Automatsko prepoznavanje automobilskih registracijskih oznaka

Anto Čabraja
Sveučilište u Zagrebu
Prirodoslovno matematički fakultet
Diplomski smjer: RiM
cabraja.anto@gmail.com

Ante Grbić
Sveučilište u Zagrebu
Prirodoslovno matematički fakultet
Diplomski smjer: RiM
antegrbic8@gmail.com

Apstrakcija – Prepoznavanje automobilski registracijskih oznaka je poznat i dosta dobro obrađen problem sa mnogim uspješnim rješenjima. Problem se sastoji od 2 dijela: detekcije objekta i prepoznavanje pronađenih objekata. Mi smo se ograničili samo na prepoznavanje objekata. Također, bitno je napomenuti da je uspješnost algoritama neusporediva jer većina autora koristi vlasitite baze podataka i analizira drugačije registarske oznake.

I. UVOD

U bilo kojem sistemu koji se bavi prepoznavanjem objekata, postoje 2 veća problema koja se trebaju riješiti: problem detekcije svih objekata traženog oblika i prepoznavanje objekata. U ovom radu fokusirati ćemo se na prepoznavanje objekata jer je detekcija objekata pravokutnih oblika područje računalnog vida. Prepoznavanje objekata je područje pogodno za strojno učenje pa smo se bazirali na njega.

Naš osnovni klasifikacijski problem je sljedeći: pretpostavimo da imamo algoritam koji nam za danu sliku vraća sve pronađene pravokutne oblike na njoj. Želimo odrediti koji su registracijske oznake, a koji nisu. Promatrani klasifikacijski problem je dio šireg (općenitijeg) klasifikacijskog problema da/ne.

Koristili smo umjetnu neuronsku mrežu kao sredstvo ostvarivanja prepoznavanje objekta nakon neuspjelih pokušaja sa linearnim regresijskim modelom i AdaBoost-om. Pri samoj implementaciji nismo razvijali nikakav vlastiti algoritam za detekciju objekata pravokutnih oblika te ćemo kasnije detaljnije opisati način na koji smo dobili skup podataka. Također, koristili smo third-party OCR software koji smo sami dodatno izmijenili kako bismo postigli bolje rezultate. No, ni takav izmijenjeni OCR software nije polučio tražene rezultate pa smo bili primorani na još jednu dodatnu modifikaciju danih podataka, što ćemo kasnije, također, objasniti u detalje.

Dakle, prvo ćemo objasniti skup podataka, odnosno kako smo prikupili podatake is ekstrahirali sve pravokutne oblike te kolika je veličina skupa za učenje, skupa za validaciju i skupa za testiranje. Nakon tog objasniti ćemo značajke koje smo promatrali. Slijedi opis neuronske mreže te rezultate testiranja. Za kraj, kratko ćemo opisati pokušaje sa linearnim regresijskim modelom i AdaBoost-om te dobivene rezultate.

II. SKUP PODATAKA

Skup podataka nismo uspjeli naći online, iako smo mogli pitati neke autore poznatih članaka da nam ustupe svoje podatke. No, odlučili smo se napraviti svoj skup podataka od 150 slika. Slike smo dobili uslikavanjem automobila i ostalih pravokutnih oblika po širem području grada Zagreba.

Skupa za učenje i testiranje konstruirani su od ručno ekstrahiranih registracijskih oznaka sa slika u boji (RGB)s. Skup za učenje čini 70% od ukupno 150 slika, skup za validaciju 15% te skupa za tesiranje preostalih 15%.

Ekstrahirane registarske oznake i ostali pravokutni oblici spremljeni su u bazi podataka gdje uz svaku sliku stoji oznaka "Ispravna" ili "Neispravna".

Potrebno je još napomenuti kako nismo proveli nikakav proces skaliranja i daljnje normalizacije korištenih pravokutnih oblika. Ukoliko bi se netko dalje koristio ovim podacima, odnosno našom bazom podataka bilo bi interestantno promatrati dobivene rezultate sa originalnim podacima i onima koji bi prošli postupak normalizacije.

