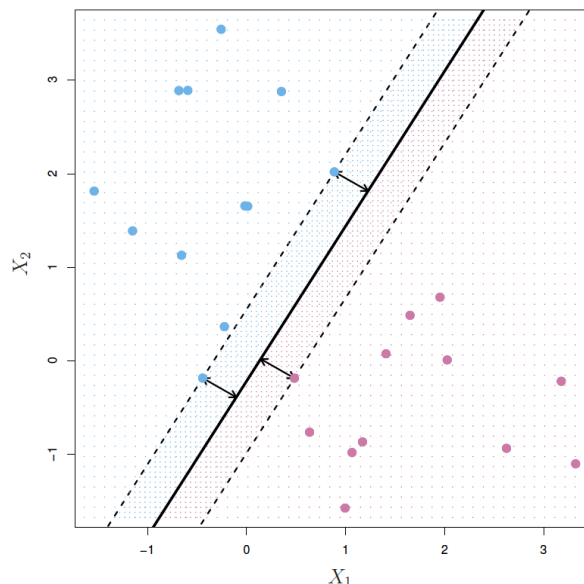
Drugo poglavlje rada opisuje najkorištenije metode klasifikacije slike, s naglaskom na neuronske mreže. U trećem poglavlju detaljnije su opisane konvolucijske neuronske mreže, njihova struktura i prednosti za klasifikaciju slike u odnosu na ostale mreže. Četvrto poglavlje bavi se implementacijskim dijelom i prikazom rezultata prepoznavanja teksta na rukom pisanim uplatnicama. Zaključak navodi prednosti i nedostatke za vrijeme pisanja rada te moguće nadogradnje ostvarenog programskog rješenja za prepoznavanje teksta.

2. Metode klasifikacija slika

Kroz povijest su ljudi razvili mnoge tehnologije i metode za klasifikaciju slika i detekciju objekata na slikama. Klasifikacija slika je proces kojim se na temelju informacija iz slike ona svrstava u neki od ponuđenih razreda. Detekcija objekata na slikama bavi se traženjem pojedinih elemenata na slikama i njihovim označavanjem. Ispod su navedene neke od najčešće korištenih tehnologija za klasificiranje slika[7] s naglaskom na neuronske mreže koje su bile alat za analizu slika u ovom završnog rada.

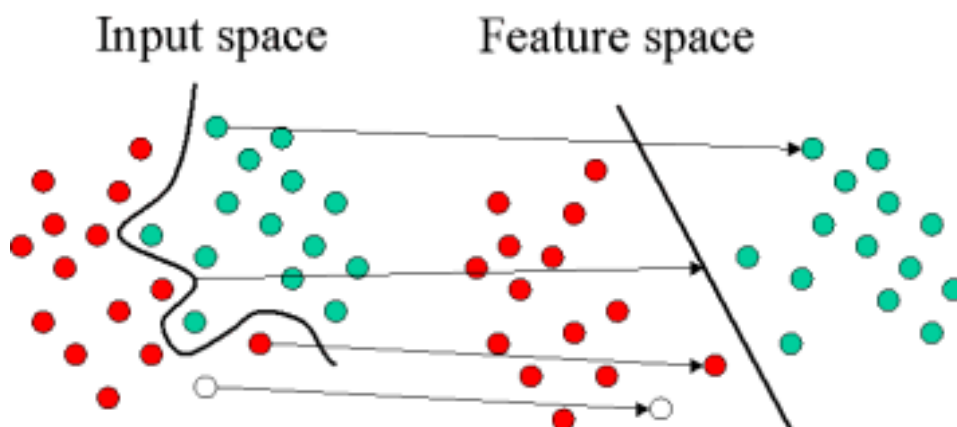
2.1. Stroj potpornih vektora

Stroj potpornih vektora (engl. Support Vector Machines, SVM) je nelinearna i neparametarska tehnologija koja nastoji transformacijama podataka svesti problem klasifikacije na linearan slučaj kojim je moguće klase slika u skupu podataka odijeliti hiperravninom.



Slika 1. Prikaz odjeljivanja klasa podataka pomoću stroja potpornih vektora

U praksi su podaci većinom nelinearno raspoređeni te višedimenzionalni te se zbog toga uvode jezgrene funkcije koje podatke transformiraju iz n -dimenzionalnog u $(n+m)$ -dimenzionalan ali linearan prostor. Odabiranje pogodne jezgrene funkcije težak je postupak te je presudan za brzinu izvođenja algoritma. Traženje hiperravnine također nije trivijalno budući da je moguće da više hiperravnina dobro odvajaju klase slika. Pri tome se koriste potporni vektori po kojem je i sam algoritam dobio ime. Odluka koja će hiperravnina biti najpogodnija za klasifikaciju jest ona koja ima najviše razmaknute potporne vektore od sebe, a da oni i sami dobro razdvajaju klase u vektorskom prostoru.

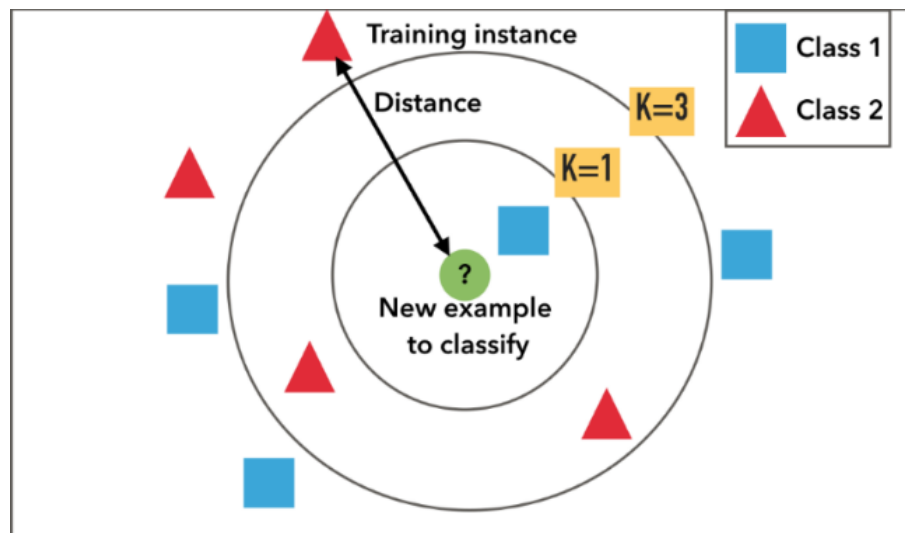


Slika 2. Transformacija skupa podataka kako bi odjeljivanje hiperravninom bilo moguće

2.2. Algoritam K – najbližih susjeda

K – najbližih susjeda (k – nearest neighbors, KNN) je jednostavan neparametarski klasifikacijski algoritam koji zahtjeva malu količinu učenja nad skupom podataka te je zbog toga jedan od korištenijih algoritama za klasifikaciju. Algoritam radi tako da se za predviđanje klase nekog elementa uzme k najbližih susjeda, te se na temelju njih „glasuje“ koja će biti klasa

elementa. Nedostatci ovog algoritma su što on pohranjuje cijeli skup za učenje kao svoju reprezentaciju da odredi susjede, te zbog toga vrijeme predviđanja klase može trajati dugo.



Slika 3. Skica algoritma K - najbližih susjeda

2.3. Umjetne neuronske mreže

Neuronske mreže su među najkorištenijim tehnologijama u računalnom vidu uopće, te predstavljaju bitan dio konektivističkog pristupa postizanju inteligentnog ponašanja. Inspiracija za neuronske mreže dolazi od želje da se prikaže rad ljudskog mozga, pa je tako temeljni građevni materijal neuronske mreže neuron.

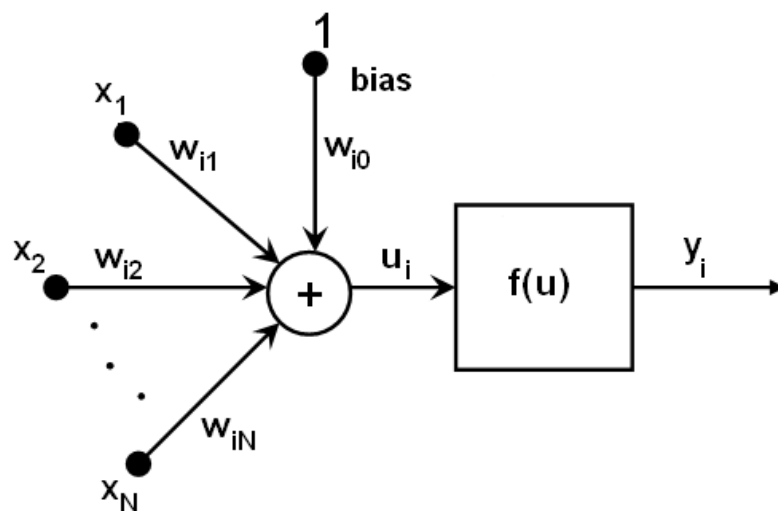
2.3.1. Model neurona

Neuron može imati više dendrita ili ulaza kojima je povezan s drugim neuronima, tijelo koje akumulira ukupnu pobudu, te akson odnosno izlaz koji služi kao pobuda koja se propagira ostalim neuronima. Svaki dendrit posjeduje svoju težinu koja služi za izračunavanje izlaza neurona.

Pobuda (engl. *net*) se računa prema izrazu:

$$net = w_1 \cdot x_1 + w_2 \cdot x_2 + \dots + w_n \cdot x_n - \theta$$

gdje θ predstavlja prag paljenja neurona, w težinu dendrita a x vrijednost izlaza prošlog neurona ili ulaznu vrijednost. Obično se θ u engleskoj literaturi naziva *bias* te je to slobodni član kada aktivacijsku funkciju pomičemo po osi apscisa.



