SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 188

RASPOZNAVANJE UZORAKA PRIMJENOM UMJETNE KOLONIJE PČELA

Dino Klemen

Zagreb, lipanj 2011.

Sadržaj

1. Uvod

Inteligencija roja (engl. *swarm intelligence*) kao grana umjetne inteligencije razvija se na temelju mnogobrojnih inspiracija iz prirode. Ideja je uzeti primjer iz prirode, odabrati ključne komponente zaslužne za njegovu uspješnost te na temelju toga stvoriti model prilagođen računalnom okviru i potrebama naših zadataka. Sličan scenarij dogodio se s pčelama.

U ovom radu se daje kratki teorijski uvod u algoritam umjetne kolonije pčela (engl. *artificial bee colony*) opisan u poglavlju 2. Algoritam se koristi kako bi se istrenirala umjetna neuronska mreža (engl. *artificial neural network*) koja ima zadatak raspoznavati novčanice, odnosno klasificirati ih u odgovarajuću skupinu. U poglavlju 3 napravljen je pregled teorijske pozadine umjetnih neuronskih mreža, dok je u poglavlju 4 formalno opisan problem raspoznavanja novčanica.

Izvedeno programsko rješenje opisano je u poglavlju 5, a naglasak je stavljen na opis skupova podataka korištenih unutar sustava te na proces donošenja odluke o klasifikaciji promatranog primjera.

U poglavlju 6 dani su rezultati provedenih mjerenja o utjecaju parametara ABC algoritma i strukture neuronske mreže na uspješnost i brzinu sustava. Nakon pronalaska optimalnih parametara, provedena su konačna ispitivanja uspješnosti sustava nad ispitnim skupom hrvatskih novčanica.

2. Umjetna kolonija pčela

Umjetna kolonija pčela (engl. *artificial bee colony*) je optimizacijski algoritam koji spada u skupinu pčelinjih algoritama, odnosno inteligenciju roja (engl. *swarm intelligence*).

Bolje razumijevanje pčela do kojeg je došlo u prošlom stoljeću u kombinaciji s uspješnim adaptacijama raznih drugih primjera iz prirode u računalni svijet dovelo je do toga da su i pčele počele dobivati svoje računalne inačice u posljednjih pet do deset godina. Razvijeno je mnogo različitih verzija, ali su sve vođene istom osnovnom idejom.

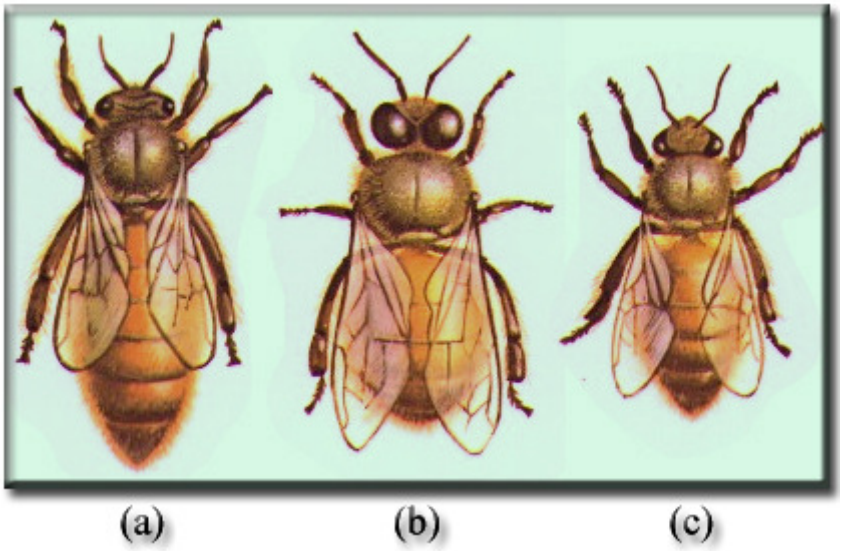
Lučić i Teodorović su 2001. godine na temelju ponašanja pčela uveli sustav pčela (engl. *bee system*) za rješavanje složenih kombinatoričkih problema [1]. Optimizacija kolonijom pčela (engl. *bee colony optimization*) uvedena je 2005. godine kao metaheuristika za rješavanje problema uparivanja putovanja [2]. Iste su godine Drias i Yahi modificirali algoritam u optimizaciju rojem pčela (engl. *bee swarm optimization*) za problem MAX-W-SAT (engl. *maximum weighted satisfiability problem*) [3]. Također 2005. godine Pham uvodi pčelinji algoritam (engl. *bees algorithm*) koji kombinira pretraživanje susjedstva i nasumično pretraživanje [4]. Te iste, 2005. godine Karaboga uvodi algoritam umjetne kolonije pčela (engl. *artificial bee colony*) prvenstveno namijenjen numeričkoj optimizaciji [5]. Upravo se posljednje spomenuti algoritam promatra u ovom radu.

2.1. Inspiracija

Za algoritam umjetne kolonije pčela je, kao i za ostale inačice pčelinjih algoritama, inspiracija pronađena u ponašanju pčela iz prirode. U nastavku se daje kratki pregled ponašanja pčela koji je bitan za kasnije razumijevanje ABC algoritma.

2.1.1. Podjela uloga

Sustav kolonije pčela karakterizira specijalizacija jedinki, podjela rada, simultanost i samo-organizacija. Kolonija pčela se tako sastoji od matice (engl. *queen*), trutova (engl. *drones*) i radilica (engl. *workers*). Podjela se može vidjeti na slici 2.1. Matice su zadužene da liježu jaja, a trutovi za oplodnju matica. Kako stvaranje i uništavanje jedinki u algoritmu nije potrebno, tako se naglasak stavlja na treću vrstu pčela – radilice. Postoje dvije vrste pčela radilica – izviđači (engl. *scout bees*) i skupljači (engl. *forager bees*).



Slika 2.1 Ilustracija tipova pčela: matica (a), trut (b) i radilica (c)

Izviđači se u potrazi za novim izvorima kreću nasumično. Nakon što pronađu izvor, tada se vraćaju u košnicu i pčelinjim plesom promoviraju posjećeni izvor ovisno o njegovoj kvaliteti. Uobičajeno je za kolonije pčela da broj skupljača osjetno nadmašuje broj izviđača. Izviđači se nakon obavljenog plesa vraćaju na svoj izvor u pratnji drugih pčela – ovisno o kvaliteti poruke koju prenose.

Skupljači se ponašaju reaktivno u odnosu na izviđače. Pčele u koloniji promatraju izviđače na prostoru predviđenom za ples. Obzirom na promoviranu kvalitetu izvora one odluče jesu li zainteresirane te pamte upute. Također i zapamte miris izvora (jer je izviđač tamo bio) da znaju kada su došli do cilja. S obzirom na to da se sunce stalno pomiče, pčele imaju i unutarnji smisao za vrijeme pa prilagođavaju informaciju trenutnom stanju od trenutka kada su je primili. Skupljači pri povratku mogu ili nastaviti posjećivati isti izvor ili krenuti u potragu za novim – pregledavanjem ponuda na "plesnom podiju". Sustav prema kojem pčele donose odluke nije do kraja poznat, ali se daje naslutiti da je razina novačenja ovisna o kvaliteti izvora.

2.1.2. Pčelinji ples

Mravi ostavljaju tragove feromona, ptice se pri letu usklađuju sa susjedima, a pčele – plešu. Pčele komuniciraju plesom (engl. *waggle dance*) čije je značenje dešifrirao Karl von Frisch [6]. Ples pčele se odvija u dvije faze – migoljenje (engl. *waggle*) i kružni povratak (engl. *return*). Kružni povratak se odvija naizmjence – prvo u jednu stranu, a zatim u drugu, što ukupno tvori kretanje u obliku osmice. Upravo je pokazano da je faza migoljenja ključna za prijenos informacija drugim pčelama.

Pčele pomoću svog plesa kodiraju tri ključna podatka:

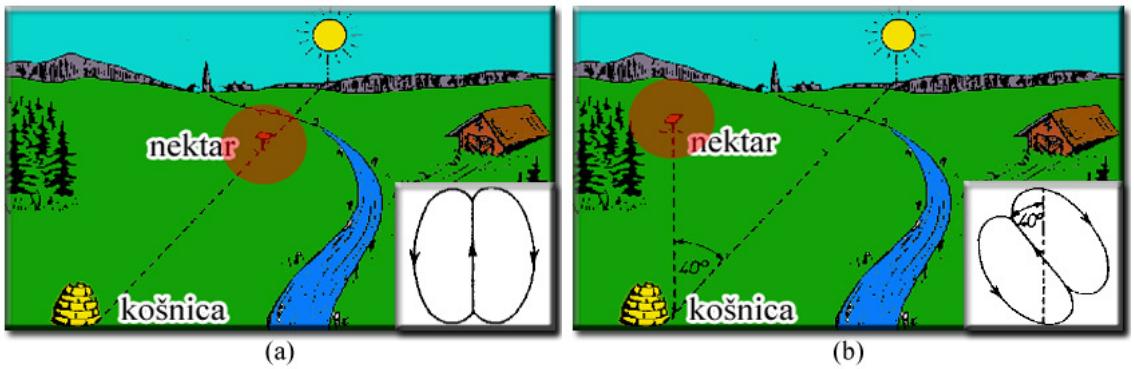
- Smjer izvora
- Udaljenost izvora
- Kvaliteta izvora

Tako svaka pčela može imati saznanja o vanjskom svijetu čak ni ako nikada nije napustila košnicu. Pčele time kombiniraju svoja saznanja s tuđim spoznajama te donose vlastite odluke o svom sljedećem potezu.

Smjer izvora – predstavlja najsloženije dizajniran podatak. Naime, pčele koriste gravitaciju i položaj sunca. S obzirom na to da se ples odvija na vertikalnoj površini košnice, pčele mogu prenijeti informaciju o smjeru hrane u odnosu na smjer sunca. Konvencija među pčelama je da smjer suprotan od gravitacije (prema gore) u vertikalnoj ravnini (košnica) odgovara smjeru prema suncu u horizontalnoj ravnini (vanjski svijet). Bilo koji smjer po horizontalnoj ravnini se tako može kodirati u vertikalnoj jednakim kutovima. Uistinu, istraživanjem se pokazalo da pčele koje se gibaju pod kutom od na primjer 40° suprotno od smjera kazaljke na satu u odnosu prema nebu (suprotno od gravitacije) promoviraju izvor hrane koji je 40° suprotno od smjera kazaljke na satu u odnosu prema suncu. Na slici 2.2 može se vidjeti kako bi pčele svojim plesom signalizirale spomenute smjerove [6].

Udaljenost izvora – može se komunicirati pomoću brzine faze migoljenja odnosno frekvencije krivulje kojom prolaze u središnjoj fazi. Udaljenost izvora se iskazuje u odnosu na košnicu gdje se izvodi ples. Veće brzine ukazuju na bliže, a time i zanimljivije izvore hrane.

Kvaliteta izvora - pčela koja je posjetila neki izvor hrane (cvijet) tamo je sakupljala nektar pa ima informaciju koje je kvalitete te koliko ga je dostupno. Tu informaciju prenosi drugima putem brzine kružne faze i trajanja plesa. Indirektno druge pčele također mogu dobiti indikaciju o kvaliteti izvora obzirom na broj pčela koje promoviraju isti izvor. Kvalitetniji izvor oglašava veći broj pčela i time prikupljaju mnogobrojniju pratnju.



Slika 2.2 Pčelinji ples za upute prema suncu (a) i 40° suprotno od smjera kazaljke na satu u odnosu na sunce (b)

2.2. Algoritam

Ključni mehanizam pčelinjih algoritma leži u dobrom omjeru istraživanja (engl. *exploration*) i eksploatacije (engl. *exploitation*). Pčele izviđači su zadužene za fazu istraživanja, a pčele skupljači za eksploataciju. Osim toga, algoritam je vrlo fleksibilan i lako se prilagođava na dinamički promjenjive probleme te se uspješno bori sa savladavanjem lokalnih ekstrema u potrazi za globalnim.

Umjetna kolonija pčela dijeli se na tri skupine: zaposlene pčele (engl. *employed bees*), promatrači (engl. *onlookers*) te izviđači (engl. *scouts*). Zaposlena pčela istražuje okolinu izvora koji je prethodno posjetila. Promatrači u prenesenom značenju promatraju pčele na plesnom podiju koje izvode pčelinji ples te donose odluku koju će pčelu pratiti. Dok izviđači provode nasumično pretraživanje prostora.

Kolonija se većinom sastoji od zaposlenih pčela i promatrača, u sličnom omjeru. Izviđačko ponašanje nastaje ili prilikom inicijalizacije ili prilikom "potrošnje" izvora. Algoritam se provodi u petlji s ponavljanjem tri osnovna koraka – svaki korak je zadužen za svoju vrstu pčela. Prvo zaposlene pčele mjere kvalitetu svojih rješenja. Zatim tu informaciju dijele s promatračima. Promatrači dalje posjećuju okolinu izvora za koju su se odlučili. Iteracija se zatvara fazom izviđanja do koje dolazi samo ako se napušta postojeće rješenje.

Preslikano na ABC algoritam, lokacija izvora nektara predstavlja moguće rješenje optimizacijskog problema, a količina nektara je ekvivalentna kvaliteti promatranog rješenja problema. Oznaka N se uvodi za veličinu populacije pčela, od kojih svaka generira svoje rješenje x_i (i = 1, 2, ..., N) gdje je x_i D-dimenzionalni vektor. Veličina dimenzije D jednaka je broju parametara koje treba optimizirati. Dodatno se uvodi oznaka iteracija kroz algoritam C = 1, 2, ... C_{max}.

2.2.1. Zaposlene pčele

Pčele u prirodi koje su vezane uz neki izvor ujedno i promatraju izravnu okolinu tog izvora u nadi da će pronaći kvalitetniji izvor od trenutnog. Taj se proces preslikava na računalo ne putem usporedbe cijele okoline, već nasumičnim generiranjem rješenja u odnosu na trenutno. Ako je novo generirano rješenje kvalitetnije od starog, tada umjetna pčela pamti novi izvor. U suprotnom ostaje vezana za stari izvor.