III. ODABIR ZNAČAJKI

Koristili smo ukupno 12 feature-a (značajki). Pokušali smo odabrati značajke koje će rezultirati sličnim rezultatima na svim registarskim oznakama te dobro razdvojiti (diskriminirati) između registarskih oznaka i pronađenih pravokutnih objekata koji to nisu.

Prve dvije značajke odnose se na broj svijetlih, odnosno tamnih pixel-a na slici. Razlog ovom je što smo primarno promatrali prepoznavanje registarskih oznaka Republike Hrvatske koje su tipično "crno na bijelo". Odredili smo intervale za crvene, zelene i plave pixele koji nam određuju tamne pixele, odnosno svijetle pixele uz zadanu točnost od strane korisnika. Još jedan argument u korist ovom pristupu je činjenica da se slike "prljave", tj. tamo gdje mislimo da je slika potpuno "bijela" ona u biti varira na nekom intervalu vrijednosti.

Sljedeće dvije značajke odnose se ne broj slova i znamenki pronađenih na slici. Za ove značajke koristili smo već gotov OCR software koji se nije pokazao naročito točnim jer je uz slova i znamenke koje reprezentiraju tablicu (npr. ZG – 3523 – JK) pronalazio i ostali tekst na slici, npr. podatke o auto-salonu gdje je automobil kupljen ili ukoliko registarska oznaka označava automobil iz Europske unije bi "pokupio" sitna slova koja označavaju zemlju iz koje navedeni automobil

potječe. No i nakon modifikacije ovog software nismo dobili tražene rezultate pa smo pretpostavili da imamo potpuno točan OCR alat, odnosno u bazi podataka smo ručno unijeli korektne podatke za svaku sliku. Ova značajka se pokazala kao najjače što će se viditi na slikama kasnije i rezultatima.

Sljedeće korištene značajke su broj crnih i bijelih pixela na crno-bijeloj slici. Dobiveni su konverzijom RGB slike u grayscale sliku.

Iduća značajka odnosi se na omjer stranica promatranog objekta. Ova značajka je interesantna jer kod promatranih pravokutnih objekata imamo objekte raznolikih dimenzija, npr. prometnih znakova, znakova za parking te reklamnih oglasa. Ova značajke je low-level feature no ipak se pokazala korisnom.

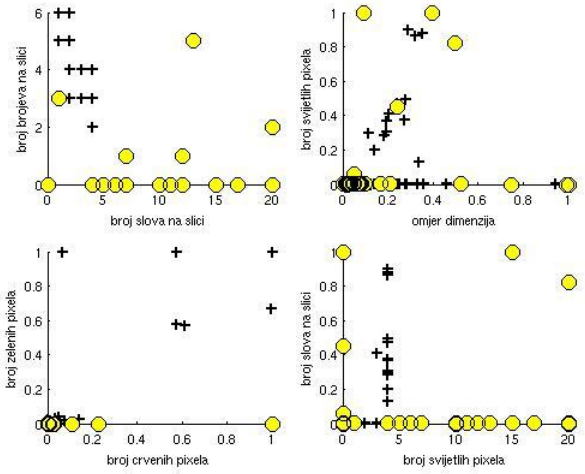
Slijedeći high-levele feature je detekcija kuteva (kod većih slova imamo više detektiranih kuteva).

Također smo uključili i značajke koje označavaju broj pobrojanih nijansi plavog, zelenog i crvenog na ekstrahiranim objektima.

Posljednja značajka je uključena zbog problema koji moramo pretpostaviti kod ekstrahiranja registarskih tablica. Razumno je pretpostaviti da će uz samu registarsku tablicu algoritam pokupiti i dio okolnog područja oko tablice koji nam smeta pri klasifikaciji. U tu svrhu možemo suziti sliku prema sredini (centru) i promatrati omjer bijelo/crno. To je ujedno i posljedna promatrana značajka.

III. NEURONSKA MREŽA

Analizom odabranih značajki, te iz same činjenice složenosti problema može se primjetiti da naš problem nije linearno separabilan. Stoga pristup nekom od jednostavnijih metoda ne bi dao zadovoljavajuće rezultate. Na sljedećoj slici možemo vidjeti nekoliko ekstrahiranih značajki stavljenih u međusoban odnos.