Slika 4. Prikaz umjetnog neurona

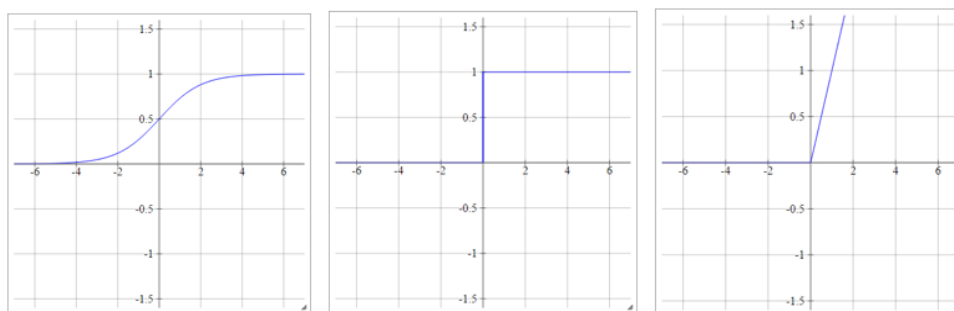
U praksi se θ zamjenjuje dodatnim dendritom s vrijednošću 1 i težinom θ :

$$\begin{aligned} net &= w_0 \cdot x_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + \dots + w_n \cdot x_n = \\ &= \sum_{i=0}^n w_i \cdot x_i \\ &= \vec{w} \cdot \vec{x} \end{aligned}$$

Odsada će se težine zajedno s *biasom* zajednički nazivati težinama neuronske mreže. Kako su mnogi problemi u računarskom svijetu nelinearni, suma svih težina i vrijednosti za pobudu neurona nije dovoljna, pa se primjenjuje i dodatna aktivacijska funkcija na ukupnu pobudu. Odabir

aktivacijske funkcije nije trivijalan problem te se po *no free lunch* teoremu[6] može reći da će za jednu skupinu problema aktivacijska funkcija A pobiti aktivacijsku funkciju B, dok će u drugoj skupini problema biti obrnut slučaj.

U praksi se koristi nekoliko ustaljenih aktivacijskih funkcija. Na slici (Slika 5) su prikazane neke od njih.



|

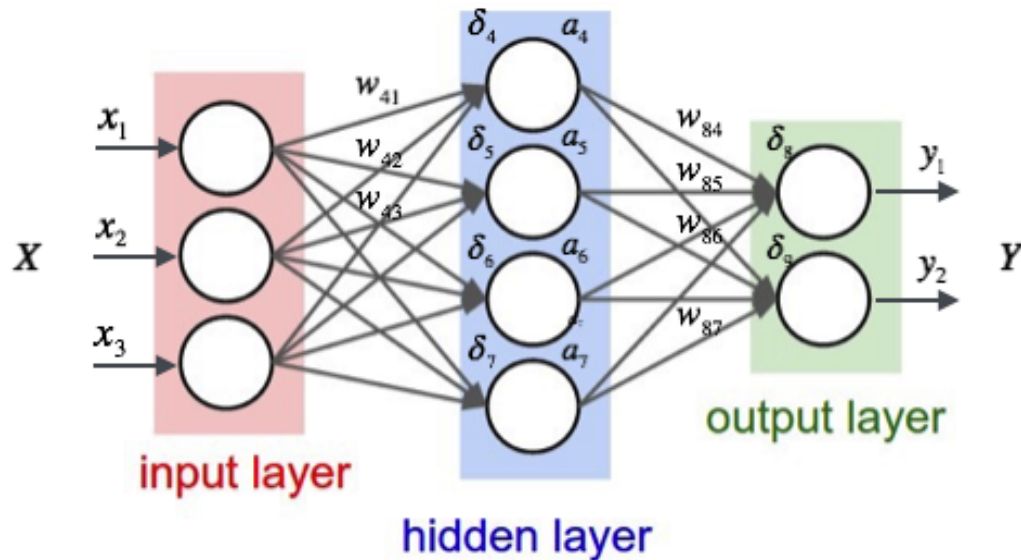
Slika 5. Slijeva nadeso: sigmoidalna, step i ReLU aktivacijska funkcija

2.3.2. Model neuronske mreže

Na slici (Slika 6) je prikazan osnovni model neuronske mreže. Ulazni vektor $\vec{X}$ označava ulazne vrijednosti u neuronsku mrežu, dok izlazni vektor $\vec{Y}$ predstavlja izlazne vrijednosti iz mreže. Krugovi predstavljaju neurone. Svaki „stupac“ neurona predstavlja pojedini sloj neuronske mreže.

Mreža mora imati ulazni (crveno) i izlazni sloj (zeleno), a može imati jedan ili više skrivenih slojeva (plavo). Uz iznimku za ulazni sloj, obično je svaki neuron potpuno povezan sa prethodnim slojem. To znači da neuron ima vezu sa svakim neuronom iz prethodnog sloja.

Spoj između 2 neurona ima svoju težinu te služi za izračunavanje pobude delta. Malim slovom a označen je izlaz iz unutarnjih slojeva neurona nakon apliciranja aktivacijske funkcije na pobudu δ .



Slika 6. Apstraktni prikaz modela neuronske mreže[12]

2.3.3. Učenje neuronske mreže – algoritam Backpropagation

Prije samog algoritma, potrebno je objasniti kako neuronska mreža iz ulaznog vektora vrijednosti kreira rezultat ili izlazni vektor. Recimo da su ulazne vrijednosti dovedene u ulazni sloj neurona u mreži. Svaka vrijednost množi se sa svakom težinom poveznice između 2 sloja te se u neuronima akumuliraju izlazne vrijednosti. Na te vrijednosti aplicira se aktivacijska funkcija te su tako dobiveni izlazi ulazne vrijednosti za sljedeći skriveni sloj ili za izlazni vektor.

Postupak se ponavlja kroz slojeve dok se ne dobije konačna vrijednost za izlazni vektor, te je to rezultat neuronske mreže za dani ulaz. Takva mreža koja iz ulaznog skupa vrijednosti propagira rezultate do izlaznog sloja naziva se unaprijedna neuronska mreža.

Postavlja se pitanje, kako naučiti neuronsku mrežu da ona može za dane ulaze davati zadovoljavajuće izlaze za naš skup podataka. Problem učenja neuronske mreže tada se svodi na traženje kombinacije težina u neuronskoj mreži koje u kombinaciji s akumuliranjem vrijednosti i aktivacijskim funkcijama daju odgovarajuće izlaze. Učenje težina se može napraviti raznim algoritmima, a najpopularniji je Backpropagation algoritam koji koristi metodu

gradijentnog spusta. Također, mogu se primjenjivati i heuristički algoritmi poput genetskog algoritma te mnogi drugi.

Bit algoritma Backpropagation jest pronalaženje minimalne greške u prostoru težina koje su definirane u neuronskoj mreži. Pri tome je kombinacija težina s minimalnom greškom rješenje problema učenja mreže. Sam algoritam zahtjeva nekoliko stvari:

- skup podataka s ulaznim i izlaznim vrijednostima $(\vec{x}, \vec{y})$ gdje je $\vec{x}$ ulaz a $\vec{y}$ željeni izlaz mreže za dani ulaz
- unaprijedna neuronska mreža, bez povezanih neurona u istom sloju (svaki neuron je povezan samo sa sljedećim slojem)
- funkcija greške E koja ovisi o procijenjenom i željenom izlazu mreže
- neprekidnost i diferencijabilnost funkcije greške

Neprekidnost i diferencijabilnost funkcije greške bitna je kako bi računanje njenog gradijenta bilo moguće. Ako je aktivacijska funkcija u mreži diferencijabilna i neprekinuta (npr. sigmoidalna funkcija umjesto step funkcije), funkcija greške će također biti diferencijabilna i neprekinuta budući da se račun izlazne vrijednosti u mreži još samo svodi na sumiranje izlaza neurona iz prethodnog sloja.

Ako se funkcija neuronske mreže označi sa φ (funkcija koja računa izlaz neuronske mreže sa svim sumama i aktivacijskim funkcijama), uz pogodne težine ona bi što bolje aproksimirala funkciju f koja nije zadana eksplicitno već implicitno kroz skup podataka.

Uz oznake $\vec{x}$ za ulazni vektor vrijednosti koje smo doveli na mrežu, $\vec{o}$ kao vektor izlaznog sloja dobivene propagiranjem ulaznih vrijednosti, $\vec{t}$ kao željeni vektor koji želimo dobiti te N koji označava broj primjera, greška neuronske mreže za skup podataka može se izračunati kao:

$$E = \frac{1}{N} \sum_{i=0}^N (\vec{o}_i - \vec{t}_i)^2$$

te se naziva srednja kvadratna greška (engl. Mean Squared Error, MSE).

Pogodnost ovakve formule leži u više faktora:

- greška je uvijek pozitivna vrijednost koja se povećava s povećanjem skupa podataka
- kažnjava veća odstupanja od željenog rezultata kvadratnim uvećanjem, dok manja približava
- lako se računa kao i njena derivacija

Da bi pronašli globalni / lokalni optimum funkcije koju tražimo, potrebno je ažurirati težine, a za to nam treba gradijent funkcije greške u ovisnosti o svakoj težini u mreži. :

$$\nabla E = \left(\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \dots, \frac{\partial E}{\partial w_l} \right)$$

gdje je su brojevi od 1 do l indeksi pojedinih težina. Te bi se težine postupno ažurirale prema izrazu:

$$w_i \leftarrow w_i - \alpha \cdot \frac{\partial E}{\partial w_i}$$

gdje je α stopa učenja. Negativni predznak ukazuje da se pomicanje prema optimumu kreće u smjeru negativnog gradijenta.

U odluci koliko treba ažurirati pojedinu težinu, koristi se pravilo ulančavanja parcijalnih derivacija:

$$\frac{\partial E}{\partial w_{ij}^k} = \frac{\partial E}{\partial a_j^k} \frac{\partial a_j^k}{\partial w_{ij}^k} = \delta_j^k \cdot \frac{\partial}{\partial w_{ij}^k} \left(\sum_{l=0}^{r_{k-1}} w_{lj}^k o_l^{k-1} \right) = \delta_j^k o_i^{k-1}$$

gdje je a_j^k vrijednost neurona j u sloju k prije apliciranja aktivacijske funkcije, a w_{ij}^k težina u k-tom sloju s i-tim neuronom sljedećeg sloja i j-tim neuronom trenutnog sloja.

Parcijalne derivacije funkcije greške različito se računaju za izlazni sloj u odnosu na skrivene slojeve, budući da su skriveni slojevi povezani s više

neurona iz sljedećeg sloja. Propagiranje greške bit će pojašnjeno na jednom primjeru da se razlomak $\frac{1}{N}$ može ispustiti.