U nastavku je dana osnovna formula pčelinjeg algoritma koju osim zaposlenih pčela koriste i promatrači. Tom formulom se na temelju starog rješenja (vektora x_i) generira novo rješenje (vektor v_i) za sve pčele:

$$v_{ij} = x_{ij} + rand(-1,1) * (x_{ij} - x_{kj}), \tag{2.1}$$

gdje su k ∈ {1, 2, ..., N} i j ∈ {1, 2, ..., D} nasumično odabrani indeksi. Pohlepnom metodom se pamti samo bolji vektor između x_i i v_i.

2.2.2. Promatrači

U nastavku pčele koje nisu zaposlene promatraju rezultate zaposlenih pčela. Logično je da pčele koje nose informaciju o kvalitetnijim rješenjima privlače veći broj "sljedbenika". Taj se princip ostvaruje metodom jednostavne selekcije (engl. *roulette wheel selection*):

$$P_i = \frac{fit_i}{\sum_{i=1}^{N} fit_i}, \tag{2.2}$$

gdje P_i označava vjerojatnost da će pčela prihvatiti rješenje "i" kao svoje. Promatrači nakon odluke koje rješenje prihvatiti provode isti postupak kao zaposlene pčele (formula 2.1).

2.2.3. Izviđači

Izviđači (osim kod inicijalizacije) čine manjinu u koloniji, u odnosu na druge dvije skupine. Potreba za izviđačima se javlja tek kada se odbaci neko rješenje pa je potrebno pronaći novo, neovisno o prethodnim rješenjima. Tada se uvodi novi izviđač koji pronalazi novo rješenje unutar zadanih granica – vektor d za donju i vektor g za gornju granicu:

$$x_{ij} = d_j + rand(0,1) * (g_j - d_j) \tag{2.3}$$

Istom se formulom prilikom inicijalizacije algoritma generira početna populacija od N rješenja.

2.2.4. Pseudokod

Kombiniranjem prethodno opisanih ponašanja dobiva se ABC algoritam koji objedinjuje četiri vrste selekcijskih procesa: lokalna selekcija (formula 2.1), globalna selekcija (formula 2.2), nasumična selekcija (formula 2.3) te pohlepna selekcija koja se očituje pohlepnim biranjem između starog i novog rješenja.

Prethodno opisanim postupkom može se ostvariti pseudokod koji je prikazan na slici 2.3 [7].

```
1   Inicijalizacija:
2     ZA  i = 1 DO N
3       Inicijalizirati vektor x_i prema formuli 2.3
4     KRAJ ZA
5     iteracija = 1
6   PONAVLJAJ
7     Faza zaposlenih pčela:
8       ZA SVAKU zaposlenu pčelu (i)
9         Izračunati v_i na temelju x_i prema formuli 2.1
10        AKO kvaliteta(v_i) > kvaliteta(x_i) ONDA x_i = v_i
11      KRAJ ZA SVAKU
12    Faza promatrača:
13      ZA i = 1 DO N
14        Izračunati vjerojatnost P_i od x_i prema formuli 2.2
15      KRAJ ZA
16      ZA SVAKU pčelu-promatrača (i)
17        Odabrati x_i = x_j (uz vjerojatnost P_j)
18        Izračunati v_i na temelju x_i prema formuli 2.1
19        AKO kvaliteta(v_i) > kvaliteta(x_i) ONDA x_i = v_i
20      KRAJ ZA SVAKU
21    Faza izviđača:
22      ZA SVAKO odbačeno rješenje (x_i)
23        Generirati novo nasumično rješenje x_i  prema formuli 2.3
24      KRAJ ZA SVAKO
25    AKO MaxRješenje(iteracija) > MaxRješenje ONDA MaxRješenje = MaxRješenje(iteracija)
26    iteracija = iteracija + 1
27    AKO (iteracija == Cmax) ONDA PREKINI
28  KRAJ PONAVLJAJ
```

Slika 2.3 Pseudokod umjetne kolonije pčela

3. Umjetna neuronska mreža

Umjetna neuronska mreža (engl. *artificial neural network*) je matematički model načinjen po uzoru na strukturu bioloških neuronskih mreža. Takva mreža se obično sastoji od više slojeva umjetnih neurona koji, međusobno povezani, prosljeđuju informacije.

Razvoj na ovom području otvara rad McCullocha i Pittsa iz 1943. godine koji su razvili model neurona i neuronskih mreža [8]. Frank Rosenblatt 1958. razvija perceptron, algoritam za raspoznavanje uzoraka temeljen na dvoslojnoj neuronskoj mreži koji koristi jednostavne matematičke operacije. Nagli razvoj u ovom području stagnira nakon istraživanja Minskyja i Paperta iz 1969. godine gdje su analizirali ograničenja neuronskih mreža [9]. Prvi problem na koji su naišli je nemogućnost postojećih neuronskih mreža da nauče logičku funkciju "isključivo ili" (engl. *exclusive or*). Drugi problem je bio vezan za ograničenja računala koja se nisu mogla nositi s visokim zahtjevima velikih neuronskih mreža. Primijećena ograničenja su uspješno savladana razvojem snage računala te uvođenjem algoritma propagacije greške unatrag (engl. *backpropagation algorithm*).

3.1. Inspiracija

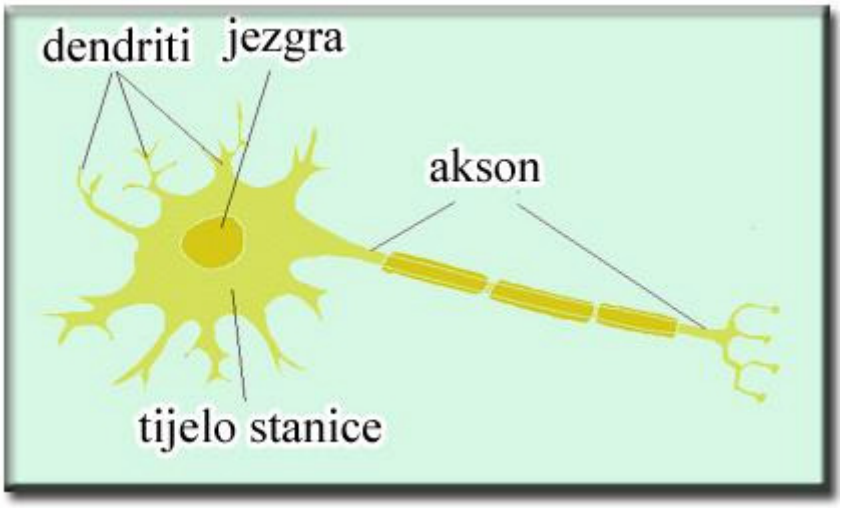
Inspiracija za umjetne neuronske mreže dolazi od načina kako ljudski mozak funkcionira. Način na koji mozak obavlja osnovne operacije se u potpunosti razlikuje od pristupa kojeg su digitalna računala koristila od svojih početaka. Mozak se oslanja na dobro usklađenu mrežu neurona koji su pojedinačno sporiji od osnovne jedinice računala. Neuroni svoju individualnu sporost nadoknađuju svojom brojnošću, efikasnošću te paralelnim i nelinearnim načinom rada. Ključ uspjeha neuronskih mreža leži u njihovoj sposobnosti učenja, odnosno stjecanja iskustva.

3.1.1. Biološki neuron

Biološki neuron (živčana stanica) je osnovna građevna jedinica bioloških neuronskih mreža. Procjenjuje se da prosječan mozak čovjeka sadrži 100 milijardi neurona od kojih je svaki povezan s 10 tisuća susjednih neurona. Iako se neuron u literaturi često dijeli na više različitih načina, ipak se uobičajeno ističu tri sastavna dijela najvažnija za razumijevanje kako ljudski živčani sustav funkcionira: tijelo stanice (engl. *soma*), akson (engl. *axon*) i dendriti (engl. *dendrites*).

Tijelo stanice je struktura kružnog oblika koje služi kao centar za primanje i slanje živčanih impulsa. Tijelo stanice sadrži jezgru (engl. *nucleus*). Akson je nastavak tijela

stanice u obliku cjevčice specijaliziran za prijenos impulsa. Dendriti također rastu iz tijela stanice te njihova struktura podsjeća na stabla. Obično dolaze po šest glavnih dendrita po jednom neuronu te im je glavna zadaća primanje impulsa, uglavnom od aksona drugih neurona. No, dendriti i aksoni se ne dodiruju direktno, već ih spaja uski prostor preko kojih se šalju impulsi – taj prostor nazivamo sinapsama (engl. *synapses*). Na slici 3.1 se mogu vidjeti svi spomenuti dijelovi biološkog neurona.



Slika 3.1 Osnovni dijelovi biološkog neurona

3.2. Modeliranje umjetne neuronske mreže

Umjetne neuronske mreže su slične svojim biološkim uzorima po tome što stječu znanje putem učenja te naučeno znanje pohranjuju u vezama između neurona. Kao prednost neuronskih mreža ističe se njihova visoka nelinearnost i prilagodljivost te otpornost na greške. U nastavku se daje kratki pregled kako se ideja iz bioloških verzija neuronskih mreža pretočila u računalnu verziju.

3.2.1. Model umjetnog neurona

U nastavku je opisan McCulloch-Pittsov model umjetnog neurona. Za ulaz se očekuje vektor, a za izlaz skalar. Bitni su sljedeći elementi: skup ulaza, sumator te aktivacijska funkcija.

Skup ulaza u perceptron je vektor $\vec{x} = (x_1, x_2, ..., x_n)$. Svaki ulaz se množi vlastitim težinskim faktorom (x_i se množi faktorom w_i) pa tako postoji i vektor težinskih faktora $\vec{w} = (w_1, w_2, ..., w_n)$. Spomenuti vektori se obično proširuju dodatnom dimenzijom – dimenzijom pomaka, koja omogućava perceptronu da korigira vlastitu sumu neovisno o

ulazima. Radi jednostavnosti zapisa, takav pomak se uvodi proširenjem vektora ulaza i težinskih faktora uz $x_0 = 1$ te proizvoljni w_0.

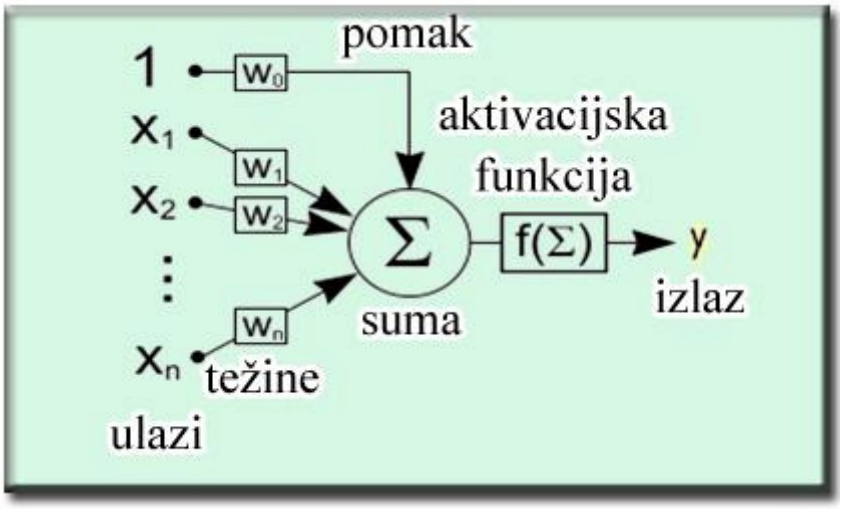
Sumator zbraja otežane ulaze, odnosno daje linearnu kombinaciju ulaza prema formuli (uz $x_0 = 1$):

$$suma = \sum_{i=0}^{n} w_i * x_i \tag{3.1}$$

Aktivacijska funkcija za ulaz prima sumu iz sumatora, a izlaz prosljeđuje na konačni izlaz iz trenutno promatranog umjetnog neurona. Pregled aktivacijskih funkcija dan je u poglavlju 3.2.2. Konačna formula za izlaz iz neurona dana je jednadžbom:

$$y = f(suma), \tag{3.2}$$

gdje je f(x) proizvoljna aktivacijska funkcija. Elementi neurona prikazani su na slici 3.2.



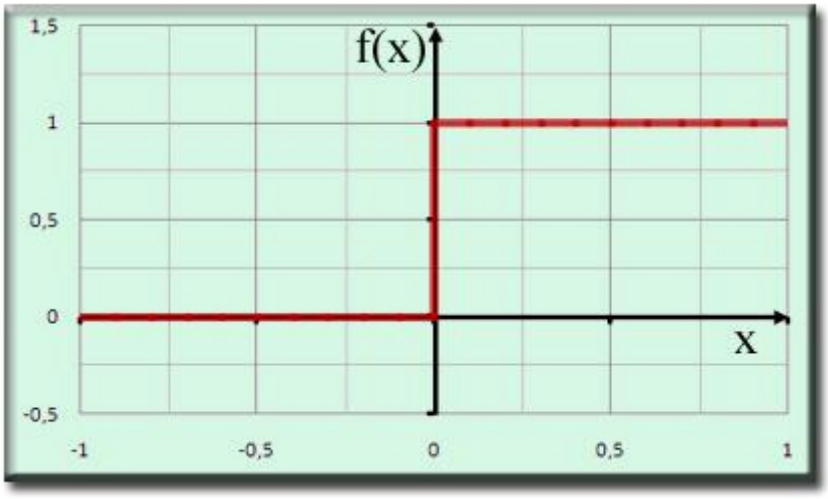
Slika 3.2 McCulloch-Pitts arhitektura umjetnog neurona

3.2.2. Aktivacijske funkcije

Kao što slika 3.2 prikazuje, svaki neuron nakon obrade ulaza treba dobiveni rezultat proslijediti do aktivacijske funkcije. Aktivacijska funkcija transformira svoj ulaz u konačni izlaz koji promatrani neuron dalje šalje sljedećim neuronima. Uobičajeno se spominju tri tipa aktivacijskih funkcija: funkcija praga, funkcija linearna po odsječcima i sigmoidna funkcija. Postoji još i četvrta uobičajena funkcija i to ona najjednostavnija koja samo proslijedi svoj ulaz na izlaz. Takva funkcija je karakteristična za model umjetnog neurona ADALINE.