SLIKA 1.Točno klasificirane slike su označene sa x, netočno klasificirane sa o

Naravno nije moguće prikazati 8 ili 10 dimenzionalni prostor, ali i ovakvom reprezentacijom možemo predočiti koliko su zapravo prethodno opisane značajke ne-separabilne u nižim dimenzijama. Upravo iz potrebe za složenijim modelom odlučili smo se za neuronske mreže. Općenito neuronske mreže se koriste kako bi se klasificirali linearno ne-separabilni problemi. Kako bi stvorili dobar klasifikacijski model preko neuronske mreže moramo biti upoznati sa principima i algoritmima koji nam omogućuju ovakav nivo apstrakcije pomoću naizgled jednostavne strukture. Sada ćemo pokušati istovremeno objasniti kako funkcionira neuronska mreža , koji su njeni osnovni djelovi te kako smo mi to iskoristili za stvaranje našeg modela. Dakle ulazni sloj neuronske mreže se sastoji od ulaznih primjera predstavljenih određenim brojem značajki te od target varijable koja predstavlja izlazni rezultat. U našem slučaju taj rezultat je binarni vektor gdje 1 predstavlja da je dotični primjer tablica, a 0 da primjer nije tablica. Ovako definirani ulazni podaci su vrlo česti u praksi i problem se zove OneVsAll klasifikacija. To je klasifikacija gdje nam je jedna skupina primjera značajna i istaknuta a ostale, iako različite, svrstavamo u istu kategoriju. Ovom metodom se često rješavaju i složeniji problemi klasifikacije više skupina objekata gdje onda za N različitih objekata izvršavamo N klasifikacija. U svakoj, od njih N, jedna skupina postaje značajna, a ostale svrstavamo u istu kategoriju. Svaka neuronska mreža se sastoji od perceptrona ili neurona. Kroz jedan neuron cijela mreža dobije novi zaključak na sljedeći način. Za svaki neuron se odrede težine te se one dodjele podacima koji se u tom neuronu računaju. Sam neuron daje jedan ili više izlaznih rezultata. Ti rezultati su trenutna sposobnost neurona i zapravo predstavljaju kako taj neuron u sklopu cijele mreže djeluje na postupak klasifikacije. Matematički gledano imamo sljedeći problem:

$$\sum_i a_i x_i$$

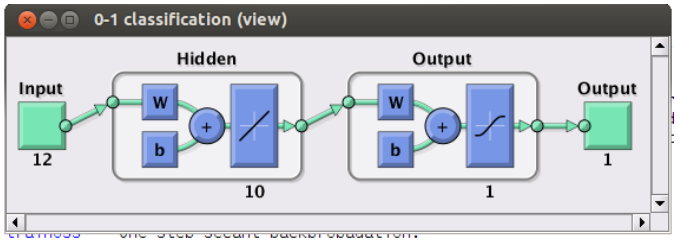
gdje su a_i značajke, a x_i značajke.

Na izlazu iz neurona postoji aktivacijska funkcija. Ovisno o problemu ta je funkcija različita. Samo se zahtijeva da funkcija bude derivabilna pa je tako jedan od primjera aktivacijske funkcije linearna funkcija ili sigmoid funkcije sljedećeg oblika:

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

Pomoću neurona sada možemo izgraditi neuronsku mrežu. Svi slojevi neuronske mreže osim ulaznog i izlaznog sloja zovu se skriveni slojevi. U našem slučaju svi neuroni skrivenog sloja imaju linearnu aktivacijsku funkciju dok izlazni sloj ima sigmoid aktivacijsku funkciju. Razlog je taj što mi i na izlazu želimo binarnu klasifikaciju, tj. da li je podatak kojeg je neuronska mreža klasificirala , slika ili nije. Izlazni podaci u intervalu [0,1] ili [-1,1] te konačnu klasifikaciju dobijemo određivanjem parametra koji dani interval djeli u dva dijela te ovisno o njemu sve lijevo smatramo 0 , a sve desno 1 odnosno tablicom. U principu parametar za dijeljenje intervala se uvijek na početku pretpostavlja kao sredina intervala, no ovisno o podacima i preciznosti, te što ona za nas predstavlja taj parametar može varirati.