Posljednji sloj

Uz oznake $\hat{y}$ za procijenjenu vrijednost mreže te y za traženu vrijednost, funkcije greške jest:

$$E = \frac{1}{2} \cdot (\hat{y} - y)^2 = \frac{1}{2} \cdot (g_o(a_1^m) - y)^2$$

gdje je $g_o(x)$ aktivacijska funkcija za završni sloj. Slovo m označava posljednji sloj u mreži. Parcijalnim deriviranjem dobije se:

$$\delta_1^m = (g_o(a_1^m) - y) g_o'(a_1^m) o_i^{m-1}$$

pa je konačan izraz:

$$\frac{\partial E}{\partial w_{i1}^m} = \delta_1^m o_i^{m-1} = (\hat{y} - y) g_o'(a_1^m) o_i^{m-1}$$

Skriveni sloj

Prvi dio parcijalne derivacije pretvara se u:

$$\delta_j^k = \frac{\partial E}{\partial a_j^k} = \sum_{l=1}^{r^{k+1}} \frac{\partial E}{\partial a_l^{k+1}} \frac{\partial a_l^{k+1}}{\partial a_j^k} = \sum_{l=1}^{r^{k+1}} \delta_l^{k+1} \frac{\partial a_l^{k+1}}{\partial a_j^k} = \sum_{l=1}^{r^{k+1}} \delta_l^{k+1} w_{jl}^{k+1} g'(a_j^k)$$

Pa je konačna parcijalna derivacija funkcije greške u odnosu na težine skrivenog sloja:

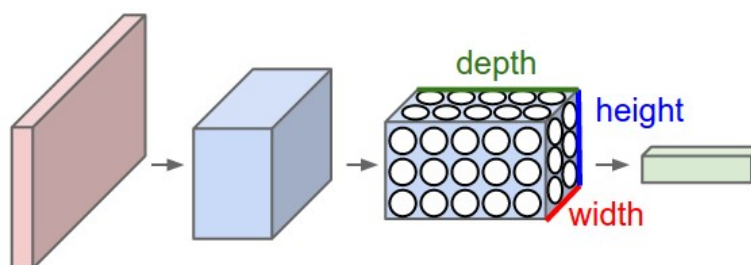
$$\frac{\partial E}{\partial w_{ij}^k} = \delta_j^k o_i^{k-1} = g'(a_j^k) o_i^{k-1} \sum_{l=1}^{r^{k+1}} w_{jl}^{k+1} \delta_l^{k+1}$$

3. Konvolucijske neuronske mreže

Primjena konvolucijskih neuronskih mreža uvelike je porasla posljednjih godina razvojem računalnog vida, a poglavito raspoznavanjem slika i govora. U narednim poglavljima opisane su njene posebitosti te prednosti u odnosu na potpuno povezanu unaprijednu neuronsku mrežu.

3.1. Struktura konvolucijske neuronske mreže

Struktura konvolucijske neuronske mreže (engl. Convolutional Neural Network, ConvNet, CNN) slična je običnoj neuronskoj mreži jer također sadrži težine koje se mogu naučiti te aktivacijske funkcije koje rade pobudu nad neuronima. Konvolucijske neuronske mreže uzimaju u obzir da je ulazni sloj slika ili njoj sličan element te se njena arhitektura organizira na drugačiji način. Tako slojevi CNN-a imaju 3 dimenzije: visinu, širinu i dubinu. Visina i širina ulaznog sloja mreže ovisi o visini i širini slike dok dubina može varirati ali obično se uzima 3 (R, G i B komponenta boje) ili 1 (za crno-bijele ili jednobojne slike). Neuroni u pojedinom sloju su povezani s jednim dijelom sloja nakon (ne moraju biti potpuno povezani). Prednost volumnog rasporeda težina i nepotpune povezanosti u odnosu na običnu unaprijednu neuronsku mrežu jest znatno manji broj parametara. Uz to, uporabom posebnih vrsta slojeva CNN radi operacije koje učenjem i pravom arhitekturom mogu efikasno klasificirati slike [1].

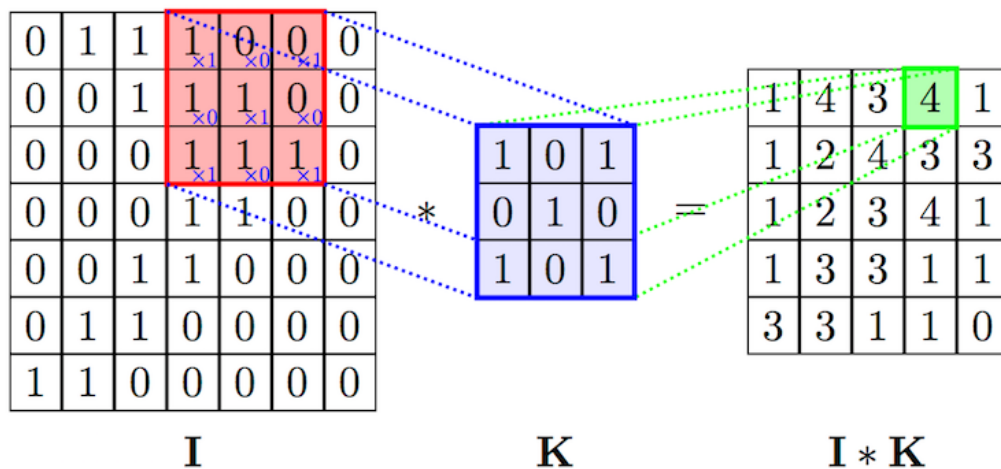


Slika 7. Gruba struktura konvolucijskih neuronskih mreža[12]

3.2. Slojevi konvolucijske neuronske mreže

3.2.1. Konvolucijski sloj

Konvolucijski sloj je temeljni gradivni element konvolucijske mreže. Ovaj sloj sadrži skup filtera, manjih volumnih elemenata koji su veličine manje od slike koja je trenutno u mreži. Primjerice ako imamo sliku dimenzija $24 \times 24 \times 3$, filter nam može biti dimenzija $5 \times 5 \times 3$. Elementi filtera kojeg još zovemo jezgra su parametri za treniranje. Dok radimo unaprijedan prijelaz, filteri rade konvoluciju s trenutnom slikom i računaju skalarni produkt preko cijele širine i dužine filtra. Kako je veličina filtra manja od slike, pomicanjem filtra možemo dobiti vrijednosti za svaki dio slike. Počevši od gornjeg lijevog kuta slike, filter se pomiče udesno svaki put te računamo dvodimenzionalnu konvoluciju. Nakon dolaska do desnog ruba, filter se pomakne prema dolje te se ponovi postupak.



Slika 8. Konvolucija izlazne mape s kernelom (filter, jezgra)

Rezultat konvolucije jest izlazna mapa svojstva kojeg je filter pronašao. Dimenzije izlazne mape ovise o sljedećim hiperparametrima mreže:

1. Dubina – određuje broj filtera koji se koristi. Prednost veće dubine jest ta što svaki filter može tražiti svoj posebni dio slike te se može zapaziti više uzoraka.

2. Skok (engl. *stride*) – određuje skok u broju piksela dok pomičemo filter nakon konvolucije. Tako stride veličine 1 pomiče filter za jedno polje udesno (ili dolje, ovisno o poziciji).

3. Punenje (engl. *padding*) – utječe na širinu i visinu izlazne mape. Najčešće se koristi zero-padding koji određen broj piksela popuni nulama s ruba. Izostanak punjenja smanjuje izlaznu mapu ovisno o širini i visini samog filtera.

U konvolucijskim slojevima dimenzije nove slike računaju se kao:

$$D_{nova} = \frac{D - F + 2P}{S} + 1$$

gdje je D stara dimenzija (može biti i širina i visina), F je širina filtera, P broj piksela punjenja, a S skok između 2 konvolucije filtera. Dubina izlazne mape jednaka je broju filtera koji se koriste.

3.2.2. Sloj sažimanja

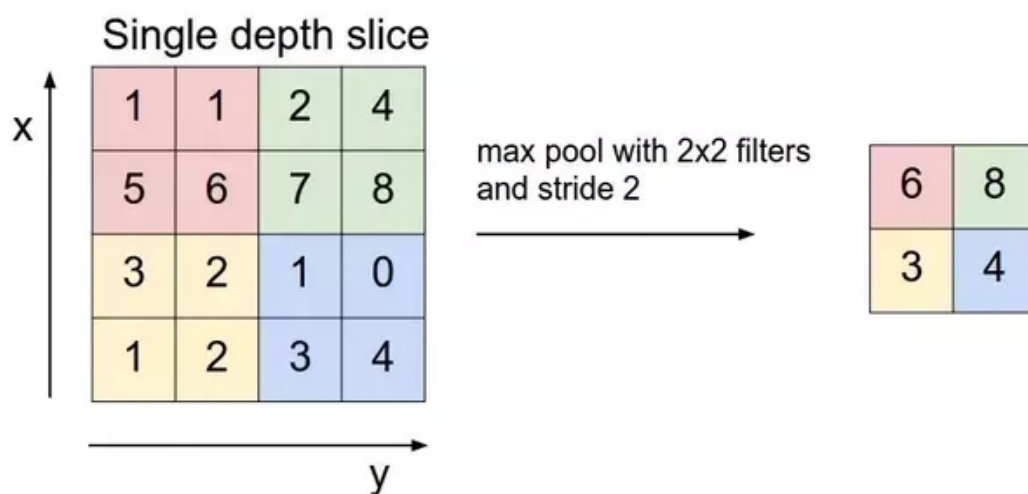
Sloj sažimanja (engl. *pooling layer*) sadrži filter koji sažima dimenzije slike. Filter je drukčiji nego onaj kod konvolucijskog sloja budući da ne sadrži težine. Uloga filtra u ovom sloju jest izabiranje vrijednosti u krugu kojeg filter tih dimenzija obuhvaća. Većinom se u CNN-ovima koristi Max Pooling (prije se znatno više koristio Average Pooling) koji izabire maksimalnu vrijednost između odabranih piksela. Rezultat je čuvanje jačih i izraženijih piksela na slici i njihovo propagiranje dalje u mreži. Također, i ovdje možemo koristiti skok filtera odnosno *stride*. U sloju sažimanja dimenzije izlaznih mapa vrijednosti računaju se kao:

$$D_{nova} = \frac{D - F}{S} + 1$$

gdje je D stara dimenzija, S skok između 2 odabira vrijednosti, a F širina filtera. Primjerice, ako imamo sliku dimenzija $24 \times 24 \times 3$, sa filtrom 2×2 i skokom 2 dobit ćemo prema formuli:

$$D_{nova} = \frac{24 - 2}{2} + 1 = 12$$

odnosno upola manju sliku.



Slika 9. Funkcija sloja sažimanja u konvolucijskim neuronskim mrežama[12]

3.2.3. Potpuno povezani sloj

Sloj je identičan slojevima u potpuno povezanoj unaprijednoj neuronskoj mreži. Obično se koristi u završnim slojevima mreže, no to ne mora biti slučaj. Razlog korištenja potpuno povezanih slojeva jest taj što se propagacijom slike kroz mrežu nastoje smanjiti njene dimenzije. Tada ima smisla raditi s njima budući da potpuna povezanost kvadratni broj veza između slojeva.

Primjerice, za sliku dimenzija $100 \times 100 \times 3$ ulazni sloj imao bi 30 000 ulaznih vrijednosti. Kad bi to potpuno povezali sa skrivenim slojem koji ima 1 000

neurona, imali bismo već 3×10^6 težina za učenje. Zbog toga se potpuno povezani slojevi koriste u kasnijim fazama.