Funkcija praga (engl. *step function*) je korištena u modelu perceptrona. Poopćena funkcija praga preslikava sve vrijednosti iznad određene granice (praga) u neku fiksnu vrijednost. Sve vrijednosti ispod te iste granice preslikavaju se u nulu. Uobičajeno se za prag koristi vrijednost 0, a sve vrijednosti iznad praga se preslikavaju u vrijednost 1. Korištenjem tako definirane aktivacijske funkcije dobiva se binarni izlaz iz neurona, što se može vidjeti na slici 3.3. Izlaz se računa prema sljedećoj jednadžbi:

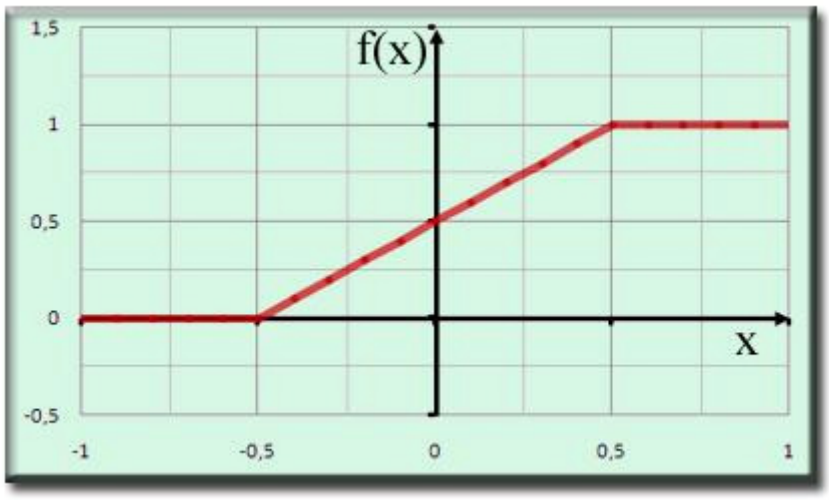
$$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases} \tag{3.3}$$



Slika 3.3 Aktivacijska funkcija praga

Aktivacijska funkcija koja je linearna po odsječcima (engl. *piecewise linear function*) dobiva se ako je izlaz ADALINE inačice ograničen na određeni interval. Tako je samo jedan dio preslikavanja linearan, dok periodi ispod i iznad zadane granice prelaze u minimalne, odnosno maksimalne vrijednosti. Poopćena funkcija ima dvije granice za ulaz i dvije fiksne vrijednosti za izlaz izvan granica. Kod spomenute funkcije se uobičajeno uzimaju -0,5 i +0,5 kao granice ulaza te 0 i 1 za vrijednosti izlaza. Takva se funkcija može vidjeti na slici 3.4 i zadaje se prema sljedećoj jednadžbi:

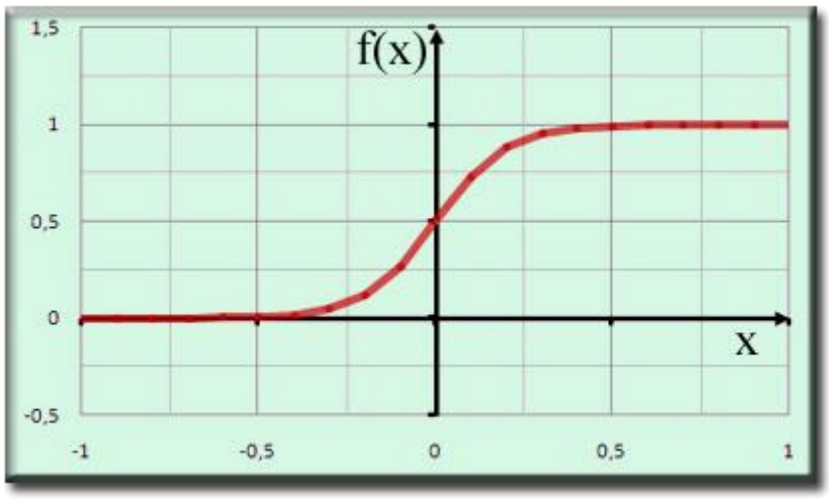
$$f(x) = \begin{cases} 1, & x \geq 0.5 \\ x + 0.5, & -0.5 < x < 0.5 \\ 0, & x \leq -0.5 \end{cases} \tag{3.4}$$



Slika 3.4 Aktivacijska funkcija linearna po odsječcima

Posljednja aktivacijska funkcija koja se ovdje razmatra je upravo i ona koja se koristi u ovome radu. Riječ je o sigmoidalnoj funkciji (engl. *sigmoid function*). Sigmoidna funkcija ima parametar nagiba "a". Povećanjem nagiba smanjuje se prostor linearnosti pa valja biti oprezan s podešavanjem tog parametra. Sigmoidna funkcija je zapravo vrlo slična funkciji praga, ali se dozvoljava područje nesigurnosti unutar određenog intervala i upravo se zato takva funkcija najčešće koristi. Sigmoidna funkcija uz nagib a = 10 prikazana je na slici 3.5, dok je poopćena jednadžba dana u nastavku:

$$f(x) = \frac{1}{1 + e^{-ax}} \tag{3.5}$$



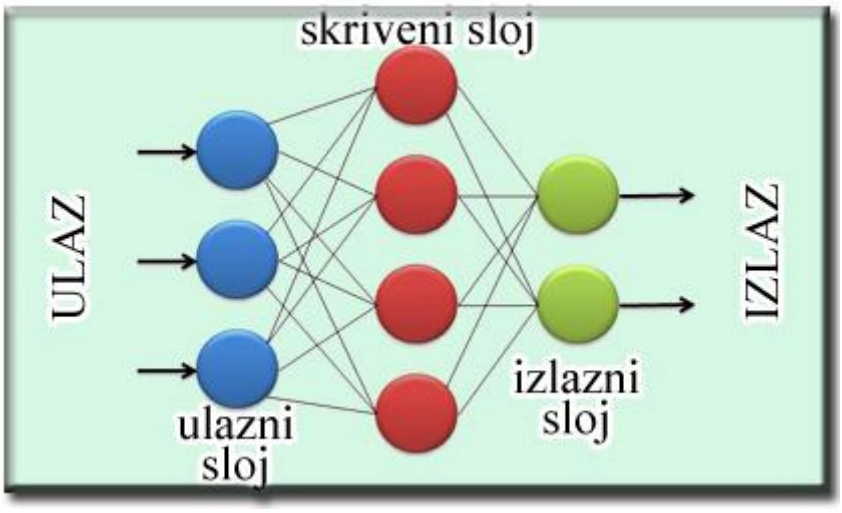
Slika 3.5 Sigmoidna aktivacijska funkcija

3.2.3. Arhitektura neuronske mreže

Arhitekturom neuronske mreže definiran je način kako su neuroni međusobno povezani. Izdvajaju se četiri najčešće vrste neuronskih mreža: jednoslojne acikličke mreže (engl. *single-layer feedforward networks*), višeslojne acikličke mreže (engl. *multi-layer feedforward networks*), mreže s povratnim vezama (engl. *recurrent networks*) te rešetkaste mreže (engl. *lattice structures*). Još se može izdvojiti posebna vrsta – hibridne mreže.

U ovom radu se koristi višeslojna aciklička mreža. Naziv "aciklička" ukazuje da takva mreža ne sadrži povratne veze pa je i struktura te njena implementacija jednostavnija jer se slanje informacija vrši u jednom smjeru. Naziv "višeslojna" govori da takva mreža ima više slojeva koji se obično dijele na: ulazni, skriveni i izlazni. Ulazni i izlazni sloj je uvijek jedan, dok skriveni sloj može imati proizvoljan broj slojeva. Struktura neuronske mreže često se opisuje kao n-torka n_1 x n_2 x...x n_n. Takva mreža ima n slojeva, gdje se ulazni sloj sastoji od n_1 neurona, izlazni od n_n neurona, a svi slojevi između su skriveni.

Za neuronsku mrežu je bitno da svaki neuron na ulaz prima izlaze od svih neurona iz njemu prethodnog sloja, dok vlastiti izlaz šalje na ulaze svih neurona njemu sljedećeg sloja. Na ulaz neuronske mreže donose se proizvoljni podaci te se očekuje da mreža na svoj izlaz daje očekivani rezultat. Neuronska mreža to postiže učenjem kroz veći broj iteracija sve dok ne počne davati željene rezultate. Arhitektura višeslojne acikličke mreže može se vidjeti na slici 3.6, gdje jedan krug predstavlja jedan neuron opisan na slici 3.2.



Slika 3.6 Primjer 3x4x2 višeslojne acikličke neuronske mreže

Skrivenih slojeva može biti i više, ali je uobičajeno uzeti samo jedan sloj. U literaturi se mogu naći razne preporuke za veličinu skrivenog sloja – za manju neuronsku mrežu od četiri klase se mogu koristiti svega tri neurona u skrivenom sloju [10].

4. Problem raspoznavanja novčanica

Za rješavanje problema raspoznavanja novčanica u implementacijskom dijelu ovog rada koristi se umjetna kolonija pčela (opisana u poglavlju 2) te umjetna neuronska mreža (opisana u poglavlju 3). Ideja je da se primjerak novčanice predaje na ulaz u neuronsku mrežu te da se na izlazu daje klasa kojoj pripada novčanica na ulazu. Za učenje neuronske mreže koja obavlja takav zadatak koristi se umjetna kolonija pčela koja optimira parametre neuronske mreže, odnosno težinske faktore i pomake perceptrona.

Problem raspoznavanja novčanica spada u problem raspoznavanja uzoraka, dok se oba problema mogu svrstati u zajedničko područje strojnog učenja. U nastavku se daje kratki opis spomenutih područja, odnosno problema – od općenitijeg prema specifičnijem.

4.1. Strojno učenje

„Za računalni program kažemo da uči iz iskustva E nad klasom zadataka T uz mjeru uspješnosti P, samo ako mu se mjera uspješnosti P nad zadacima T povećava kroz iskustvo E.“ (Tom Mitchell) [11]

Strojno učenje (kao grana umjetne inteligencije) bavi se razvojem algoritama koji bi omogućili računalima razvoj nekih specifičnih sposobnosti na temelju iskustva, odnosno primanjem podataka iz okoline. Jedan od glavnih ciljeva strojnog učenja je dati algoritmima dobru moć prepoznavanja složenih uzoraka te sposobnost donošenja inteligentnih odluka temeljem dostupnih podataka. Dodatni je izazov ostvariti uspješan algoritam koji djeluje uz minimalno uplitanje čovjeka – odnosno postizanje maksimalno automatiziranog procesa učenja.

U području strojnog učenja uobičajeno se javlja problem velike dimenzionalnosti ulaza pa se često radi na smanjenju dostupnih podataka i ekstrakciji onih bitnih te uklanjanju manje bitnih. Također je bitno da algoritam nije previše specifičan, da je prilagodljiv te da ima sposobnost generalizacije.

4.2. Raspoznavanje uzoraka

Raspoznavanje uzoraka se svodi na pridruživanje oznake (izlaz) obzirom na dani primjerak (ulaz). Primjerci mogu biti vizualnog, zvučnog ili podatkovnog karaktera. U nastavku je dana formalna definicija problema raspoznavanja uzoraka.

4.2.1. Formalna definicija problema

Ovdje je dana definicija raspoznavanja uzoraka uz nadgledanje. Zadan je skup ulaza x ∈ X, izlaza (oznaka) y ∈ Y te nepoznata funkcija g : X → Y koja preslikava ulaze u željene (ispravne) izlaze. Uz to je zadan skup za učenje T = {(x_1,y_1), (x_2,y_2), ..., (x_n,y_n)} koji predstavlja ispravna preslikavanja funkcije g. Zadatak raspoznavanja uzoraka je pronaći funkciju h: X → Y koja aproksimira preslikavanje g (što je bolje moguće). Takva se aproksimacija može definirati na više načina, ali za potrebe ovog rada uzima se ona najjednostavnija. Definira se pojedinačna pogreška nad jednim uzorkom, error(x, g, h):

$$error(x,g,h) = \begin{cases} 1, g(x) \neq h(x) \\ 0, g(x) = h(x) \end{cases}, \tag{4.1}$$

gdje je x ulazni uzorak, g(x) ispravna klasifikacija te h(x) klasifikacija koju daje pretpostavljena h-funkcija. Sada se može definirati pogreška nad cijelim skupom za evaluaciju E = {(x_1,y_1), (x_2,y_2), ..., (x_n,y_n)}:

$$ERROR\ (E,g,h) = \frac{\sum_{x \in X'} error(x,g,h)}{|E|}, \tag{4.2}$$

gdje je X' = {x_1, x_2, ..., x_n}. Zadatak raspoznavanja uzoraka je: nad danim skupom za evaluaciju E, pronaći takvu funkciju "h" koja minimizira ERROR(E,g,h).

4.2.2. Karakteristični podzadaci

Raspoznavanje uzoraka je složen zadatak i valja istaknuti neke karakteristične podzadatke, odnosno probleme koji mogu nastati. [12]

Ekstrakcija značajki – zadatak u ovoj fazi je izdvojiti samo one podatke iz ulaznog skupa podataka koji će preostali proces učiniti najlakšim mogućim. Drugim riječima – valja odabrati one informacije koje su najbitnije za donošenje ispravne odluke.

Uklanjanje šuma – gotovo je neizbježno imati zadatak raspoznavanja uzoraka bez postojanja šumova na danim ulazima. Šumovi predstavljaju problem za proces raspoznavanja uzoraka jer oni mijenjaju informacije koje se očekuju u idealnom slučaju. Kao primjer mogu poslužiti sjene na slikama ili pozadinska glazba kod prepoznavanja govora.

Izbjegavanje prenaučenosti – nije uvijek poželjno imati savršenu klasifikaciju skupa primjera za učenje ako se kao rezultat dobiva presložena funkcija odlučivanja.

Obično je bolje žrtvovati mali dio uspješnosti za smanjenje složenosti pretpostavljene funkcije. Upravo zato što se ne može očekivati da će se primjeri za ispitivanje ponašati identično kao oni za učenje. Zadržavanjem previše složenog načina donošenja odluke povećava se vjerojatnost lošijih rezultata na skupu za ispitivanje, u odnosu na jednostavniji način. Time se ujedno postiže bolja sposobnost generalizacije.