Da bi neuronska mreža učila potrebno je propagirati grešku kroz slojeve neuronske mreže i tako za svaki neuron korigirati težine da one što bolje predstavljaju skup za učenje. Jedna od standardnih metoda je backpropagation algoritam (vidi neka referenca). Postoji nekoliko verzija tog algoritma, ali iz različitih literatura kao najpogodniji za naš problem pokazao se Levarberg-Marquard backpropagation algoritam te backpropagation sa gradijentnim spustom. Kako nam okruženje u kojemu smo razvijali neuronsku mrežu pruža podršku za ove algoritme propagacije iste smo samo koristili bez implementacije. U konačnici naša neuronska mreža izgleda:



SLIKA 2.

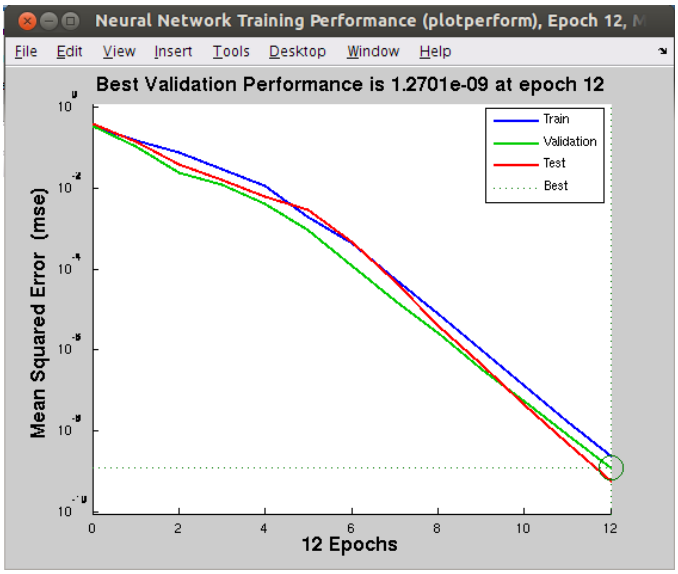
IV. TESTIRANJE

Svrha ovog testiranja je ustanoviti koliko je neuronska mreža zapravo jak klasifikator, te kako podjela podataka na skupove za učenje, validaciju i testiranje utječe na ponašanje našeg klasifikatora. No uz ove osnovne karakteristike koje smo ispitivali, pokušali smo mijenjati i topologiju same mreže. Konkretno, pokušali smo povećati i smanjiti broj skrivenih slojeva. Zaključak do kojeg smo došli takvim testiranjem jest da je najbolje imati između 3 do 7 skrivenih slojeva. Problemi koji su nam se javili kog većeg broja slojeva je prenaučenost modela. Vjerojatni razlog tomu je ipak nedovoljn skup podataka za učenje ili jednostavo je model postao ekspert za trening podatke zbog velike količine težinskih faktora (svaki neuron ima svoje težine). Ako pak broj skrivenih slojeva smanjimo na jako mali broj ,onda naš model postaje prejednostavan za dane podatke.

Greška se računa kao srednja kvadratna vrijednost, odnosno MSE.

Iz nekoliko sljedećih grafova i matrica konfuzije te podataka koji dolaze uz njih došli smo do zaključka da je neuronska mreža dobar klasifikator za naš problem, no da nam je skup za testiranje ipak možda premalen te smo pustili da naša mreža u svakoj instanci pokretanja sama na slučajan način odabere između svih podataka najbolje. Dakle učenje smo pokrenuli određeni broj puta i svaki puta smo na slučajan način dobili trening, test i validacijski skup.