Izlazi iz konvolucijske neuronske mreže za klasifikaciju slika većinom predstavljaju klase kojima slika može pripadati, dok se za aktivaciju tipično koristi Softmax funkcija:

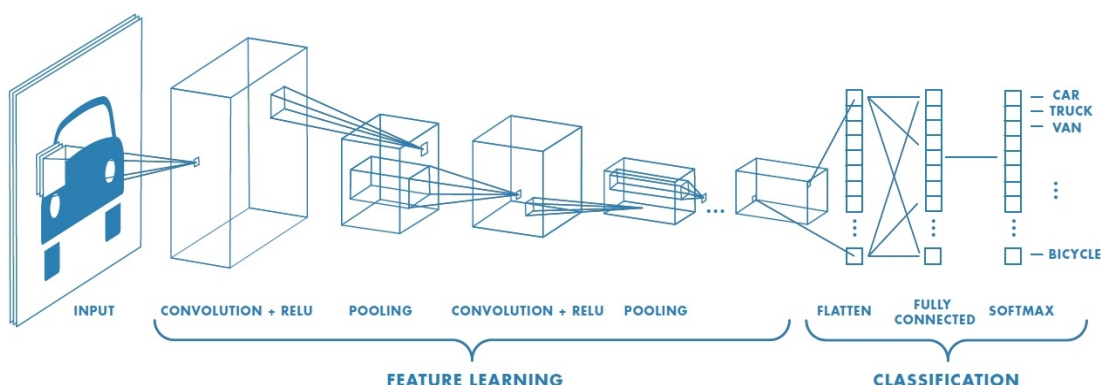
$$s(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

Softmax aktivacijska funkcija normalizira rezultate dobivene na izlazima mreže transformirajući ih da budu pogodni za klasifikaciju, u rasponu $[0, 1]$.

3.2.4. Slaganje slojeva konvolucijske neuronske mreže i primjeri

Slaganje slojeva

Kod pristupa topologiji konvolucijske neuronske mreže, obično se prate ustaljeni tipovi arhitektura. Na slici (Slika 10) je prikazana jedna od ustaljenih arhitektura. Može se uvidjeti ponavljanje konvolucijskih slojeva s ReLU aktivacijom koja je popraćena sa slojem sažimanja. Dimenzije aktivacijskih mapa u širinu i visinu se tako postupno smanjuju dok se dubina povećava za neki manji faktor. Nakon određenog broja takvih slojeva, dobivene izlazne mape vrijednosti se „izravnavaju“ u potpuno povezane slojeve sa završetkom u izlaznim klasama na koji utječe Softmax funkcija ili neka druga.

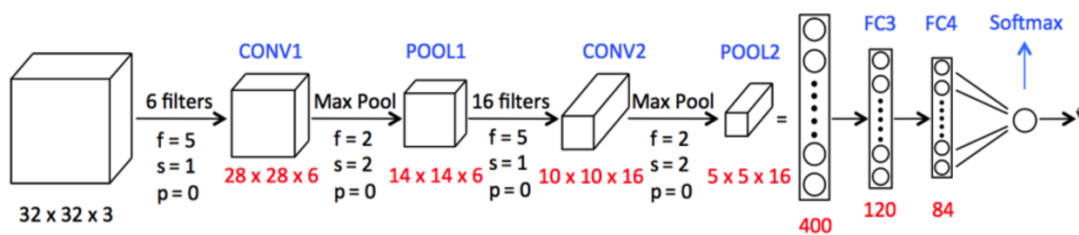


Slika 10. Primjer strukture slojeva u CNN-u

Osim gore navedene arhitekture, ustaljeno je kombinirati više konvolucijskih slojeva koji su tek onda popraćeni sa slojem sažimanja i dr.

Broj parametara

Na slici (Slika 11) je prikazan detaljniji opis konvolucijske neuronske mreže sa svim filterima. Slična arhitektura korištena je i u ovom završnom radu. Ulaz u konvolucijsku neuronsku mrežu jest slika dimenzija $32 \times 32 \times 3$ koja se postupno smanjuje do izlazne mape od 400 neurona u potpuno povezanom sloju.



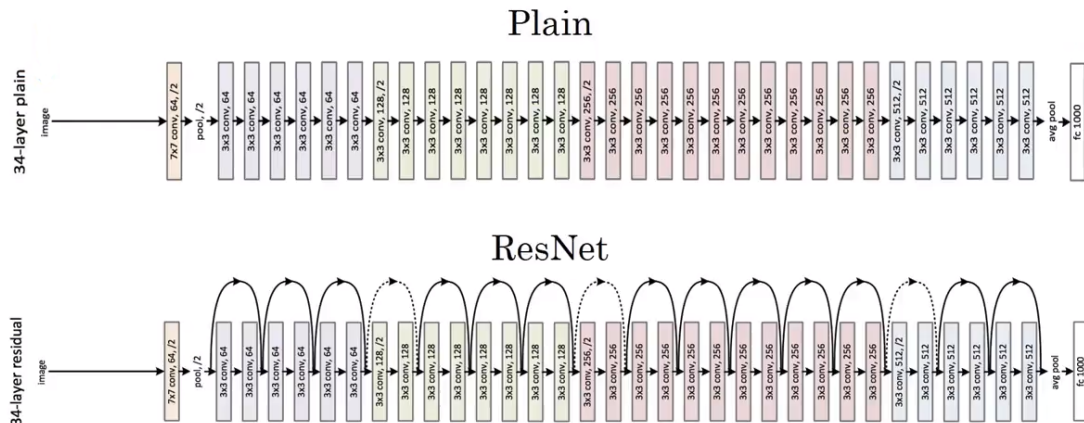
Slika 11. Detaljniji prikaz strukture slojeva uz veličinu izlaznih mapa nakon svakog sloja

Broj težina koje se treniraju u ovom sloju je:

$$\begin{aligned} \sum CONV1 + \sum POOL1 + \sum CONV2 + \sum POOL2 + \sum FC2FC3 + \\ \sum FC3FC4 = 5 \cdot 5 \cdot 6 + 0 + 5 \cdot 5 \cdot 16 + 0 + 400 \cdot 120 + 120 \cdot 84 \\ = 58640 \end{aligned}$$

U pooling slojevima nema težina za treniranje, dok u konvolucijskim slojevima težine predstavljaju sami jezgrieni filteri s kojima se radi konvolucija. Isti problem modeliran običnom unaprijednom potpuno povezanom neuronskom mrežom bi iziskivao znatno veći broj težina za treniranje. To je jedna od velikih prednosti CNN-a i razlog zašto prednjače u analizi slike i govora [1].

Rezidualne konvolucijske neuronske mreže



Slika 12. Gore: obična CNN, dolje: rezidualna CNN

Kod složenijih klasifikacijskih i detekcijskih problema, dubina neuronske mreže igra bitnu ulogu. Međutim, treniranje nad skupom podataka se tada otežava. S povećanjem dubine neuronske mreže, točnost predviđanja dolazi u zasićenje. U nastojanju da se održi dubina mreže, stvoreni su rezidualni

blokovi koji spajaju svakih nekoliko slojeva mreže što se naziva prečac (engl. *shortcut*, *skip connection*). Primjerice, ako se prečac napravi s n -tim slojem koji povezuje $(l+2)$ -sloj, izraz za izlaz iz $(l+2)$ -sloja bi izgledao kao:

$$a^{[l+2]} = g(W^{[l+2]} \cdot a^{[l+1]} + b^{[l+2]} + a^{[l]})$$

gdje je $a^{[l]}$ prečac doveden do $(l+2)$ -sloja. S ovakvom konstrukcijom ne može se naškoditi modelu, budući da neuronska mreža može lako naučiti funkciju identiteta:

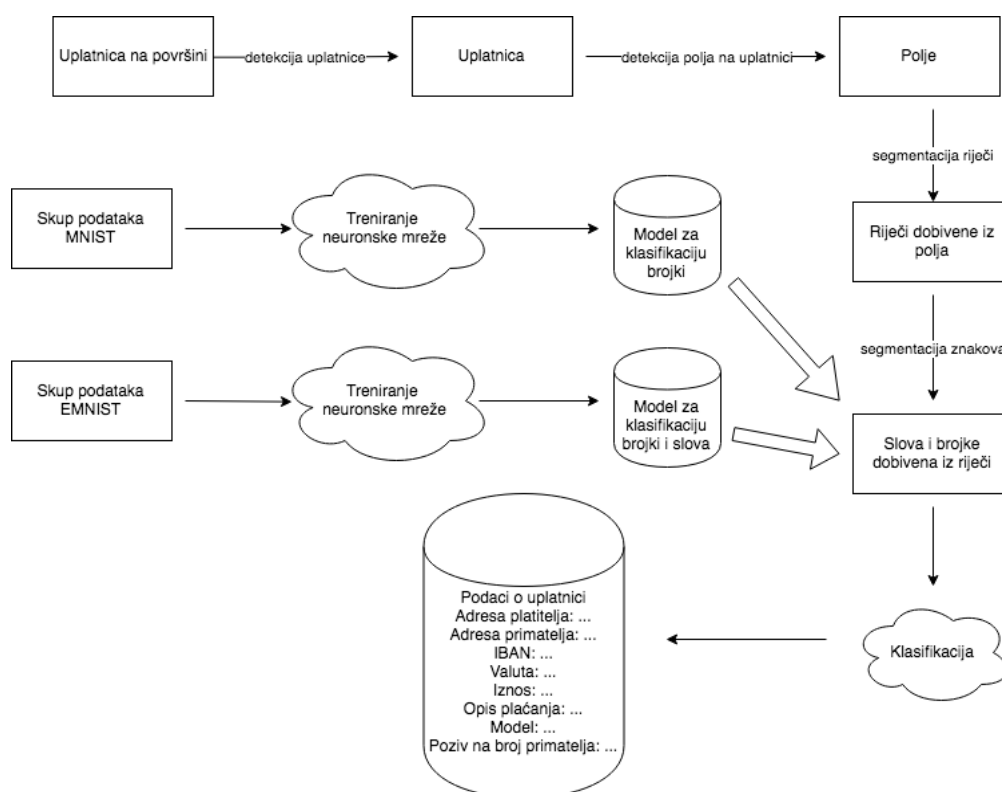
$$a^{[l+2]} = g(0 \cdot a^{[l+1]} + 0 + a^{[l]}) = g(a^{[l]}) = a^{[l]} \text{ (za ReLU)}$$

U slučaju da je aktivacijska funkcija ReLU, izraz $a^{[l]}$ je već pozitivan pa djelovanje aktivacijske funkcije ne mijenja rezultat.