Rad s nepotpunim informacijama – kako se nositi s parcijalnim slikama gdje je dio potrebnih informacija nedostupan ili kako prepoznati riječ na zvučnom zapisu ako je došlo do kratkog prekida tijekom prijenosa. Ovo je vrlo složen problem i neki pristupi rješavanju raspoznavanja uzoraka se ne mogu nositi s tim problemom.

Segmentacija ulaza – slično prethodnom problemu, ovaj se javlja kada je više primjeraka dano na istom ulazu pa je potrebno prvo tijekom obrade razdvojiti ulazne podatke na različite primjerke i dalje ih tretirati kao individualne. Kod raspoznavanja novčanica taj se problem javlja ako se na ulaz predaje slika s tri novčanice odjednom – može li sustav prepoznati sve tri novčanice odjednom?

Korištenje specifičnog znanja – korištenje informacija karakterističnih za dani problem prilikom dizajniranja sustava. Na primjeru raspoznavanja novčanica to može biti informacija da su novčanice pravokutnog oblika (a ne kružnog ili nekog drugog).

Korištenje konteksta – ovaj pojam često obuhvaća apstraktne i složene koncepte, ali kao primjer se može navesti automat za prepoznavanje novčanica koji može prepoznati bilo koju valutu na svijetu. Ako na ulaz primi američki dolar, tada je očekivanje sljedećeg primjerka u obliku američkog dolara veće nego za bilo koju drugu valutu jer novčanice obično dolaze jedna za drugom.

Podrška invarijantnosti – ako je govor na zvučnom zapisu dubljeg ili višeg glasa, muškog ili ženskog – sustav bi uvijek trebao jednake riječi svrstavati u jednake grupe bez obzira na to tko ih je rekao. Kod vizualnih primjera sustav mora biti invarijantan na rotacije, promjene u veličini i slične transformacije. Općenito govoreći – sustav treba biti neosjetljiv na neke transformacije ulaza.

4.2.3. Primjene raspoznavanja uzoraka

Područje raspoznavanja uzoraka je širokog spektra primjene i ovdje se navode samo neke primjene, pored raspoznavanja slika. Klasifikacija pojmova u semantičke kategorije (posebno korišteno za Internet). Sortiranje dokumenata po vrsti, npr. filtriranje neželjene elektronske pošte (engl. *spam*) ili grupiranje novinskih članaka po temama.

Prepoznavanje znakova i riječi s dokumenata, odnosno njihovih slika. Raspoznavanje lica ili otiska prstiju za identifikaciju. Ekstrakcija riječi iz zvučnog signala. Odvajanje smislenih uzoraka nad velikim bazama podataka (engl. *data mining*). Dijagnoze u medicini na temelju specifičnih slika. Automatski pronalazak potencijalnih prijetnji u vojne svrhe.

4.3. Raspoznavanje novčanica

Za raspoznavanje novčanica vrijedi sve prethodno spomenuto za raspoznavanje uzoraka, no ovdje se još posebno definira zadatak i neke specifičnosti za ovaj podskup problematike raspoznavanja uzoraka.

4.3.1. Zadatak

Osnovni zadatak ovog rada je ostvariti raspoznavanje novčanica dovoljno visoke uspješnosti (više od 90%) uz zadovoljavajuće vrijeme izvođenja (manje od 1 sekunde za jednu novčanicu). Ispitivanja su provedena nad hrvatskim kunama i to nad novčanicama od 10, 20, 50, 100, 200, 500 i 1000 kuna, uključujući lice i naličje te rotacije. Bitno je da je sustav otporan na šumove te da je sposoban generalizirati izbjegavajući prenaučenost.

4.3.2. Pojednostavljenja

Ekstrakcija značajki je fiksna – sa slike novčanice se predaju samo određene definirane regije na ulaz u neuronsku mrežu. Drugim riječima – proces nije dinamički u smislu da sustav sam dohvaća željene značajke, već dohvaća točno određena područja prema naputku korisnika. Kao kompromis pružena je mogućnost automatskog pronalaska takvih regija nad proizvoljnim skupom primjera.

Problemi rada s nepotpunim informacijama i segmentacije ulaza nisu riješeni u ovome radu prvenstveno zato što odabrani pristup rješavanju (korištenjem umjetnih neuronskih mreža) nije namijenjen za rješavanje takvih problema. Na ulazu se očekuje samo jedna i to neprekrivena novčanica koju sustav treba prepoznati. Kontekst se također ne koristi radi jednostavnijeg ispitivanja.

Naglasak je stavljen na uklanjanje šuma te podrška invarijantnosti. Novčanice na ulazu se očekuju različitih stupnjeva oštećenosti i zgužvanosti te sustav treba ispravno klasificirati i takve novčanice. Također, sustav ne treba raditi razliku između rotiranih novčanica ili obzirom na dimenzije slike.

5. Programsko rješenje

Nakon razmatranja prethodno proučene teorijske pozadine, ostvareno je programsko rješenje za dani problem. U ovom se dijelu opisuju načini kako su ostvareni pojedini koncepti.

Kao razvojna okolina korišten je Microsoftov Visual Studio 2008, programski jezik C#. Od vanjskih biblioteka korišten je paket AForge radi lakšeg ostvarenja složenih algoritama vezanih za obradu slika.

5.1. Skup podataka

Program koristi tri interne strukture koje se zapisuju na disku: ispitni skupovi slika (.set), skup težinskih faktora neuronske mreže (.net) i upravljačke datoteke (.man). Ispitni skupovi slika objedinjuju primjere slika koje se predaju neuronskim mrežama na ulaze. U skupu težinskih faktora se pamte težinski faktori (između ostalog) istrenirane umjetne neuronske mreže. Upravljačke datoteke pohranjuju upute za donošenje konačne odluke na temelju jedne ili više neuronskih mreža, birajući određene regije unutar slika koje se koriste pri donošenju odluke. U nastavku su detaljnije opisane navedene strukture.

5.1.1. Ispitni skupovi slika

Ispitni skupovi slika predstavljaju internu strukturu koja pohranjuje reference na slike (novčanice) koje se kao zajednički skup prosljeđuju neuronskoj mreži na učenje ili ispitivanje. Takve strukture se pohranjuju u datotekama ".set" ekstenzije, a sadržaj se može pregledavati u običnom tekstualnom editoru (izvan aplikacije) ili preko opcije "Upravljanje skupovima slika" (unutar aplikacije).

Tablica 5.1 Definicija strukture ispitnog skupa slika

	rbr	**stavka**	**oznaka**	**format**
	1	Ukupni broj primjera	S	INTEGER
	2	Ukupni broj ispitivanja skupa	T	INTEGER
S puta (stavke odvojene s /n)	3	Lokacija slike primjera		PATH
	4	Globalna klasifikacija primjera		STRING
	5	Lokalna klasifikacija primjera		INTEGER/CHAR
	6	Ukupni broj ispravnih klasifikacija	C	INTEGER

Očekivana struktura ispitnih skupova slika se može vidjeti u tablici 5.1. Uspješnost klasifikacije pojedinog primjera unutar skupa može se izračunati omjerom C / T. Za

lokaciju slike primjera (stavka 2) bitno je napomenuti da se zapisuje relativna staza u odnosu na poziciju pokretanja aplikacije. Globalna klasifikacija primjera (stavka 4) predstavlja internu oznaku promatranog primjera koja je jedinstvena za sve ispitne skupove slika s kojima radi aplikacija. Korisnik se mora pobrinuti da, primjerice 10 hrvatskih kuna u jednoj datoteci bude označeno na jednak način kao i svih drugih 10 hrvatskih kuna u ostalim datotekama kako bi sustav znao da je riječ o jednakoj klasi novčanice u svim slučajevima. S druge strane, lokalna klasifikacija primjera (stavka 5) ima doseg pojedinog učenja neuronske mreže i ovdje se daju upute kakvi se izlazi od mreže očekuju. Lokalne oznake unutar ispitnog skupa slika trebaju početi od broja 0 i nastaviti u rastućem nizu za svaku novu klasu (ne nužno i primjer). Ako se želi da neuronska mreža daje odgovor "ništa od navedenog" za promatrani primjer, tada se na mjestu lokalne klasifikacije postavlja oznaka "-", što sustav interpretira kao posljednju klasu unutar mreže.

5.1.2. Skup težinskih faktora neuronske mreže

Skup težinskih faktora neuronske mreže se odnosi na internu strukturu koja pamti težinske faktore trenirane neuronske mreže. Drugim riječima, ova struktura pohranjuje iskustvo mreže, odnosno naučene veze između neurona. Osim težinskih faktora pohranjuju se i dodatni parametri potrebni za pokretanje rada neuronske mreže koji su opisani u nastavku. Spomenute strukture se nalaze u datotekama ".net" ekstenzije, a sadržaj se ne može pregledavati unutar aplikacije jer se takve informacije žele sakriti od korisnika.

Tablica 5.2 Definicija strukture skupa težinskih faktora neuronske mreže

	rbr	**stavka**	**oznaka**	**format**
	1	Lokacija skupa slika za učenje		PATH
	2	Faktor pouzdanosti neuronske mreže	Q	DOUBLE
	3	Ukupni broj uspješnih prepoznavanja	C	INTEGER
	4	Ukupni broj ispitivanja neuronske mreže	T	INTEGER
	5	Struktura neuronske mreže		[INTEGER \t] +
X puta	6	Težinski faktor neuronske mreže		DOUBLE

Definirana struktura skupa težinskih faktora neuronske mreže dana je u tablici 5.2. Budući da neuronske mreže daju izlaze poput 0, 1, 2 ... ili "ništa od navedenog", onda je potrebno pamtiti referencu na ispitni skup slika nad kojim je mreža učena (stavka 1) kako bi se odgovori neuronske mreže (brojevi) doveli u vezu s klasifikacijama na razini aplikacije (npr. 500 hrvatskih kuna). Faktor pouzdanosti neuronske mreže (stavka 2) služi za davanje određene težine odgovoru kojeg promatrana neuronska mreža poručuje sustavu.

U interesu je postaviti taj broj na što manji u slučaju nepouzdanosti neuronske mreže. Sustav tako prilikom donošenja konačne odluke (ako više neuronskih mreža zajedničkim snagama donosi odluku) zna da promatranoj neuronskoj mreži ne treba previše vjerovati. Taj se broj može postaviti na omjer C / T čime se dobiva stvarna uspješnost neuronske mreže nad svim do sad ispitanim primjerima. Struktura neuronske mreže se zapisuje u jednom retku (stavka 5) tako da se zapisuje broj neurona po svakom sloju uz odvajanje tab znakovima. Veličina ulaznog sloja mora odgovarati broju piksela regije koja se prosljeđuje na ulaz, dok veličina izlaznog sloja treba odgovarati broju klasa na koji neuronska mreža treba odvajati primjere (uključujući jedan neuron za "ništa od navedenog" odgovor).

Na kraju se zapisuju težinski faktori neuronske mreže (stavka 6), svaki u svoj redak datoteke. Neuronska mreža n_U x n_S x n_I arhitekture sadrži ukupno (n_U * n_S + n_S * n_I) težinskih faktora te (n_S + n_I) faktora pomaka.

5.1.3. Upravljačke datoteke

Upravljačke datoteke daju upute za donošenje konačne odluke kada se sustavu predoči neki primjer novčanice koji valja prepoznati. Upravljačke datoteke se prepoznaju po ".man" ekstenziji te popisuju informacije o svim klasama od interesa te regijama, odnosno neuronskim mrežama koje se uzimaju u obzir kod donošenja odluke.

Tablica 5.3 Definicija strukture upravljačke datoteke

		rbr	stavka	oznaka	format
		1	Ukupni broj klasa	N	INTEGER
N puta (stavke odvojene s /t)		2	Globalni kod klasifikacije (interni)		STRING
		3	Globalni kod klasifikacije (korisnički)		STRING
		4	Valuta novčanice		STRING
		5	Vrijednost novčanice		INTEGER
		6	Rotacija novčanice		CHAR
		7	Strana novčanice		CHAR
		8	Ukupni broj regija	R	INTEGER
R puta (stavke odvojene s /t)		9	Oznaka regije (interna)		STRING
		10	Oznaka regije (korisnička)		STRING
		11	Početna pozicija regije na X osi	XS	DOUBLE
		12	Početna pozicija regije na Y osi	YS	DOUBLE
		13	Širina regije (X os)	dX	DOUBLE
		14	Visina regije (Y os)	dY	DOUBLE
		15	Ukupni broj mreža (za danu regiju)	NN	INTEGER
	NN puta	16	Lokacija skupa težinskih faktora mreže		PATH

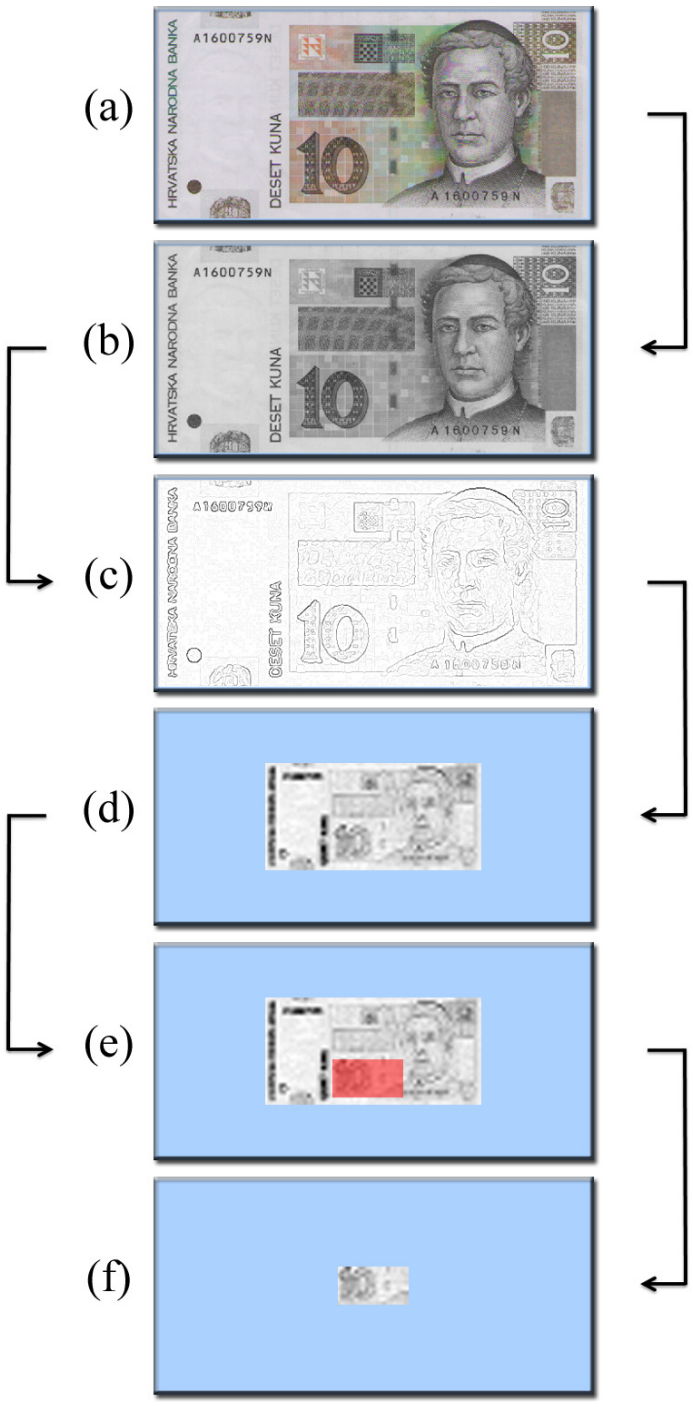
U tablici 5.3 može se vidjeti očekivana struktura upravljačkih datoteka. Logički se takva datoteka može podijeliti na klase (stavke 1 - 7) te na regije (stavke 8 - 16).