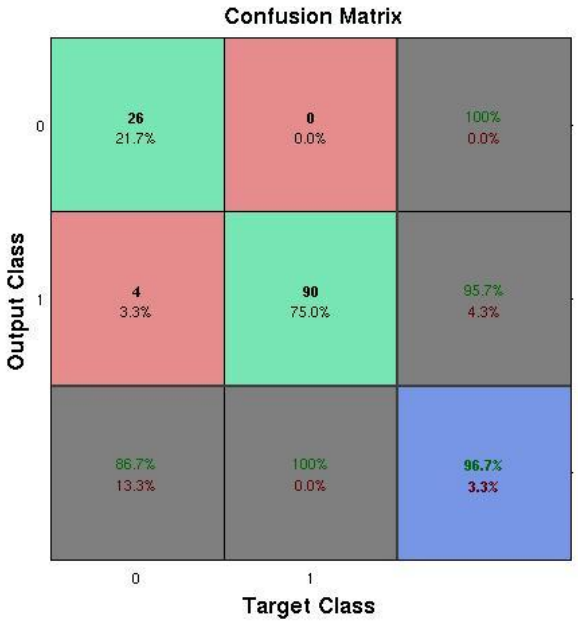
Neke od podataka možete vidjeti na sljedećim slikama. Obratite posebno pažnju na sliku koja opisuje najbolje dobivenu klasifikaciju. Na novih 10 slika (koje smo posljednje uključili u skup podataka točno je klasificirano 9 od 10 slika. U konfuzijskoj matrici nije bilo pogrešno klasificiranih primjera.



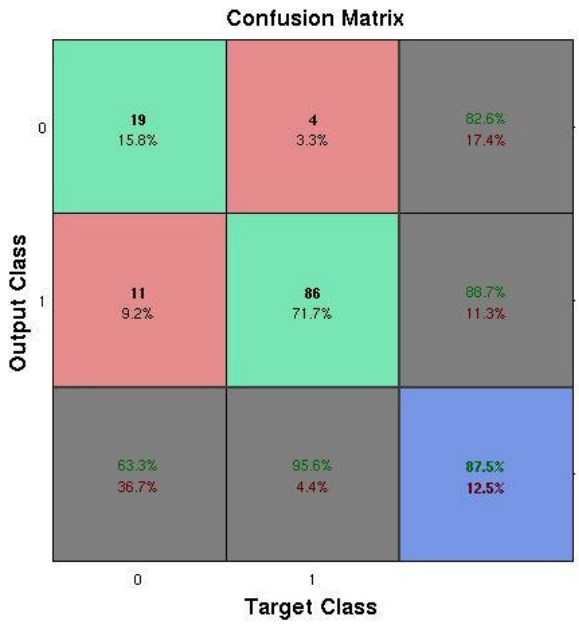
SLIKA 3. Najbolje dobivena klasifikacija

Kada je dobro odabran test za validaciju i učenje kako se greška smanjuje i kako nemamo prenaučenost. Stoga mislimo da bi k-fold cross validacija još bolje povečala ekspresivnost naše mreže te je to jedna od budućih nadogradnji ovo modela.

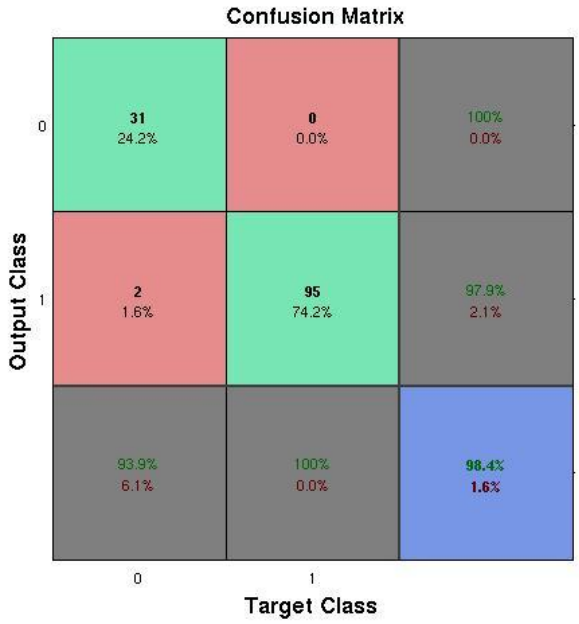
Imamo primjere konfuzijskih matrica 4 različita pokretanja sa slučajno odabranim skupovima za testiranje, validaciju i učenje. Na njima se vidi kako je razlika u rezultatima ako uzmemo drugačije skupove.