Takva konstrukcija nam može samo pomoći, a u velikom broju slučajeva će nam i pomoći u treniranju i uspješnije klasificirati slike[4].

4. Identifikacija popunjenih polja na uplatnici i rezultati

Prepoznavanje rukom pisanog teksta na uplatnicama se odvija u više faza. Prije same obrade uplatnica i postupka prepoznavanja potrebno je naučiti neuronsku mrežu na poznatom skupu podataka slova i brojeva. Zatim slijede postupci obrade slike uplatnice te konačno prepoznavanje slova i brojeva te ekstrahiranje same informacije. Na slici (Slika 13) je prikazan okvirni dijagram tijeka za analizu teksta na uplatnici. Svaki korak obrade uplatnice te pripremanja mreže za klasifikaciju objašnjeni su kasnije.



Slika 13. Dijagram tijeka u identifikaciji teksta na uplatnicama

4.1. Skup podataka

MNIST skup podataka

MNIST skup podataka[11] obuhvaća zbirku crno-bijelih slika rukom pisanih jednoznamenkastih brojeva. Dimenzija su 28x28 s bijelim znakovima na crnoj pozadini. Skup sadrži oko 50 000 primjera za treniranje (engl. *training set*) te oko 10 000 primjera za validaciju (engl. *validation set*). Slika 14 prikazuje kako izgleda dio skupa podataka MNIST. Svaka slika je dodatno povezana s klasom kojom pripada, pa tako ima 10 raspoloživih klasa za klasificiranje novih slika.

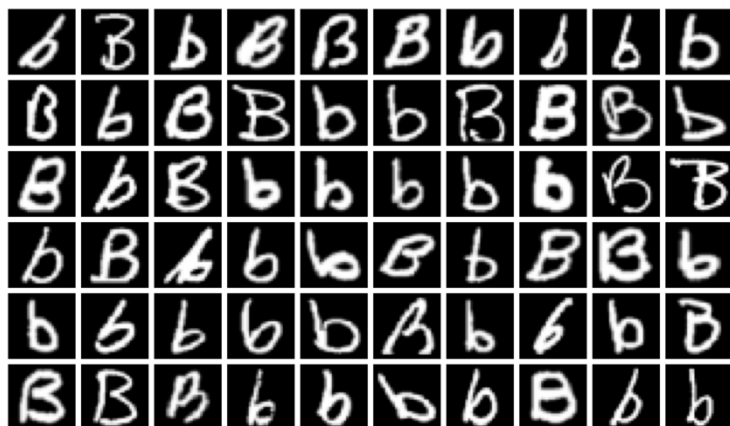


Slika 14. Dio MNIST skupa podataka; podaci su prikazani kao bijeli znakovi na crnoj podlozi

NIST - SD19 skup podataka

NIST - SD19 je jedna od edicija baze NIST koja osim brojki sadrži mala i velika slova engleske abecede. Skupovi podataka su dostupni u različitim verzijama te tako korisnik može preuzeti skup podataka samo sa slovima, samo s brojkama, te kombinirano ovisno jesu li mala ili velika slova. Zbog toga broj klasa ovisi o preuzetom skupu za obradu, pa je tako MNIST baza znakova samo jedan podskup baze NIST. Osim toga, postoje balansirani i

nebalansirani skupovi. Razlika je u tome što balansirani imaju jednak broj slika za svaku klasu. Slika 15 prikazuje dio skupa podataka NIST za slovo B.



Slika 15. Prikaz dijela MNIST skupa podataka za slovo B

4.2. Korištene tehnologije

Za implementaciju identifikacije popunjenih polja na uplatnici razvijen je program u programskom jeziku Python koji koristi različite biblioteke za rad sa strojnim učenjem i obradom slike. Za prepoznavanje slova i njihovo klasificiranje korišten je *framework* Keras[8] te Tensorflow[10] koji podržavaju API za rad s konvolucijskim neuronskim mrežama. Dodatno, za detekciju uplatnice te pripremu teksta na uplatnici za klasifikaciju korištena je biblioteka OpenCV koja nudi razne funkcije za obradu slike, detekciju kontura, eroziju dilataciju i dr[9].

4.3. Treniranje konvolucijske neuronske mreže

Za treniranje konvolucijske neuronske mreže korišten je framework Keras u programskom jeziku Python koji nudi programsku podršku za izradu i treniranje neuronskih mreža. Keras također nudi i mogućnost prikaza grafova s prosječnom točnošću te prosječnom pogreškom. Prosječna točnost se računa kao omjer između točno klasificiranih slika naprema broju svih primjera u skupu, dok se prosječna pogreška izračunava po formuli:

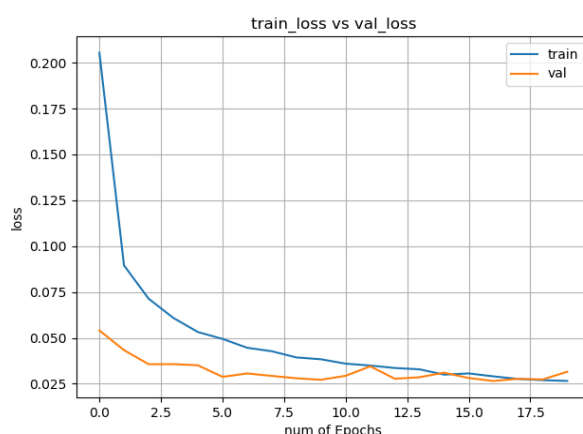
$$E = \frac{1}{N} \sum_{i=0}^N (\vec{o}_i - \vec{t}_i)^2$$

što predstavlja srednju kvadratnu pogrešku između svih trenutno dobivenih izlazima u mreži i željenih izlaza u mreži nad skupom podataka.

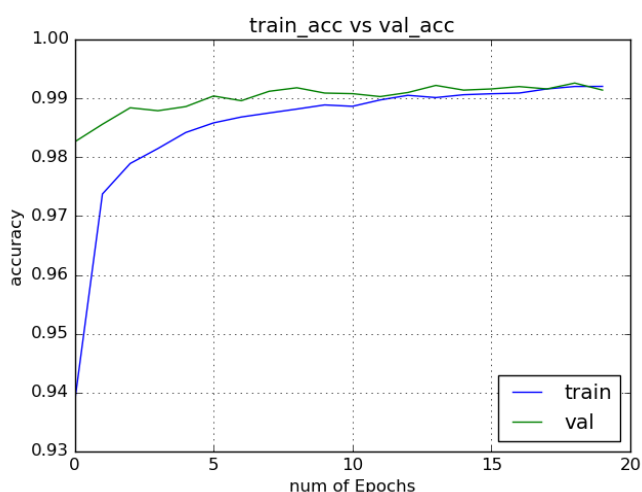
Skup MNIST za treniranje nad brojevnim podacima pokazao se kao iznimno lagan za treniranje, a Keras ima dostupan skup podataka MNIST za korištenje. Korištena je sljedeća arhitektura:

1. **Sloj – konvolucijski sloj**, 32 filtera veličine 3x3, ReLU aktivacija
2. **Sloj – konvolucijski sloj**, 32 filtera veličine 3x3, ReLU aktivacija
3. **Sloj – sloj sažimanja**, filter veličine 2x2
4. **Sloj – potpuno povezani sloj** - 128 neurona, ReLU aktivacija
5. **Sloj – izlazni sloj** – 10 neurona (broj klasa), Softmax aktivacija

Odabrani algoritam treniranja bio je ranije opisani backpropagation, a treniralo se 20 epoha. Na skupu za validaciju model je ostvatio 99.14% točnosti, s prosječnom pogreškom od 3.15%. Na sljedećim slikama (Slika 16, Slika 17) je prikazano kretanje prosječne pogreške te kretanje točnosti za gore navedeni model.



Slika 16. Prikaz prosječne pogreške po epohama za prvu arhitekturu (MNIST)



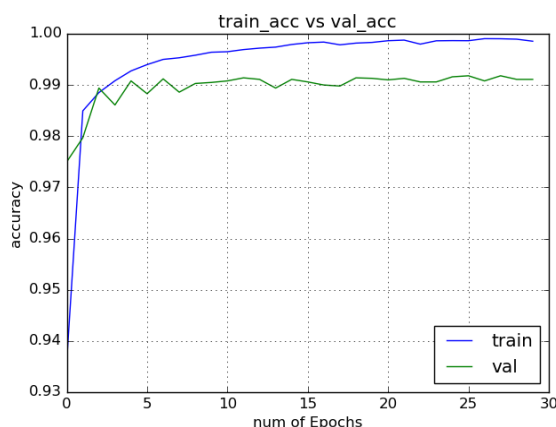
Slika 17. Prikaz točnosti na skupu MNIST za treniranje i validaciju za prvu arhitekturu

Budući da je točnost modela bila prilično visoka ispočetka, radi predostrožnosti izabrana je malo složenija arhitektura:

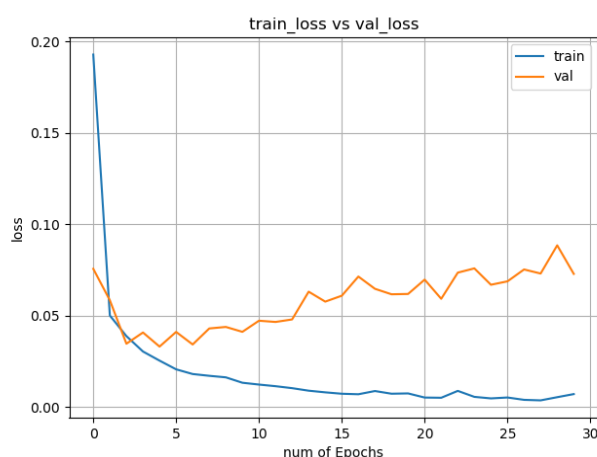
1. **Sloj – konvolucijski sloj**, 64 filtera dimenzija 5x5, ReLU aktivacija
2. **Sloj – sloj sažimanja**, filter dimenzija 2x2
3. **Sloj – konvolucijski sloj**, 64 filtera dimenzija 5x5, ReLU aktivacija
4. **Sloj – sloj sažimanja**, filter dimenzija 2x2

5. **Sloj – konvolucijski sloj**, 32 filtera dimenzija 5x5, ReLu aktivacija
6. **Sloj – potpuno povezani sloj**, 120 neurona, ReLU aktivacija
7. **Sloj – potpuno povezani sloj**, 120 neurona, ReLU aktivacija
8. **Sloj – izlazni sloj**, 10 neurona (broj klasa brojeva), Softmax aktivacija

Rezultat je bio identičan u početku ali je zbog većeg broja epoha (ovog puta 30) prosječna pogreška počela rasti dok se točnost na skupu za validaciju nije mijenjala. Pretpostavka o razlogu takvog ponašanja jest ta da je mreža prestala generalizirati klasificiranje slika te počela prenaučavanje na skupu za učenje. Iako točnost na skupu za validaciju nije pala što je čest slučaj kod prenaučavanja, prosječna pogreška se povećala, što znači da su se preostale krivo klasificirane slike počele još lošije klasificirati. Model je ostvario rezultat točnosti od 99.11% s prosječnom pogreškom od 7.2%, što su lošiji rezultati nego li je ostvario prijašnji model. Na slikama (Slika 18, Slika 19) je prikazano kretanje prosječne pogreške modela te točnosti. Zbog toga se za klasifikaciju brojeva na uplatnici ipak odabrao prvi model. Za treniranje se eksperimentiralo s par modela, ali ova 2 su najreprezentativnija što se tiče cijele faze treniranja MNIST skupa podataka.