Prvi dio upravljačke datoteke sadrži informacije o svim očekivanim klasama koje se mogu pojaviti kao odgovor unutar promatranog sustava. Za ostvarenje sustava koji prepoznaje više valuta mora se navoditi svaka valuta (i njene denominacije). Interni globalni kod klasifikacije (stavka 2) se dovodi u vezu s ispitnim skupom slika (.set) i njihovom stavkom 4 prikazanoj u tablici 5.1. Korisnički globalni kod klasifikacije (stavka 3) daje korisniku prilagođen opis promatrane klase. Valuta novčanice (stavka 4) se predstavlja s tri znaka (HRK za hrvatske kune), ali vrijednost može biti proizvoljna. Vrijednost (stavka 5) predstavlja vrijednost novčanice unutar promatrane valute (10, 20 ...). Rotacija (stavka 6) može biti normalna (oznaka "N") ili rotirana za 180° (oznaka "R"). Unutar upravljačke datoteke ne treba navoditi rotirane novčanice, već sustav interno generira sve rotirane primjerke i njihovo navođenje bi dovelo do duplikacije podataka. Strana (stavka 7) predstavlja oznaku na koju stranu je okrenuta novčanica – lice se označava oznakom "F", a naličje oznakom "B". Za ispitne primjere unutar aplikacije se koristio standard internog koda (stavka 2) na način: valuta – vrijednost – strana – rotacija. Na primjer, novčanica od 10 kuna koja je rotirana za 180° i prikazana s prednje strane je označena s "HRK-10-F-R".

U drugom dijelu upravljačke datoteke nalaze se podaci o tome koje se regije slika prosljeđuju neuronskim mrežama te koje mreže sustav treba koristiti pri donošenju odluke. Informacije o koordinatama regija su dane u decimalnom formatu te su standardizirane na [0, 1] interval. Svaka je regija definirana pravokutnikom: početna točka u lijevom gornjem kutu (stavke 11 i 12) te dimenzije regije (stavke 13 i 14). Regija zapisana kao (0 0 0,5 0,5) promatra samo gornju lijevu četvrtinu slike, dok se preostali pikseli ignoriraju. Dodatno, za svaku regiju može postojati više neuronskih mreža koje primaju ulaze iz zajedničke regije te donose internu odluku. U pravilu nema razloga da bude više od jedna neuronska mreža po regiji, ali u nekim situacijama bi moglo biti poželjno imati više neuronskih mreža koje su trenirane pod različitim uvjetima kako bi sustav bio otporniji na greške. Na kraju se lokacija svakog skupa težinskih faktora neuronske mreže (.net) mora navesti za svaku regiju (stavka 16). Skupovi su zapisani svaki u vlastitom redu i zapis se ponavlja za svaku navedenu regiju.

5.2. Obrada slike

Prije nego što se slika pošalje na ulaz neuronske mreže za njenu klasifikaciju, svaka slika mora proći niz zahvata kako bi daljnji postupak bio što efikasniji. Pretpostavka je da promatrana slika sadrži jednu novčanicu prikazanu u potpunosti i to preko cijele slike (slika ne smije sadržavati pozadinske elemente). Valja napomenuti da aplikacija podržava način rada gdje je pozadina dozvoljena, ali je taj način rada opisan kasnije – tada sustav mora pripremiti sliku u formatu kakav se očekuje kako bi se slika mogla dalje koristiti.



Slika 5.1 Proces obrade slike prije ulaska u neuronsku mrežu

Slika 5.1 grafički prikazuje korišteni proces obrade slike na primjeru novčanice od 10 kuna. Na ulaz se prima neobrađena slika (a) kojoj se uklone boje tako da ostanu samo nijanse sive boje (b). Slika se dalje predaje algoritmu za detekciju rubova (c) nakon čega se veličina slike promijeni na interne dimenzije 61x31 piksela (d). Zatim se iz dobivene umanjene slike odabere željena regija (e) nakon čega ostaje izlaz iz procesa obrade slike (f). Tek se tako dobiveni pikseli mogu proslijediti neuronskoj mreži na obradu. Treba istaknuti kako slika 5.1 na prikazima od (c) do (f) ima invertirane nijanse od onih koje se zapravo koriste unutar sustava radi bolje preglednosti. U sljedećim poglavljima su detaljnije opisani zahvati koji su upravo navedeni.

5.2.1. Pretvorba u crno-bijelu sliku

Kao prvi korak prilikom obrade slike, slici se oduzimaju boje te se slika ostavlja crno-bijelom. Postoje tri razloga za takvu odluku. Kada bi se zadržale informacije o boji, tada bi se povećala količina informacija koja se predaje neuronskoj mreži, a prethodna iskustva su pokazala kako slični sustavi funkcioniraju dobro bez ikakvih informacija o boji. Na ovaj je način svaki piksel predstavljen brojem iz intervala [0, 255], odnosno decimalnim brojem iz intervala [0, 1] nakon normalizacije. Drugi razlog za uklanjanje boja leži u činjenici da starije novčanice mogu mijenjati nijanse svoje boje pa se na ovaj način uklanja nepotrebni šum. Treći razlog leži u koraku koji slijedi (detekciji rubova) koji bi sliku ionako pretvorio u crno-bijelu. Čak i kada bi se zadržale informacije o bojama, tada se ne bi mogla provoditi detekcija rubova koja se pokazala vrlo korisnom.

Kao protuargument može se navesti primjer hrvatskih kuna kod kojih svaka novčanica ima vlastitu prepoznatljivu boju pa bi takav podatak mogao biti od velike važnosti za naš proces klasifikacije. No, zanimljiv je primjer kod hrvatskih novčanica od 10 kuna gdje postoje dvije verzije novčanice – ona standardna, s prevladavajućom sivom bojom te druga verzija, ona ljubičaste boje (više nije u opticaju). Obje su novčanice istog dizajna, ali različitih boja. Uklanjanjem boja, te dvije novčanice se poistovjećuju, što i ima smisla. Dodatno, ako se pogleda primjer američkih dolara, kod kojih prevladava jednaka boja na svim novčanicama, tada argument važnosti boja gubi na vrijednosti.

Za uklanjanje boja koristila se jednostavna formula gdje se uzima 30% crvene, 59% zelene i 11% plave boje, po RGB notaciji:

$$K = 0.3R + 0.59G + 0.11B \tag{5.1}$$

Dobivena vrijednost K nalazi se u jednakom intervalu kao R, G i B vrijednosti jer suma koeficijenata daje 1.

5.2.2. Detekcija rubova

Nakon što je slika pretvorena u crno-bijelu, tada se ulaz dalje prosljeđuje do algoritma za detekciju rubova. U programskom rješenju korišten je Cannyjev detektor rubova. Spomenuti algoritam se vrlo često koristi, a uveo ga je John Canny u svom djelu iz 1986. godine. [13]

Cannyjev detektor rubova zapravo je filtar koji pretvara ulaznu crno-bijelu sliku u sliku na kojoj su vidljivi samo rubovi kao izlaz. Postupak se odvija kroz četiri glavna koraka: glađenje slike (redukcija šuma), određivanje gradijenta intenziteta slike, stanjivanje rubova te usporedba s pragovima. Kao tri glavne značajke koje takav algoritam mora zadovoljiti često se izdvajaju – kvalitetna detekcija (prepoznavanje ispravnih rubova), kvalitetna lokalizacija (precizno pozicioniranje rubova u odnosu na originalnu sliku) te minimalni odziv (da se svaki rub označi samo jednom, a oni lažni niti jednom).

Detalji spomenutog algoritma nisu proučavani jer je programski ostvaren korištenjem gotove AForge biblioteke za C#. AForge inače pruža mnoge funkcionalnosti u području analize slika. Cannyjev detektor se nalazi u AForge.Imaging.Filters biblioteci, a funkcija se zove CannyEdgeDetector. Funkcija zahtjeva tri parametra: donja granica, gornja granica te vrijednost sigma. Za parametre su (nakon nekoliko pokušaja) korištene sljedeće vrijednosti: donja granica = 5, gornja granica = 10 te sigma = 1,0.

5.2.3. Promjena dimenzija

Tek nakon što su izdvojeni rubovi, tada se veličina slike mijenja na interne dimenzije zato što sve što se predaje neuronskoj mreži mora biti jednake veličine svaki put, neovisno o veličini početne slike te zato što je poželjno smanjiti dimenzionalnost ulaza radi bržeg učenja te ispitivanja. Naime, predavanje slika većih dimenzija na ulaz u neuronsku mrežu (čak i da se takva mreža uspije istrenirati) ne bi nužno davalo bolje rezultate jer bi u tom slučaju slike bile podložnije šumovima. Kada se slika smanji na manje dimenzije, tada se i šumovi u nekoj mjeri uklanjaju jer se više piksela svodi na jedan uzimanjem srednje vrijednosti.

Što se tiče internih dimenzija, isprobano je više vrijednosti dok nije nađena vrijednost 61x31 iz nekoliko razloga. Prvi razlog je da dimenzije nisu smjele biti premale, inače bi dohvat regija od već ionako male slike bio nemoguć. Stoga su odbačeni ekstremi

poput 10x5 ili čak 20x10 dimenzija. Ova je mjera pronađena nakon nekoliko pokušaja jer se pokazala dovoljno dobrom bez da se nepotrebno povećava dimenzionalnost. Gotovo jednako kvalitetni rezultati dobivali su se i za nešto manje dimenzije. Omjer dimenzija od približno 2 (preciznije: 1,97) preuzet je na temelju dimenzija hrvatskih kuna što se može vidjeti u tablici 5.4 gdje su dodatno prikazani i omjeri eura te američkih dolara.

Tablica 5.4 Omjeri novčanica za hrvatske kune, eure i američke dolare

Valuta	Vrijednost	Dimenzije	Širina	Dužina	Omjer
HRK	10	126 × 63 mm	126,00	63,00	2,00
HRK	20	130 × 65 mm	130,00	65,00	2,00
HRK	50	134 × 67 mm	134,00	67,00	2,00
HRK	100	138 × 69 mm	138,00	69,00	2,00
HRK	200	142 × 71 mm	142,00	71,00	2,00
HRK	500	146 × 73 mm	146,00	73,00	2,00
HRK	100	150 × 75 mm	150,00	75,00	2,00
EUR	5	120 × 62 mm	120,00	62,00	1,94
EUR	10	127 × 67 mm	127,00	67,00	1,90
EUR	20	133 × 72 mm	133,00	72,00	1,85
EUR	50	140 × 77 mm	140,00	77,00	1,82
EUR	100	147 × 82 mm	147,00	82,00	1,79
EUR	200	153 × 82 mm	153,00	82,00	1,87
EUR	500	160 × 82 mm	160,00	82,00	1,95
USD	1	156 x 66 mm	156,00	66,00	2,36
USD	2	156 x 66 mm	156,00	66,00	2,36
USD	5	156 x 66 mm	156,00	66,00	2,36
USD	10	156 x 66 mm	156,00	66,00	2,36
USD	20	156 x 66 mm	156,00	66,00	2,36
USD	50	156 x 66 mm	156,00	66,00	2,36
USD	100	156 x 66 mm	156,00	66,00	2,36
Sredina					**2,08**

Ostvarenje promjene dimenzija unutar programskog rješenja nije bio trivijalan zadatak. Naime, korištenjem već ugrađenih funkcija za promjenu dimenzija slika primijećen je zanimljiv problem kod rotacija slika. Kako bi sustav uspješno prepoznao novčanice rotirane za 180°, tada sustav mora proći sljedeći ispit: interni prikaz bilo koje slike na ulazu mora biti identičan rotiranom prikazu rotirane slike na ulazu. Drugim riječima, ako su zadane dvije jednake slike na ulazu (jedna normalna i jedna rotirana), tada interni prikaz tih slika nakon svih obrada mora biti također jednak, ali pod kutom od 180° bez ikakvih dodatnih odstupanja. Takav zahtjev osigurava da sustavu nije bitno je li novčanica rotirana ili nije. Korištenjem gotovih funkcija za promjene veličine slika sustav

nije prošao na spomenutom ispitu. Zato se stvorila potreba da se ostvari vlastita metoda za promjenu dimenzija slike koja bi bila invarijantna na rotacije od 180° u smislu da sustav prođe postavljeni ispit.