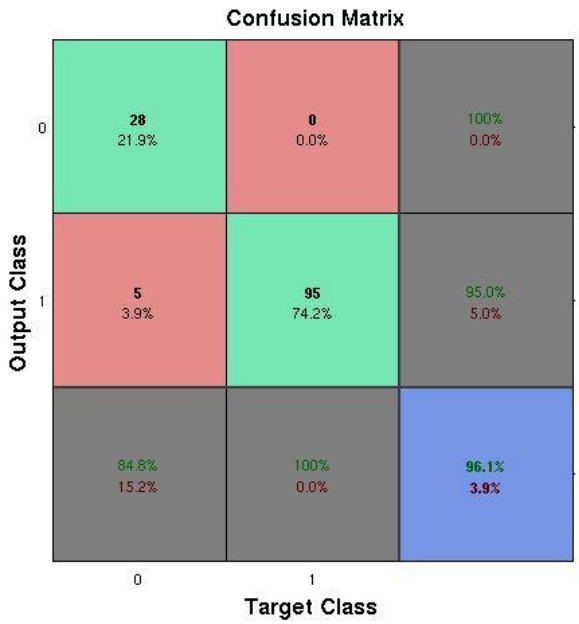
SLIKA 4.



SLIKA 5.



SLIKA 6.



SLIKA 7.

V. LR & ADABOOST

Prvi pristup predstavljenom problemu je bio linearnom regresijom te nakon inicijalnih loših rezultata smo pokušali popraviti stvar korištenjem ansambla, tj. AdaBoost algoritma no ni to nije popravilo rezultate. Linearnu regresiju smo rješavali korištenjem MATLAB-ove funkcije *fminunc* koja rješava minimizacijski problem $\min_x f(x)$ za proizvoljnu funkciju f.

AdaBoost algoritam smo koristili po uzoru na [1]. Bagging principom iz početnog skupa podataka D (l = broj negativnih primjera, m = broj pozitivnih primjera) smo dobili skup podataka D'. Cilj nam je konstruirati L slabih klasifikatora $h_1, \dots, h_L$ da bi dobili jaki klasifikator h. Svakom primjeru pridruzimo njegovu težinu w gdje joj vrijednost dodijeljujemo na način da pozitivni primjeri dobivaju težinu $\frac{1}{2m}$, a negativni $\frac{1}{2l}$. Sada u L koraka treniramo svaki slabi klasifikator j računamo njegovu grešku kao $\sum w_i * |h_j(x_i) - y_i|$ gdje sumiramo po i. Nakon tog odabiremo onaj klasifikator s najmanjom greškom ε_t te modificiramo težine primjera na način da one mijenjamo težine onih primjera koji su uspješno klasificirani na način da je $w_{t+1} = w_t(\varepsilon_t/1 - \varepsilon_t$.

Nakon L rundi, konstruiramo jaki klasifikator h uz pomoć koeficijenata $\alpha_t = \ln(1 - \varepsilon_t/\varepsilon_t)$ na način da za neku vrijednsot x fja h vraća 1 ako je $\sum_{i=1}^{L} \alpha_i h_i(x) \geq \frac{1}{2}\sum_{i=1}^{L} \alpha_i$, a 0 inače.

Ovakvim pristupom imali smo uspješnost klasifikacije od 20-25% .

VI. ZAKLJUČAK

Neuronska mreža se pokazala kao dobar klasifikator za promatrani problem te se nudi moguće daljnje poboljšanje uz cross-validation metodu. Potrebno je tražiti dobar skup na kojemu će model dati najbolje rezultate. Iz slučajno odabranih skupova za učenje, testiranje i validaciju se ispostavilo da odabir jako utječe na rezultate. Stoga smatramo da bi cross-validation metodom postigli da se model nauči na većini kombinacija i tako bi dobili puno bolji rezultat jer očito mala promjena kombinacije mjenja rezultate. Pristup linearnom regresijom i uz pomoć AdaBoost algoritma se nije ispostavio dobrim. Metoda SVM sa kernelima nudi moguće poboljšanje jer su naši podaci slabo separabilni pa bi dizanje u višu dimenziju moglo pomoći pri rješavanju problema.

REFERENCE

[1] L. Dlagnekov, "Licencse Plate Detection using AdaBoost", Department of Computer Science & Engineering, UC San Diego

[2] Raul Rojas, Neural Networks – A Systematic Introduction, Springer-Verlag, Berlin, New York, 1996.

[3] William H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, Numerical Recipes in C: The Art of Scientific Computing, 1992. (second edition)