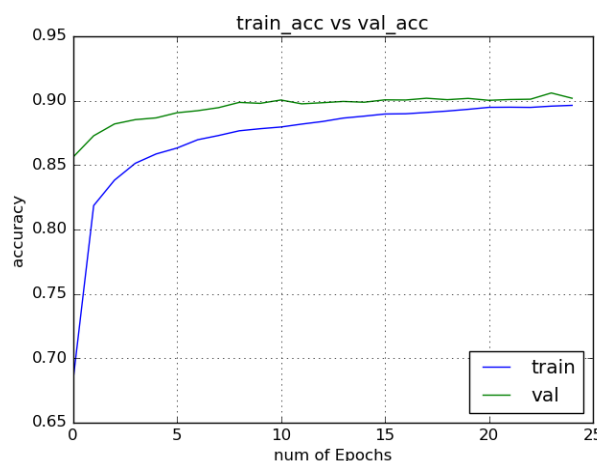


Slika 18. Prikaz točnosti na skupu MNIST za treniranje i validaciju za drugu arhitekturu

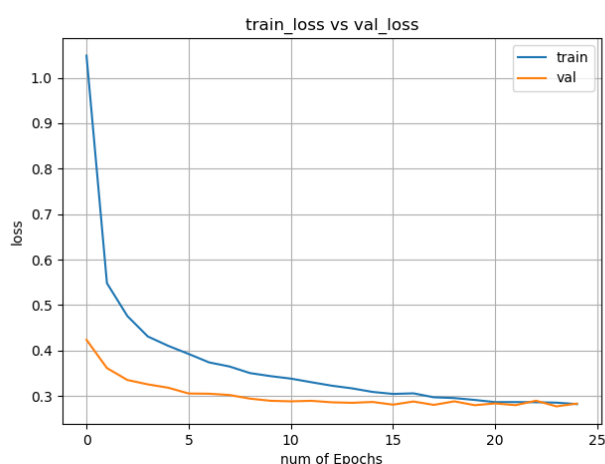


Slika 19. Prikaz prosječne pogreške po epohama za drugu arhitekturu (MNIST)

Kod EMNIST skupa koristio se balansirani skup podataka s 36 klasa (10 za brojeve i 26 za slova engleske abecede). Svaka klasa sadržavala je 2400 primjera za treniranje i 400 za validaciju, ukupno 86400 primjera za treniranje i 14 400 za validaciju. Nakon par ispitivanja izabrana je arhitektura kao za MNIST skup podataka te je pokazala slične rezultate. Model je imao točnost od 90.14% te prosječnu pogrešku oko 28%. To je razumno za očekivati zbog puno većeg broja klasa u izlaznom dijelu. Na slikama (Slika 20, Slika 21) je prikazano kretanje točnosti i prosječne pogreške za skup EMNIST.

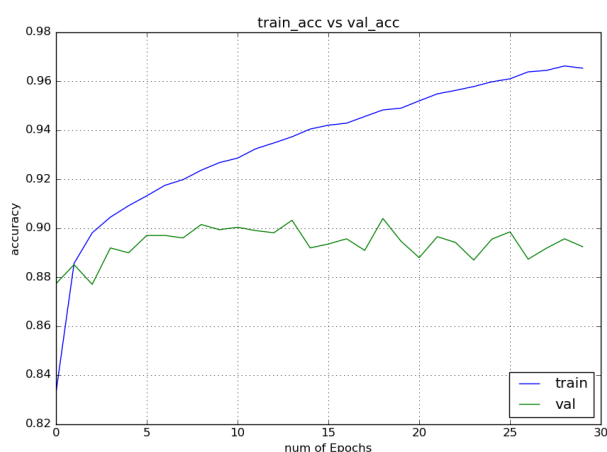


Slika 20. Prikaz točnosti na skupu EMNIST za treniranje i validaciju za prvu arhitekturu

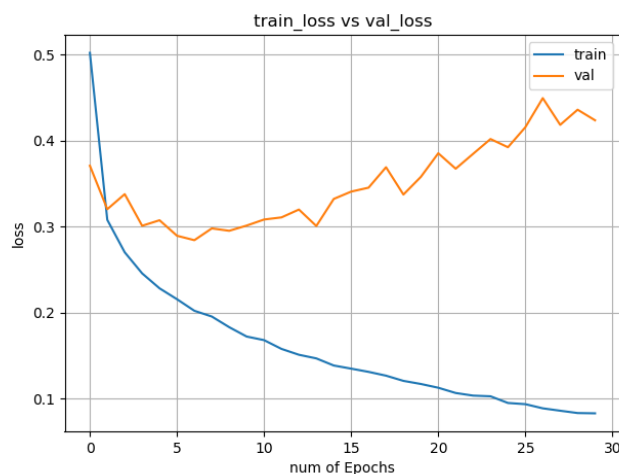


Slika 21. Prikaz prosječne pogreške po epohana za drugu arhitekturu (EMNIST)

EMNIST skup je testiran i na drugoj arhitekturi a rezultati su bili slični kao i sa MNIST skupom. Zbog mogućeg prenaučavanja mreže, izabran je prvi model kao klasifikator brojki i slova. Na slici su prikazani rezultati s drugom arhitekturom i 30 epoha. Model je ostvario rezultat od 89.58% točnosti s prosječnom pogreškom od oko 43% što je dosta lošije u odnosu na prvi model. Na slikama (Slika 22, Slika 23) je prikazan graf kretanja prosječne točnosti te pogreške za navedeni model.



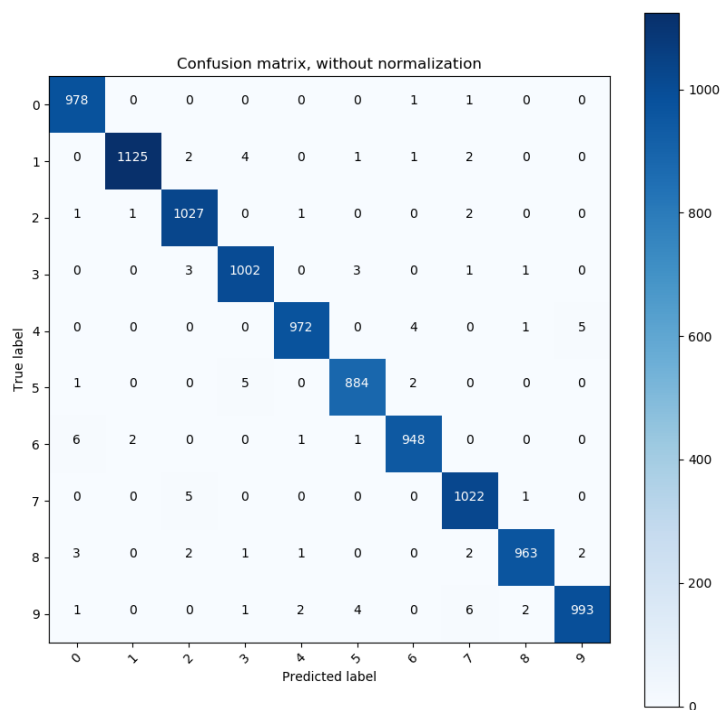
Slika 22. Prikaz točnosti na skupu EMNIST za treniranje i validaciju; druga arhitektura



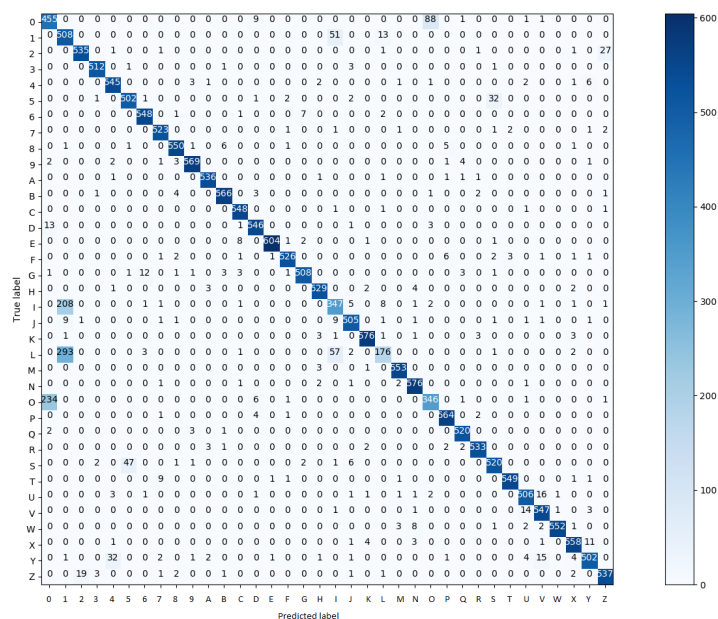
Slika 23. Kretanje prosječne pogreške po epohama za drugu arhitekturu (EMNIST)

Iako je druga arhitektura pokazala izvjesno lošiji rezultat, veću ulogu u tome je imao broj epoha koji je vjerojatno doveo do stagnacije mreže na skupu za validaciju, negoli sama arhitektura mreže.

U nastavku su prikazane matrice zbunjenosti (Slika 24, Slika 25) za skup MNIST te za skup EMNIST. Matrice zbunjenosti su jedan od oblika prikaza točnosti istreniranog modela. Istrenirani model testira se nad nekim skupom podataka te se njegove predikcije zapisuju u tablicu. Na slici vidimo kako je glavna dijagonala matrica najtamnija, odnosno da je model pogodio koja je istinita klasa slike. U svim ostalim slučajevima, model je pogriješio koja je prava klasa slike. Retci označavaju pravu vrijednost klase, dok stupci predviđenu.