Vlastito ostvarenje promjene dimenzije slika načinjeno je na sljedeći način. Uveden je zahtjev da ciljane dimenzije budu neparni (cijeli) brojevi iz razloga kako bi uvijek postojao središnji piksel – što osigurava invarijantnost na rotacije. Zbog tog je uvjeta u konačnici i uzeta vrijednost 61x31 za dimenzije slika, gdje su obje dimenzije neparni brojevi. Naravno, uvjet na početne dimenzije slike nije postavljen.

U prvom koraku se početne dimenzije po osima podijele na ciljane dimenzije tako da podjela uvijek bude simetrična u odnosu na središnji piksel (da prvi izlazni piksel pokriva jednak broj originalnih piksela kao i zadnji izlazni piksel). Takav pristup osigurava da se uvijek u obzir uzimaju isti pikseli, neovisno o rotaciji slike. Nakon podjele dimenzija uzima se aritmetička sredina početnih vrijednosti piksela kao vrijednost za izlazne piksele.

5.2.4. Dohvat regija

Konačni korak predstavlja dohvaćanje regija sa slike internih dimenzija 61x31. Regije su fiksne u smislu da, jednom kada ih korisnik odredi, svi procesi koji se pozivaju na tu regiju uzimaju piksele dalje na obradu uvijek na istom mjestu. Zato je bitno istaknuti kako sustav uvijek očekuje prije obrade da slika bude poravnata u smislu da se različite novčanice preklapaju, a ne da budu previše pomaknute u odnosu jedna na drugu. Naravno, odstupanja su neizbježna i neuronska mreža se provjereno za manja odstupanja ponaša dovoljno kvalitetno, odnosno – mali pomaci u pravilu ne utječu na konačnu odluku sustava jer ima sposobnost generalizacije.

Korak dohvata regija može se zaobići od strane korisnika tako da se definira samo jedna regija s vrijednostima (0 0 1 1). Takva regija prekriva cijelu sliku – uz početnu točku (0 0) te dimenzije (100% 100%). No, ovakav zaobilazak se treba uzeti s oprezom. Naime, na novčanicama obično postoje određena mjesta koja se razlikuju unutar klasa istih novčanica (na primjer serijski kod novčanice) i nije preporučljivo takve piksele predavati neuronskoj mreži na ispitivanje jer to uzrokuje dodatni šum za mrežu. Također, mnoge novčanice imaju prazna područja koja ne odmažu neuronskoj mreži u smislu kvalitete, ali bi se nepotrebno trošili resursi na analizu tih podataka. Iz navedenih razloga su uvedene regije koje se predaju kao konačni ulaz na pojedinu neuronsku mrežu.

5.3. Klasifikacija

Jezgru izvedenog sustava predstavlja mehanizam koji usmjerava proces učenja pomoću funkcije dobrote (engl. *fitness function*), odnosno način donošenja odluke o klasifikaciji ulaza. U nastavku je dan detaljan opis kako je to ostvareno unutar izvedenog sustava.

5.3.1. Funkcija dobrote

Algoritam umjetne kolonije pčela treba imati definiranu funkciju dobrote koju valja minimizirati postupkom učenja neuronske mreže. Budući da je zadatak mreže ispravno klasificirati novčanice, tada se za funkciju dobrote bira uspješnost klasifikacije neuronske mreže nad skupom ispitnih primjera.

Još treba definirati uspješnost klasifikacije neuronske mreže. Prva je opcija uzeti grubu uspješnost prema formuli 4.2, ali se rezultati uz takvu funkciju nisu pokazali najboljim u slučaju velikih skupova. Naime, evaluacija neuronske mreže na takav način je pregrubo definirana. Na primjer, ako se uzme skup za učenje veličine pet primjeraka, tada su sve mreže koje uspješno klasificiraju svih pet primjeraka jednako kvalitetne te proces učenja može završiti čim se pronađe jedna 100% uspješna konfiguracija mreže. Ako se takvoj neuronskoj mreži preda skup za ispitivanje od 100 primjera, tada se vidi koliko je tako pronađeno rješenje zapravo bilo nekvalitetno. Kada bi se uzela u obzir neka preciznija mjera, tada bi se ovaj problem mogao izbjeći.

Pretpostavlja se da promatrana neuronska mreža ima N različitih razreda za razvrstati. Za dani primjer "x" i njegovu ispravnu klasifikaciju "i" definira se očekivani vektor na izlazu iz neuronske mreže $\vec{e}(x)$ = (0, 0, ..., 0, 1, 0, ..., 0) gdje su sve vrijednosti jednake 0, osim i-te vrijednosti koja je jednaka 1. Dodatno se definira dobiveni izlaz iz neuronske mreže kao vektor $\vec{o}(x)$ = (o_1, o_2, ... o_N), ovisno o izlazima koje je mreža dala za dani primjer x. Sada se može definirati pojedinačna greška nad primjerom x:

$$error(x) \;=\; 0.5 f_x \sum_{i=1}^{N} (e_i(x) - o_i(x))^2, \tag{5.2}$$

gdje je f_x jednak 1 u slučaju regularnih klasa te 0,1 u slučaju "ništa od navedenog" klase. Ta posljednja klasa je regularna u smislu da je klasa kao i svaka druga, ali se ovim faktorom ipak daje veća prednost neuronskim mrežama koje uspješno klasificiraju više

primarnih klasa. Greške se množe faktorom 0,5 zato što je maksimalna moguća greška jednaka vrijednosti 2 pa se ovime vrijednosti skaliraju na [0, 1] interval.

Na temelju pojedinačne greške može se izračunati greška nad cijelim skupom primjera E, što ujedno i čini konačnu funkciju dobrote u izvedenom sustavu:

$$ERROR(E) = \frac{\sum_{x \in E} error(x)}{\sum_{x \in E} f_x} \tag{5.3}$$

5.3.2. Donošenje odluke unutar neuronske mreže

Prema prethodno uvedenoj notaciji, neuronska mreža za ulaz x daje kao izlaz vektor $\vec{o}(x)$. Na temelju izlaznog vektora neuronska mreža će svrstati primjer "x" u klasu "i" samo ako vrijedi:

$$o_i \geq o_j, \forall j : 1 \leq j \leq N, j \neq i \tag{5.4}$$

5.3.3. Donošenje odluke unutar skupa neuronskih mreža

Načiniti sustav koji prepoznaje sve novčanice neke valute nije prikladno koristeći regije i samo jednu neuronsku mrežu. Općenito, za regije se uzimaju područja visoke različitosti unutar skupa za učenje. Tako se za skup od sedam prednjih strana hrvatskih novčanica može uzeti regija na kojoj se jasno vidi vrijednost novčanice. No, u trenutku kada se želi uključiti i skup od sedam stražnjih strana hrvatskih novčanica, tada više nema smisla na istoj regiji promatrati i takve novčanice s obzirom na to da raspodjela karakterističnih elemenata nije jednaka za prednje i stražnje strane hrvatskih novčanica. Iz tog se razloga za skup prednjih strana novčanica promatraju određene regije, dok se za skup stražnjih strana tih istih novčanica koriste druge regije. Iz tog je razloga potrebno imati više neuronskih mreža za ostvarenje potpunog sustava za raspoznavanje. Sada se nameće pitanje – kako odgovore različitih neuronskih mreža objediniti u jedan zajednički odgovor cijelog skupa neuronskih mreža. Tek se u ovom trenutku uključuje logika rotacija unutar sustava.

Za donošenje odluke unutar skupa neuronskih mreža koristi se struktura ekstenzije ".man" opisana u poglavlju 5.1.3. Prije pokretanja primjera kroz različite neuronske mreže, sustav pripremi sve potencijalne odgovore koje može dati (zapisano u .man strukturi). Sustav također generira po jedan odgovor sustava za svaku rotaciju navedenog odgovora.

Konačno sustav dodaje i posljednji odgovor: "ništa od navedenog". Tako se početni skup od 14 različitih novčanica pretvara u skup od 29 mogućih različitih odgovora.

Jedna neuronska mreža može biti zadužena samo za podskup potencijalnih odgovora koje nudi sustav (na primjer, samo prednje strane hrvatskih kuna). Zato valja lokalnu oznaku odgovora neuronske mreže (0, 1, 2, ...) pretvoriti u globalnu oznaku klase koju koristi sustav. Postupak je sljedeći: svaki ulaz u sustav se pošalje na svaku neuronsku mrežu koja sudjeluje u odluci – dobiva se izlaz iz neuronske mreže $\vec{o}(x)$. Postupak se ponovi za rotirani ulaz – dobiva se novi izlaz iz neuronske mreže $\vec{r}(x)$.

Uvode se tri mjere, za tri različita slučaja:

- Normalna mjera: novčanica nije bila rotirana (pretpostavka)
- Rotirana mjera: novčanica je bila rotirana (pretpostavka)
- Preostala mjera: novčanica ne pripada niti jednoj klasi (pretpostavka)

Normalna mjera se računa za svaku klasu "i" koju neuronska mreža podržava (osim nesvrstane). Kako bi se izračunala normalna mjera neke klase, valja pomnožiti regularni izlaz iz neuronske mreže koji je dobiven za tu klasu s izlazom rotiranog ulaza, ali za klasu "ništa od navedenog". Ta se mjera računa pod pretpostavkom da novčanica nije bila rotirana pa će (ako je pretpostavka točna) obje vrijednosti biti visoke, što će u konačnici dati i visoku mjeru za ispravnu klasu. Mjera se računa prema formuli:

$$M_N(x,i) = o_i(x) * r_N(x) \tag{5.5}$$

Rotirana mjera se računa analogno normalnoj, ali pod pretpostavkom da je novčanica na ulazu bila rotirana. Mjera se također računa za svaku klasu "i" koju neuronska mreža podržava (osim nesvrstane). Mjera se definira na sljedeći način:

$$M_R(x,i) = o_N(x) * r_i(x) \tag{5.6}$$

Preostala mjera se računa samo jednom jer predstavlja slučaj da novčanica na ulazu ne spada u niti jednu klasu koju podržava trenutno promatrana neuronska mreža. Preostala mjera računa se prema formuli:

$$M_O(x) = o_N(x) * r_N(x) \tag{5.7}$$

Kada se izračunaju sve tri skupine mjera, tada se preslikavaju lokalni odgovori neuronske mreže u one globalne unutar sustava. Sustav prolazi kroz svaki mogući odgovor (unutar sustava) te uspoređuje svoj globalni kod s globalnim kodovima povezanim s lokalnom klasom neuronske mreže. Kada naiđe na poklapanje, tada promatranom odgovoru pridruži odgovarajuću mjeru. Normalne i rotirane mjere se koriste kod poklapanja odgovora, a preostala mjera se koristi kod nepoklapanja odgovora. U konačnici se prolaskom kroz sve neuronske mreže sumiraju vjerojatnosti svih potencijalnih odgovora. Kao konačni zaključak sustav ponudi onaj odgovor koji je prikupio najveću sumu njemu pripadajućih mjera. Sustav tako sumiranjem individualnih odluka svake neuronske mreže donosi zajedničku odluku svih mreža unutar sustava.

5.4. Ostale mogućnosti

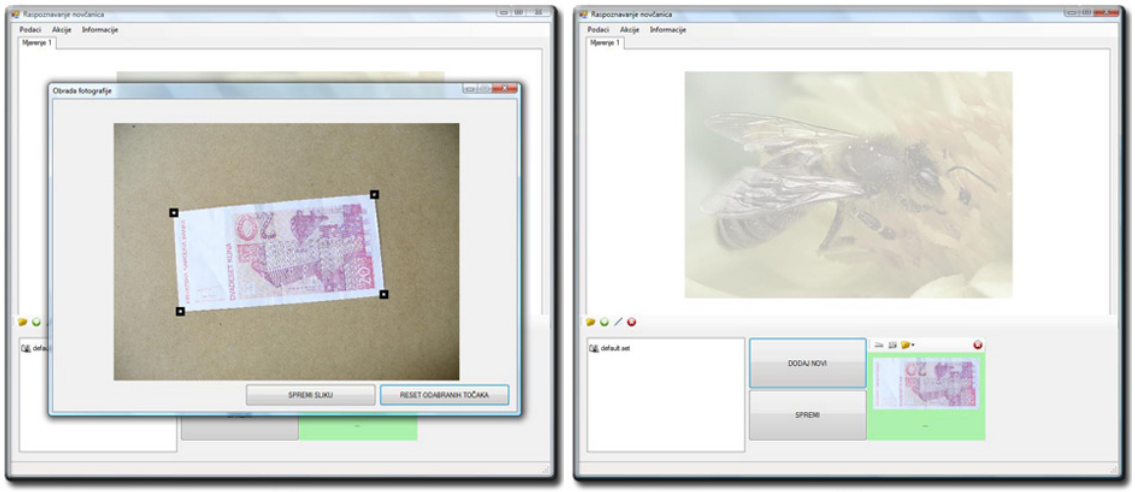
Do sada su opisane osnovne funkcionalnosti ključne za rad sustava. U ovom se poglavlju predstavljaju neke od preostalih mogućnosti unutar programa.

5.4.1. Obrada fotografija

Ostvareno programsko rješenje nudi mogućnost ručne ekstrakcije novčanice sa slike na kojoj se nalaze i drugi objekti (ili pozadina), pored same novčanice. Ova je opcija dodana s namjerom da se sustav može ispitati i nad fotografijama novčanica, za koje se ne može očekivati da su u pretpostavljenom formatu kao i skenirani primjerci koji su prvotno korišteni unutar sustava.

Obrada fotografija se obavlja unutar opcije "Upravljanje skupovima slika" nakon čega se odabere akcija "Stvaranje novog skupa slika". Dodavanjem novog primjera u skup, pritiskom na gumb "Obrada slike (fotografija)" može se postići opisana funkcionalnost. Prvo se otvara dijalog za odabir slikovne datoteke. Nakon odabira, slika se otvara u novom prozoru te korisnik zatim treba na slici označiti četiri kuta (rubne točke) novčanice unutar promatrane slike. Važno je da korisnik bira točke sljedećim redom: valja krenuti od lijevog gornjeg kuta novčanice te nastaviti birati kutove u smjeru kazaljke na satu. Odabrane četiri točke ne moraju činiti pravokutnik kako bi obrada bila uspješna. Odabirom bilo kojeg četverokuta odabrana se regija "razvlači" na pravokutnik ciljane veličine. Nakon što se slika tako obradi, ona se pohranjuje u mapu "Clips" pod jednakim nazivom kao i naziv originalne slike, ali se na početak naziva datoteke dodaje prefiks "_photo_". Original se ovim postupkom ne obriše nego se zadrži na originalnoj lokaciji ako korisnik želi ponoviti postupak.