Slika 24. Matrica zbunjenosti za izabrani model nad skupom MNIST



Slika 25. Matrica zbunjenosti za izabrani model nad skupom EMNIST

4.4. Pretprocesiranje uplatnice

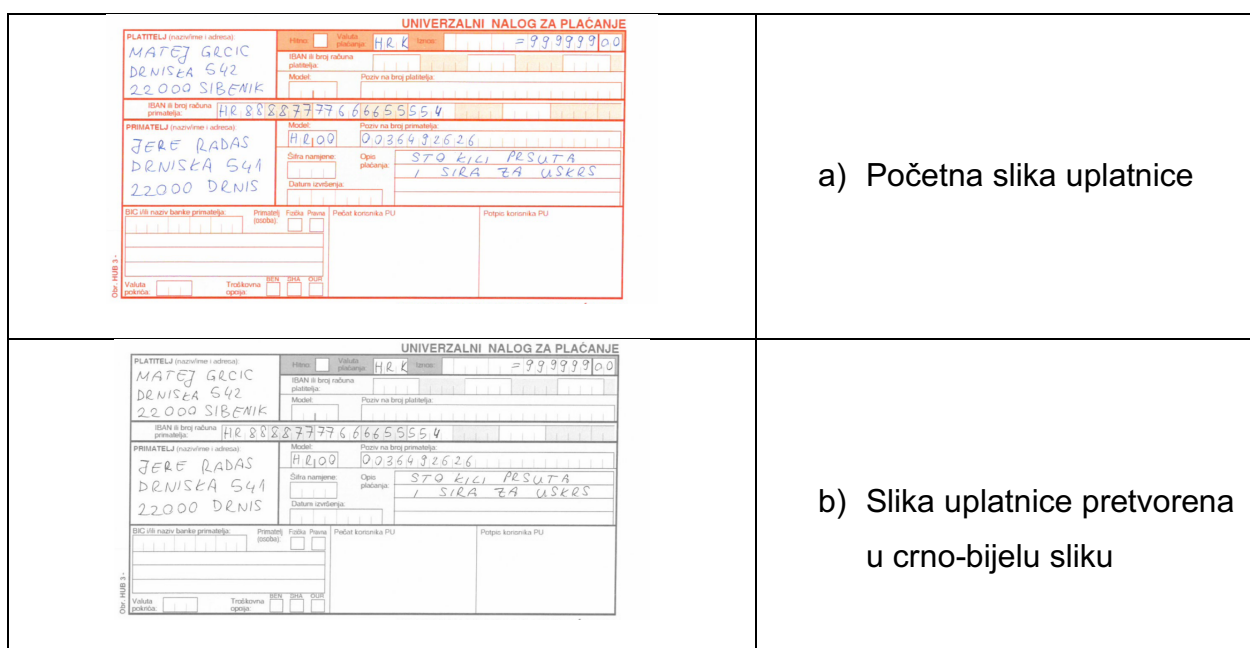
Kako bi podatci bili pripremljeni za klasifikaciju, iz same slike uplatnice trebaju se detektirati slova i brojeke potrebne za ulaz u konvolucijsku neuronsku mrežu[2]. U nastavku je prikazan pseudokod (Slika 26) algoritma pretprocesiranja uplatnice:

```
pripremiSlovaZaKlasifikaciju:
    rjecnikSlova = {}
    polozej = detektiraj_polozej_uplatnice()
    uplatnica = izrezi_uplatnicu(polozej)
    ukloni_crvenu_boju_s_uplatnice(uplatnica)
    popunjena_polja = detektiraj_popunjena_polja(uplatnica)
    za polje iz popunjena_polja:
        rijeci = detektiraj_rijeci(polje)
        za rijec iz rijeci:
            slova = detektiraj_slova(rijec)
            dodaj(rjecnikSlova, polje, slova)
```

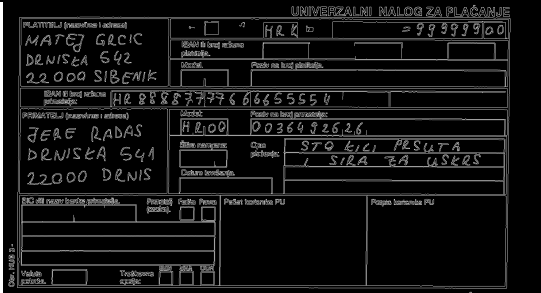
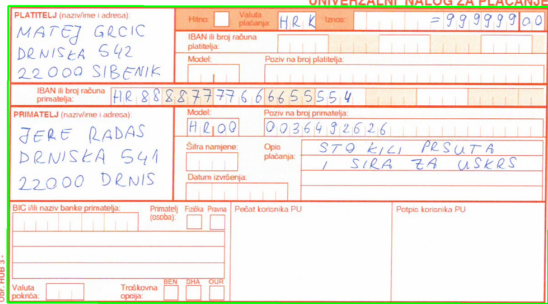
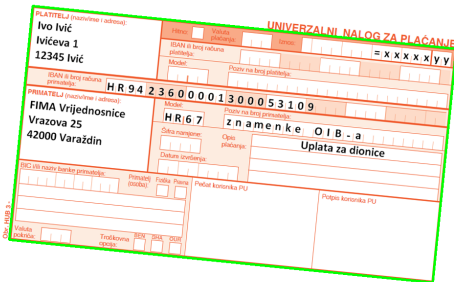
Slika 26. Pseudokod za fazu detekcije slova na uplatnici

4.4.1. Detekcija uplatnice na površini

Prvi korak u procesu pripreme uplatnice za klasifikaciju teksta je detekcija uplatnice na površini. U svrhu toga OpenCV je poslužio kao pomoćna biblioteka, a cilj je pronaći najveći pravokutnik na slici, što bi bio obrub same uplatnice. Sliku treba dodatno rotirati ako nije u pravilnom položaju. Pravilna detekcija uplatnice uhvatila bi vanjski crveni obrub te ga izrezala, ostavivši tako samu uplatnicu na slici. Izrezivanje uplatnice provodi se u nekoliko faza. Zbog jednostavnosti se prvo provodi pretvaranje u crno-bijelu sliku. Zatim se detektiraju obrisi slike i rubovi, te se nakon toga pronalaze najveće konture na slici. Na slikama (Slika 27) su redom prikazani međukoraci u detekciji uplatnice. Na zadnjoj slici prikazan je omeđujući pravokutnik (engl. *bounding box*) uplatnice koja bi se nakon toga izrezala (engl. *crop*).



4. Identifikacija popunjenih polja na uplatnici i rezultati

	c) Rezultat nakon traženja rubova na uplatnici
	d) Konačna slika uplatnice s omeđujućim pravokutnikom
	e) Rotirana detektirana uplatnica

Slika 27. Tablica s postupkom detekcije uplatnice na površini

Rezultati detekcije uplatnice su bili ispravni za svaku uplatnicu. Algoritam je uspio uvijek odrezati odgovarajući dio i pripremiti sliku za sljedeći korak detekcije polja.

4.4.2. Detektiranje polja na uplatnici

Budući da na slici ostaje samo uplatnica, dimenzioniranjem na fiksnu predefiniranu veličinu te izrezivanjem dijelova mogu se dobiti pojedina polja na uplatnici. Uplatnica ima jedinstven oblik tako da se sa sigurnošću slika može točno izrezati. Zbog jednostavnosti je odabrano nekoliko polja koji će se izrezati i proslijediti na daljnju detekciju te naposljetku na klasifikaciju. To su sljedeća polja:

- adresa platitelja
- adresa platitelja
- valuta
- iznos u kunama i lipama
- iban primatelja
- opis plaćanja
- poziv na broj primatelja
- model

Na slikama (Slika 28) su prikazana izrezana polja uplatnice. Dodatno je za znakove upotrijebljena Otsuova metoda. Otsuova metoda sadrži histogram monokromatskih boja te se na temelju sličnosti trenutnog piksela s nekom bojom iz histograma odlučuje njegova boja u konačnoj slici. U ovom završnom radu korišten je bimodalni Otsuov histogram sa 2 boje – crnom i bijelom. Željeni rezultat je bijeli tekst na crnoj pozadini – onakav kao i skup podataka u bazama MNIST i EMNIST.



Slika 28. Ekstrahirana polja uplatnice uz primjenu Otsuove metode

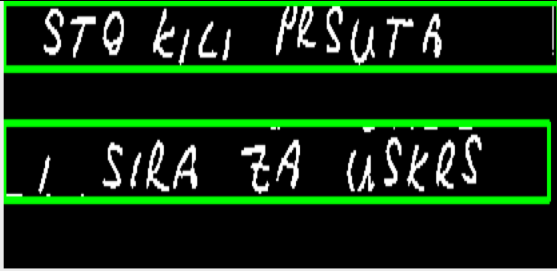
4.4.3. Detekcija linija, riječi i slova iz polja uplatnice

Sljedeći korak jest segmentacija dijelova riječi i slova. Kod polja adrese primatelja, platitelja te opisa, potrebno je prvo napraviti segmentaciju linija, zatim segmentaciju riječi te naposljetku za svaku riječ detektirati slova pojedine riječi. Sličan postupak proveo se i za ostala polja na uplatnici. OpenCV podržava funkcije za dilataciju i eroziju kod slike. Dilatacija zadebljava rubove na slici dok erozija radi obrnuto odnosno stanjuje rubove piksela. Željeni učinak koji se nastoji dobiti jest spojiti piksele u redu za dobivanje bijele mrlje koja predstavlja liniju. Kod riječi bi se koristila dilatacija i erozija u manjoj mjeri, dovoljno da se ne spoji razmak između 2 riječi u jednu. Kod slova nije potrebna dilatacija i erozija. Nakon toga se provodi traženje kontura kao i na uplatnici.

Rezultati su bili prosječni. Kod tiskanih uplatnica algoritam je uspješno izdvojio linije i skoro sve riječi i slova. Kod rukom pisanih uplatnica imao je dosta problema.

Naime, svaki čovjek različito piše (usko ili široko, uspravno ili koso) pa se neka slova unutar riječi mogu protumačiti kao više riječi, dok će se neke sasvim dovoljno prepoznati. Također, kod segmentacije slova zbog kosog pisanja kod nekih ljudi, algoritam će segmentirati više slova u jedno iako to

nije željeni učinak. Na slici (Slika 29) su prikazani rezultati segmentacije linija, riječi i slova.

	a) Dobro izdvojene linije iz polja Opis na uplatnici
	b) Dobro izdvojene riječi i slova iz uplatnice
	c) Loše izdvojene riječi i slova iz uplatnice

Slika 29. Tablica s prikazom izdvajanja linija, riječi i slova iz uplatnice

4.5. Prepoznavanje teksta na uplatnici

Nakon ekstrahiranja samog teksta, potrebno je klasificirati slova pomoću istreniranih modela konvolucijske neuronske mreže. Slova se prilagođavaju na dimenzije 28x28 te se centriraju uz punjenje rubom od 3 piksela. To je format kojim su pisani i MNIST znakovi i EMNIST znakovi, a uvelike pomaže modelu kod klasifikacije.

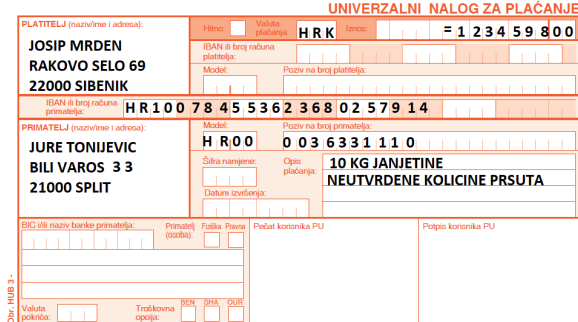
Rezultati konačne klasifikacije teksta na uplatnicama bili su zadovoljavajući budući da detekcija riječi i slova nije prošla najbolje. Kod tiskanih uplatnica model je jako dobro prepoznavao riječi i brojke (kako bi i

trebao, jer je to najjednostavniji slučaj), dok se kod rukom pisanih uplatnica vidjela greška segmentacije riječi i slova.

Unatoč tome, kod dijelova za IBAN, iznos i poziv na broj primatelja gdje je korišten MNIST model rezultati su bili dobri zbog toga što se svaki broj unosi između pregrada te je ostalo dovoljno mjesta da se slova ne preklope. S druge strane, adresa primatelja i platitelja su se prepoznavala donekle dobro s problemom segmentacije riječi i slova. Tako se većina slova razdvojilo u posebne riječi uz iznimke gdje je algoritam dobro segmentirao slova.

Nadalje, zbog skeniranja uplatnica vjerojatno je došlo do pojave neželjenog šuma te su nakon Otsuove metode ostale male točkice na ekstrahiranim poljima (vidi sliku 32.). Sve to je utjecalo na lošu segmentaciju linija, riječi i slova.

Na sljedećim slikama (Slika 30) prikazani su rezultati na pojedinim uplatnicama.

	
a) Tiskana uplatnica (sanity check)	
<pre> ADRESA PLATITELJA: JOSIP MRDEN RAKOVO SELO 59 ZZO00 SIBENIK ADRESA PRIMATELJA: JURE TONIJEVIC BIL1 VAROS 3 3 ZZO00 SPL1T IBAN PRIMATELJA: HR100784553623680257914 POZIV NA BROJ PRIMATELJA: 0036331110 MODEL: HR00 VALUTA: HRK IZNOS: =2234598,00 OPIS: 10 KG JANJETINE NEUTVRDENE KOLICINE PRSUTA </pre>	
b) Rezultat prepoznavanja na tiskanoj uplatnici	

4. Identifikacija popunjenih polja na uplatnici i rezultati

UNIVERZALNI NALOG ZA PLAĆANJE	
PLATITELJ (naziv i adres) JOSIP JUKIC AVENIJA KRALJA JOSIPA 1	Plaćanje Valuta plaćanja: VSS Iznos: =1221,00 IBAN i broj računa platitelja: HR99 Model: Poziv na broj platitelja: 0
PRIMATELJ (naziv i adres) EUGENE DOGGOWSKI DOG KETTLE 4	Plaćanje Model: HR99 Poziv na broj primatelja: 1 Sifra namjene: OPIS: TRDSKOVI PSECE HRANE Datum izdavanja: DIDA CUSPAJE DZINGISKAN Datum izvršenja: 13021983
BIC ili naziv banke primatelja: Primatelj (osoba): Potpis primatelja:	Potpis korisnika PU: Potpis korisnika PU:
Valuta plaćanja: VSS Tržišna vrijednost: BEN, SEK, EUR	

c) Rukom pisana uplatnica (1)

ADRESA PLATITELJA: J O S P J U K I C A V E N I J A K R A L J A J O S I P A A
 ADRESA PRIMATELJA: E U G E N E D O G G O W S K I D O G K E T T L E 4
 IBAN PRIMATELJA: HR
 POZIV NA BROJ PRIMATELJA: 221
 MODEL: HR9Y
 VALUTA: VSS
 IZNOS: =1221,00
 OPIS: TR05TWV1 PSECE HRANE T D1 DA CU SPA JZ DZ1N6 S K A N

d) Rezultat prepoznavanja na rukom pisanoj uplatnici (1)

UNIVERZALNI NALOG ZA PLAĆANJE	
PLATITELJ (naziv i adres) MATEJ GRIC DENISKA 542 22000 SIBENIK	Plaćanje Valuta plaćanja: HRK Iznos: =99999,00 IBAN i broj računa platitelja: HR88887777666655554 Model: Poziv na broj platitelja: 0036492626
PRIMATELJ (naziv i adres) JERE RADAS DRNISKA 541 22000 DRNIS	Plaćanje Model: HC00 Poziv na broj primatelja: 0036492626 Sifra namjene: OPIS: STQ KIC PRESUTA Datum izdavanja: 1 SIRA ZA VASKES
BIC ili naziv banke primatelja: Primatelj (osoba): Potpis primatelja:	Potpis korisnika PU: Potpis korisnika PU:
Valuta plaćanja: HRK Tržišna vrijednost: BEN, SEK, EUR	

e) Rukom pisana uplatnica(2)

ADRESA PLATITELJA: M A 1M CFZ G 2 C 1C DR N 1S EA 5 L1 2 22 0 00 S BEN1 K
 ADRESA PRIMATELJA: 2 E R E RA DAS D R N1 S K A 5 4 1 220 00 D R N S
 IBAN PRIMATELJA: HR88887776666655554
 POZIV NA BROJ PRIMATELJA: 0036492626
 MODEL: HC00
 VALUTA: HRK
 IZNOS: =999999,00
 OPIS: S T Q E 1 C1 P R S U T A T 1 S 1 R A Z A 1 A S K T R T S

f) Rezultat prepoznavanja na rukom pisanoj uplatnici (2)

Slika 30. Rezultati detekcije i prepoznavanja testa na odabranim uplatnicama

5. Zaključak

Prepoznavanje teksta danas je vrlo istražen problem s pouzdanim skupom podataka za učenje koji se pokazao izvrsnim kod klasifikacije pisanog teksta na uplatnicama u ovom završnom radu. To se pokazala kao glavna prednost u samoj implementaciji, potvrđujući time da su konvolucijske neuronske mreže vjerodostojan klasifikator slika u području računalnog vida. Nasuprot tomu, egzaktna detekcija uplatnice, linija, riječi i slova pokazala se kao vrlo labilna komponenta u ostvarenju programskog proizvoda te zbog koje je većinom dolazilo do krive detekcije (te naposljetku i klasifikacije).

Za proširenje rada na ovom programskom proizvodu izdvojila bi se preinaka komponente za detekciju te odvajanje modela za klasifikaciju od ostalih komponenti, koji može služiti i za klasifikaciju teksta u nekim drugim proizvodima. Što se tiče izdvojenog teksta nakon prepoznavanja, moduće je proširenje u vidu raznih prilagodnika (engl. *wrapper*, *adapter*), primjerice za unos podataka u sustav bankarskih aplikacija i slično.

Ovaj proizvod samo je mali dio onoga što područje računalnog vida nudi u smislu aplikativnih rješenja za olakšanje života čovjeku. Na ljudskoj mašti je da primijeni znanje tog područja u daljnjim izumima i istraživanjima.

6. Literatura

- [1]. Yann LeCun, Leon Bottou, Yoshua Bengio, Patrick Haffner: „Gradient-Based Learning Applied to Document Recognition“, 1998.
- [2]. Santosh, Dr. Jenila Livingston L.M.: „Text Detection From Documented Image Using Image Segmentation“, 2013.
- [3]. Ian Goodfellow, Yoshua Bengio, Aaron Courville: „Deep Learning“, 2016.
- [4]. Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun: „Deep Residual Learning for Image Recognition“, 2015.
- [5]. R.Rojas: „Neural Networks“, 1996.
- [6]. Marko Čupić: „Prirodom inspirirani optimizacijski algoritmi. Metaheuristike.“, Sveučilište u Zagrebu, kolovoz 2013.
- [7]. D. Lu, Q.Weng: „A survey of image classification methods and techniques for improving classification performance“, 2007.
- [8]. Keras Documentation: <https://keras.io/>
- [9]. OpenCV Documentation: <https://opencv.org/>
- [10]. Tensorflow Documentation: https://www.tensorflow.org/api_docs/
- [11]. MNIST Dataset: <http://yann.lecun.com/exdb/mnist/>
- [12]. Stanford, Convolutional Neural Networks:
<http://cs231n.github.io/convolutional-networks/>

7. Sažetak

Napretkom u računarskoj znanosti te razvojem dubokog učenja, područje računalnog vida je u zadnjem desetljeću dobilo velik zamah. U moru primjena, detekcija i prepoznavanje teksta (bilo tiskanog ili pisanog) je jedno od istraženijih područja te domene. Ovaj završni rad bavi se upravo prepoznavanjem teksta, za potrebu ekstrakcije teksta na rukom pisanim uplatnicama. Za potrebe toga korištene su konvolucijske neuronske mreže za čisto prepoznavanje teksta te elementi pretprocesiranja slika i detekcije pravopisnih elemenata kao dio izvan domene strojnog učenja. Klasifikacija slika u slova i brojeke uz pretprocesiranje pokazala se kao uspješan pothvat s visokom stopom točnosti, dok je detekcija zbog različitog šuma na svakoj uplatnici te jednostavnih detekcijskih metoda bila glavni uzrok nezadovoljivosti dobivenih podataka na određenom broju uplatnica.

Ključne riječi: Prepoznavanje teksta, Konvolucijske neuronske mreže, Duboko učenje, Računalni vid, Segmentacija slika, Detekcija objekata

8. Abstract

With the development of computer science (particularly deep learning) in the recent years, the computer vision gained a big swing of advancement. In the sea of applied techniques, detection and text recognition (wether printed or handwritten) is one of the most explored fields of that domain. This paper deals with text recognition in the aim of its extraction on handwritten payment slips. For that purpose, convolutional neural methods were used for plain text recognition while the methods for preprocessing of images and detection of the ortographic elements were excluded from machine learning domain. Classification of pictures into numbers and letters with a little preprocessing showed to be a successful method for text recognition with a high rate of accuracy. On the other hand, detection of ortographic elements was the main reason for unsatisfiable results on some of the slips, due to the noise on the cover of each slip and the simplicity of the detection methods used.

Keywords: Optical character recognition, Convolutional neural networks, Deep learning, Computer Vision, Image Segmentation, Object detection