Kako bi se opisana funkcionalnost mogla ostvariti korištena je transformacija četverokuta (engl. *quadrilateral transformation*) pomoću gotove funkcije unutar AForge biblioteke. Funkcija kao parametre preuzima četiri točke, a zatim se kao filter primjenjuje na željenu sliku.



(a) (b)

Slika 5.2 Primjer pokretanja obrade fotografije

Na slici 5.2 može se vidjeti primjer korištenja funkcionalnosti obrade fotografije. Prvo korisnik bira četiri točke na prethodno opisan način. Točke su označene crnim pravokutnicima (a). Nakon izvršenja algoritma prvo se prikaže uvećana dobivena slika, a zatim se korisnik zatvaranjem forme vraća do glavnog prozora gdje može vidjeti upravo dobivenu sliku koja se dodala u ispitni skup koji se trenutno izrađuje (b).

5.4.2. Prijedlog regija

Iz razloga što uspješnosti rada neuronskih mreža doprinosi kvalitetan odabir promatranih regija, a kako to može biti zamoran posao za korisnika, tada se uveo jednostavan algoritam pomoću kojeg sustav pronalazi potencijalno kvalitetne regije unutar promatranog skupa primjera. Ova metoda služi kao preporuka korisniku koji može taj savjet prihvatiti ili odbaciti.

Korisnik može zatražiti takav prijedlog odabirom naredbe "Prijedlog regija" u glavnom izborniku. Nakon toga se od korisnika traži odabir ispitnog skupa slika putem dijaloga. Tada algoritam počinje tražiti regije te rezultate prikazuje u prozoru koji ispisuje pronađene regije različitih veličina te novčanice iz skupa koji je promatran.

```
1   Inicijalizacija:
2       ZA SVAKI primjer (slika)
3         Obradi (slika)
4       KRAJ ZA SVAKI
5   Računanje matrica:
6       ZA SVAKI piksel (p)
7         InterMatrica[p] = 0
8         IntraMatrica[p] = 0
9         ZA SVAKI primjer (slikaA)
10          ZA SVAKI primjer (slikaB)
11            AKO Razred(slikaA) == Razred(slikaB) ONDA
12              IntraMatrica[p] += Razlika(slikaA[p], slikaB[p])
13            INAČE
14              InterMatrica[p] += Razlika(slikaA[p], slikaB[p])
15          KRAJ ZA SVAKI
16        KRAJ ZA SVAKI
17        Matrica[p] = InterMatrica[p] – koeficijent * IntraMatrica[p]
18      KRAJ ZA SVAKI
19  Pronalazak najbolje regije:
20      iteracija = 1
21      najboljaRegija = MinVrijednost()
22      PONAVLJAJ
23        širinaRegije = Random()
24        visinaRegije = Random()
25        ZA x = 0 DO (širinaMax – širinaRegije)
26          ZA y = 0 DO (visinaMax – visinaRegije)
27            sumaRegije = SumiranjeMatrice(x, y, širinaRegije, visinaRegije)
28            prosjekRegije = sumaRegije / (širinaRegije * visinaRegije)
29            AKO prosjekRegije > najboljaRegija ONDA najboljaRegija = prosjekRegije
30          KRAJ ZA
31        KRAJ ZA
32        iteracija = iteracija + 1
33        AKO (iteracija == Cmax) ONDA PREKINI
34      KRAJ PONAVLJAJ
```

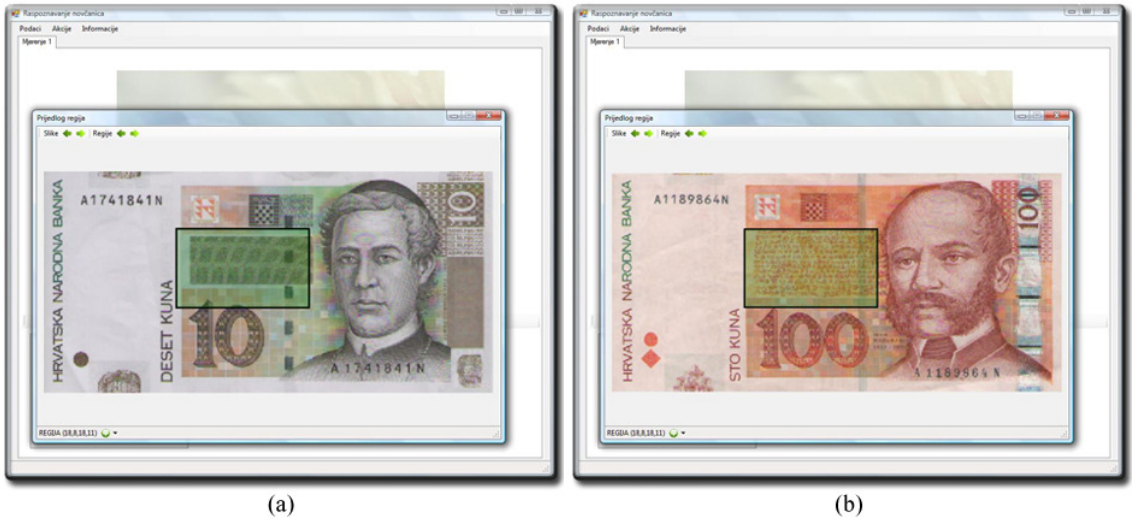
Slika 5.3 Pseudokod algoritma za pronalazak optimalnih regija

Osmišljeni algoritam automatskog odabira regije prikazan je u pseudokodu na slici 5.3. Algoritam započinje inicijalizacijom tako da se prvo obrade slike na uobičajen interni format opisan u poglavlju 5.2. U tom trenutku sustav više ne radi sa slikama, nego s matricom brojeva.

Tijekom koraka "računanje matrica" računaju se dvije matrice razlika. Prva matrica (intra-matrica) se računa unutar razreda. Naime, ako sustav radi s dva primjera novčanice od 10 kuna, tada nije poželjno uzeti regiju koja sadrži razlike unutar iste klase. Konkretno, ovom matricom se uklanjaju regije poput serijskih kodova novčanica. Druga matrica (inter-matrica) se računa između razreda. Sada je u interesu uzeti regiju koja sadrži velike razlike među različitim klasama. Konačna poželjnost uzimanja nekog piksela u obzir kao dio regije se računa linearnom kombinacijom intra-matrične te inter-matrične vrijednosti na poziciji promatranog piksela (slika 5.3, redak 17).

Na kraju treba pronaći skup piksela (koji tvore pravokutnu regiju) tako da se maksimizira omjer poželjnosti pripadajućih piksela i površine same regije. Posljednji korak se zove "pronalazak najbolje regije", gdje se u petlji nasumično biraju dimenzije regije za svaki korak. Nakon odabira dimenzija, prolazi se kroz sve moguće regije promatranih dimenzija te se izdvaja ona regija koja ima najveću prosječnu vrijednost poželjnosti regije (omjer sume poželjnosti i dimenzije regije). Funkcija SumiranjeMatrice (slika 5.3, redak 27) vraća sumu vrijednosti strukture Matrica počevši od gornje lijevog piksela (x, y) uz dimenzije širine "širinaRegije" te visine "visinaRegije". Cijeli postupak se prekida prelaženjem iteracijskog ograničenja (slika 5.3, redak 33).

Opisani postupak sadrži određenu dozu nasumičnosti i to od trenutka kada se traže regije. Izračuni matrica i poželjnosti piksela su strogo deterministički. Iz tog razloga uzastopno pokretanje algoritma nad istim skupom primjera ne daje nužno jednake rezultate, iako će najvjerojatnije biti približno jednaki.



(a) (b)

Slika 5.4 Prijedlog regije od strane sustava nad različitim novčanicama

Slika 5.4 prikazuje primjer regije koju je preporučio sustav na skupu prednjih novčanica hrvatskih kuna. Na lijevoj slici vidi se novčanica od 10 kuna te što na toj novčanici obuhvaća pronađena regija, što je označeno zelenim transparentnim pravokutnikom (a). Na desnoj slici je prikazana ista regija nad novčanicom od 100 kuna (b). Sustav je uistinu pronašao na skupu novčanica područje koje je vrlo karakteristično za svaku od klasa novčanica. Na dnu ekrana (status traka) može se vidjeti zapis trenutno promatrane regije, dok se pritiskom na gumb "+" koordinate transformiraju u [0, 1] interval te pohranjuju u međuspremnik da ih korisnik može dalje koristiti.

Korisnik unutar prozora "Prijedlog regija" može strelicama u gornjoj traci mijenjati trenutno promatranu sliku, odnosno trenutno promatranu regiju. Time se mogu dobiveni rezultati predloženih regija pregledati na jednostavan i intuitivan način.

5.4.3. Provođenje mjerenja

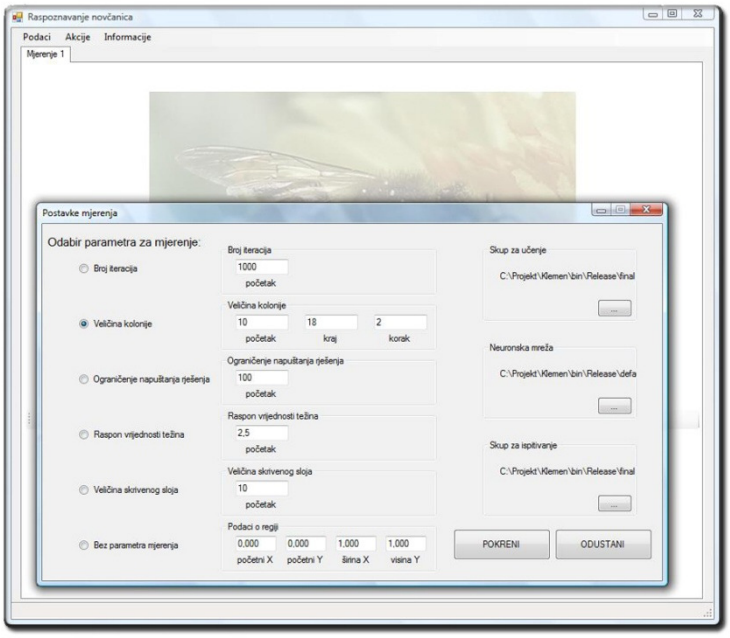
Ova se opcija koristila prilikom mjerenja u okviru ovog rada, čiji se rezultati mogu vidjeti u poglavlju 6.1. Akciju se poziva odabirom naredbe "Provođenje mjerenja" iz glavnog izbornika. Ova opcija služi za promatranje utjecaja parametara algoritma umjetne kolonije pčela te strukture umjetne neuronske mreže na konačnu kvalitetu dobivenog sustava. Mjerenja se izvode jednodimenzionalno i to na način da od pet ponuđenih parametara, njih četiri se uvijek fiksira, dok se peti mijenja i tada promatra ovisnost kvalitete o promatranom jednom parametru.

Mogući parametri mjerenja su: broj iteracija (koliko dugo će se algoritam pokretati), veličina kolonije (broj pčela koje traže rješenja), ograničenje napuštanja rješenja (nakon koliko iteracija bez poboljšanja će pčele napustiti neko rješenje), raspon vrijednosti težina (koje vrijednosti mogu poprimiti težinski faktori unutar neuronske mreže) te veličina skrivenog sloja (koliko se neurona nalazi u skrivenom sloju). Za parametar koji se želi mijenjati zadaju se tri parametra: početna vrijednost, završna vrijednost te korak (analogno for petlji). Algoritam se pokreće 1 + (završna vrijednost – početna vrijednost) / korak puta.

Tablica 5.5 Format zapisa izvješća o mjerenju

rbr	stavka	grupa	format
1	Broj iteracija unutar jednog pokretanja	parametri	INTEGER
2	Broj pokretanja		INTEGER
3	Veličina kolonije		INTEGER
4	Ograničenje za napuštanje dobivenog rješenja		INTEGER
5	Donja vrijednost težine unutar neuronske mreže		DOUBLE
6	Gornja vrijednost težine unutar neuronske mreže		DOUBLE
7	Broj neurona u skrivenom sloju		INTEGER
8	Početna pozicija regije na X osi		DOUBLE
9	Početna pozicija regije na Y osi		DOUBLE
10	Širina regije (X os)		DOUBLE
11	Visina regije (Y os)		DOUBLE
12	Datum početka pokretanja	mjere uspješnosti	DATE
13	Vrijeme početka pokretanja		TIME
14	Vrijeme kraja pokretanja		TIME
15	Formatirani zapis trajanja izvođenja		STRING
16	Trajanje izvođenja (u minutama)		INTEGER
17	Jednostavna mjera neuspjeha neuronske mreže		DOUBLE
18	Složena mjera neuspjeha neuronske mreže		DOUBLE

Za svako pokretanje se iscrtavaju grafovi, dok se posebni zapisi upisuju u datoteku "mjerenja.txt" i to u formatu kakav je prikazan u tablici 5.5, gdje je svaka stavka odvojena tab znakom, a u svakom redu je zapisano po jedno pokretanje algoritma unutar mjerenja.



Slika 5.5 Primjer biranja postavki za mjerenje

Slika 5.5 prikazuje primjer korištenja opisane funkcionalnosti. Na lijevoj se strani (šest radio gumba) bira po kojem se parametru provodi mjerenje. Za jednokratno mjerenje (bez promjena parametara) treba odabrati posljednju opciju "bez parametra mjerenja". U sredini su grupirani svi parametri, gdje je uvijek jedan zadan s trima vrijednostima, dok su ostali zadani s jednom. Kao posljednji parametar zadaju se "Podaci o regiji" i oni se ne mogu iterativno mijenjati ovim mjerenjem. Na desnoj strani se učitavaju skupovi za učenje neuronske mreže te njeno ispitivanje, kao i lokacija zapisa podataka o dobivenoj neuronskoj mreži (skup težinskih faktora). Sustav generira po jedan zapis o neuronskoj mreži za svako pokretanje unutar mjerenja tako da se odabranoj datoteci stavi indeks u zagradi na kraju njenog imena. Tako prvo pokretanje, uz odabranu datoteku "default.net" podatke o neuronskoj mreži zapisuje unutar datoteke "default(0).net".

5.4.4. Pronalazak optimalnih parametara

Provođenje mjerenja opisano u poglavlju 5.4.3 može biti naporan posao za korisnika. Želi li korisnik otkriti optimalnu kombinaciju svih pet parametara, tada bi se moralo provesti pet serija mjerenja (po jedna serija za svaki parametar), a i konačni zaključci ne bi nužno bili ispravni, s obzirom na to da se ne promatraju promjene nad više parametara istovremeno.

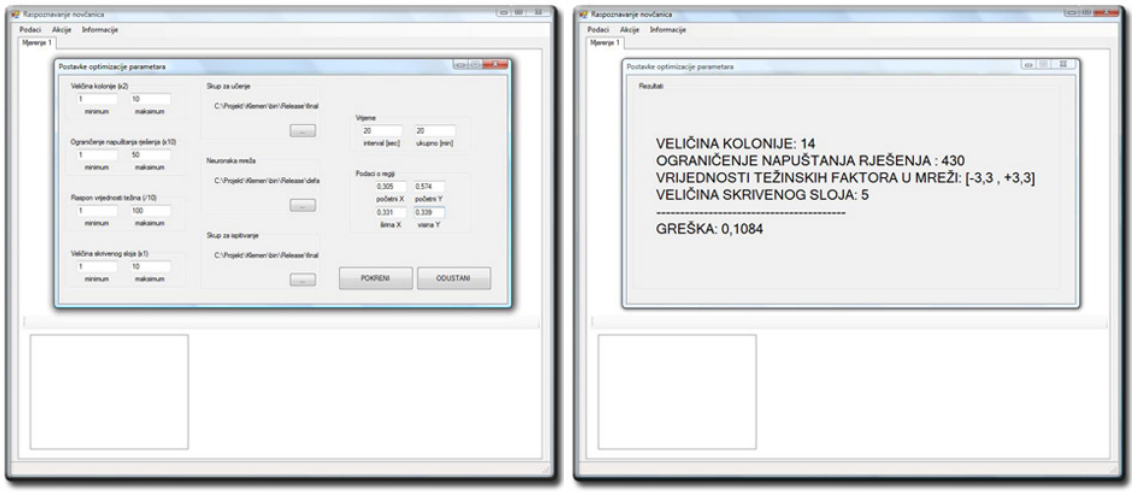
Zbog spomenutog nedostatka dana je mogućnost da sustav pronađe optimalne parametre putem jednog skupa mjerenja. Eventualni nedostatak ovakvog postupka su visoki vremenski zahtjevi. Pristup je sljedeći: korisnik zadaje skupove za učenje i ispitivanje te dozvoljene intervale vrijednosti za svaki promatrani parametar. Pokretanjem procesa paralelno rade dvije umjetne kolonije pčela. Prva kolonija ima zadatak optimirati tražene parametre, dok je druga pomoćna, koja služi kao evaluacija prve. Drugim riječima, koristi se jedan ABC algoritam koji uči parametre drugog ABC algoritma. Odnosno, koristi se uspješnost drugog algoritma kao funkcija dobrote prvog algoritma.

Može se postaviti logično pitanje – ako prvi algoritam trenira parametre drugog, kako onda odrediti parametre prvog algoritma? Parametri u ABC algoritmu ne utječu toliko na kvalitetu rješenja pod uvjetom da se algoritam ostavi dovoljno dugo pokrenut. ABC algoritam uglavnom sve nedostatke uslijed lošijih parametara može nadoknaditi dužim vremenom učenja. Inače to ne bi bilo prihvatljivo za općenito učenje prepoznavanja novčanica, ali za traženje optimalnih parametara takvo što može biti prihvatljivo, s obzirom na to da se takav proces treba pokrenuti samo jednom.

Parametri prvog algoritma su podešeni ovako: veličina kolonije = 10 (VK), ograničenje napuštanja rješenja = 5 (ONR), veličina skrivenog sloja = 10 (VSS) te broj pokretanja = 3 (BP). Trajanje pokretanja drugog algoritma može zadati korisnik (oznaka t) te se iskazuje u sekundama. Ukupno trajanje potrage za optimalnim parametrima (oznaka T) također zadaje korisnik te se iskazuje u minutama. Proces se prekida kada prođe više od dozvoljenog vremena T. Trajanje jedne iteracije vanjskog algoritma traje približno:

$$trajanje = BP * VK * t[\text{sec}] \tag{5.8}$$

Uz dane parametre jedna iteracija traje 30*t sekundi, odnosno 0,5*t minuta. Na tu formulu treba obratiti pozornost ako se stave premale razlike između intervala (t) i ukupnog trajanja (T) pa pravo trajanje izvođenja algoritma može biti različito od željenog.



(a) (b)

Slika 5.6 Priprema i rezultati pronalaska optimalnih parametara

Slika 5.6 prikazuje prozor za postavke optimizacije parametara (a) te rezultat nakon što sustav izvrši algoritam (b). Paralelno se zapisuju svi podaci o pokretanjima unutarnje neuronske mreže u "parametri.txt" datoteku prema formatu prikazanom u tablici 5.6, gdje su stavke odvojene tab znakovima.

Tablica 5.6 Format zapisa izvješća o potrazi za optimalnim parametrima

rbr	**stavka**	**grupa**	**format**
1	Veličina kolonije	parametri	INTEGER
2	Ograničenje za napuštanje dobivenog rješenja		INTEGER
3	Broj neurona u skrivenom sloju		INTEGER
4	Granica težine unutar neuronske mreže		DOUBLE
5	Prosječna mjera neuspjeha neuronske mreže		DOUBLE

6. Mjerenja i rezultati

U ovom su poglavlju provedena mjerenja o utjecaju parametara algoritma umjetne kolonije pčela na karakteristike izvođenja (uspješnost i vrijeme izvođenja). Nakon što su načinjene preporuke za pojedine parametre, ti parametri su korišteni u konačnoj provjeri uspješnosti sustava nad hrvatskim novčanicama od 10, 20, 50, 100, 200, 500 i 1000 kuna, uključujući prednje i stražnje strane te uz podršku rotacija od 180°. Sustav je provjeren nad 2 skupa novčanica – jedne su bile skenirane, dok su druge bile fotografirane. U nastavku su prikazani rezultati spomenutih mjerenja.

6.1. Utjecaj parametara na kvalitetu rješenja

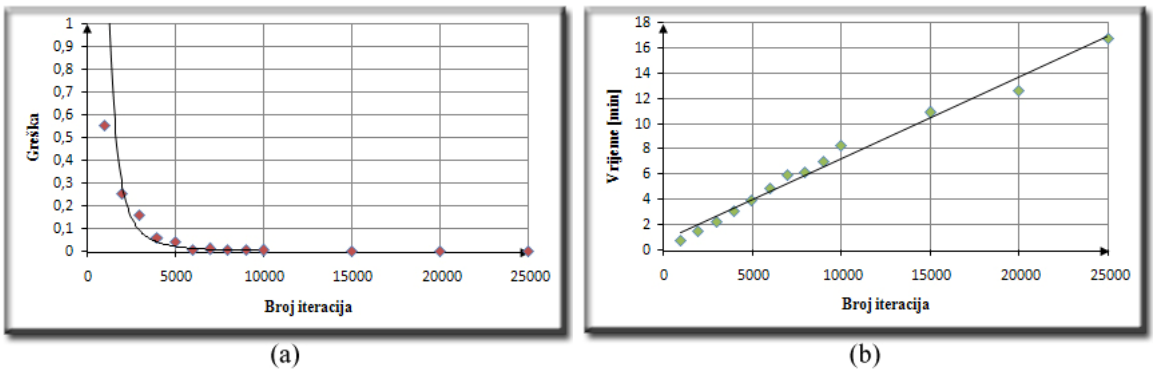
Za početak su provedena mjerenja nad parametrima algoritma umjetne kolonije pčela i strukture neuronske mreže kako bi se okvirno pronašli zadovoljavajući parametri za provjeru uspješnosti cijelog sustava prilikom prepoznavanja novčanica. Mjerenja su se provela nad četiri parametara ABC algoritma: broj iteracija, veličina kolonije, ograničenje napuštanja rješenja te raspon vrijednosti težina u neuronima. Peti parametar se odnosi na strukturu mreže – broj neurona u skrivenom sloju (ulazni i izlazni slojevi su već definirani).

Mjerenja su provedena uz korištenje već opisane funkcionalnosti u poglavlju 5.4.3. Dvije promatrane vrijednosti su greška mreže i vrijeme izvođenja algoritma. Mjerenje se provelo nad prednjim stranama novčanica hrvatskih kuna. Početne vrijednosti od kojih se krenulo su:

- Broj iteracija [BI] = 1000
- Veličina kolonije [VK] = 20
- Ograničenje napuštanja rješenja [ONR] = 100
- Raspon vrijednosti težina [RVT] = 2,5
- Veličina skrivenog sloja [VSS] = 10

6.1.1. Broj iteracija

Prvo se promatrao parametar broja iteracija. Što je parametar veći, to se algoritam duže vremena izvodi. Mjerene su vrijednosti od 1000 do 10000 (uz korak 1000) te od 10000 do 25000 (uz korak 5000).



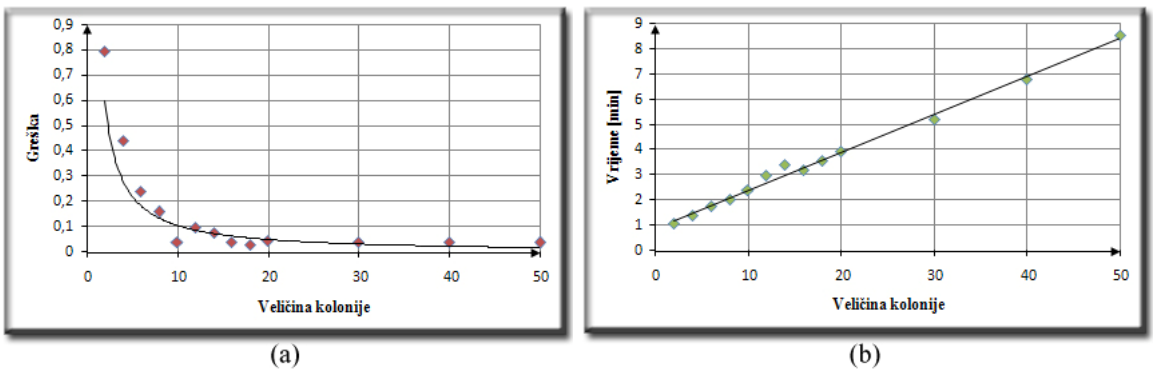
Slika 6.1 Utjecaj broja iteracija na grešku (a) i trajanje izvođenja (b)

Slika 6.1 prikazuje dobivene rezultate. Greška sustava se očekivano smanjuje s porastom broja iteracija (logaritamski), dok se vrijeme izvođenja povećava (linearno). Rezultati ovog mjerenja su najočitiji, ali zanimljivu informaciju predstavlja odgovor na pitanje koliko je iteracija dovoljno pokretati algoritam kako bi svaka nova iteracija davala manje poboljšanje nego što je uloženo vremena. Na slici 6.1 se vidi da već nakon 5000 iteracija rezultati su već toliko dobri da se teško mogu dalje popraviti. Iznad 10000 iteracija greška postaje manja od 0,001.

U sljedećim mjerenjima vrijednost iteracija se s početnih 1000 zamijenila s 5000 jer su rezultati puno bolji na 5000 iteracija. Ipak, tijekom mjerenja nije u interesu imati previše kvalitetne rezultate jer se kasnije ne bi vidjele nijanse između preostalih parametara.

6.1.2. Veličina kolonije

Sljedeći parametar na redu za mjerenje je veličina kolonije u ABC algoritmu. Kolonija se dijeli na 1/2 zaposlenih pčela, dok preostalu polovicu čine promatrači. Veličina kolonije je iz tog razloga parni (prirodni) broj. Izviđači se stvaraju po potrebi nakon napuštanja rješenja. Uz porast kolonije se očekuje smanjenje greške te povećanje vremena izvođenja. Mjerile su se vrijednosti od 2 do 10 (korak 2) te od 10 do 50 (korak 10).

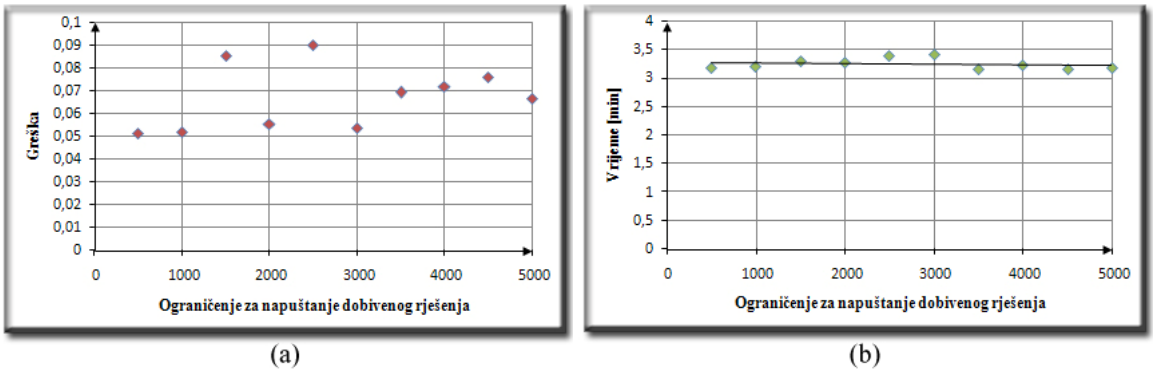


Slika 6.2 Utjecaj veličine kolonije na grešku (a) i trajanje izvođenja (b)

Rezultati mjerenja prikazani na slici 6.2 potvrđuju očekivanja o utjecaju veličine kolonije na grešku (a) i vrijeme izvođenja (b). Greška je ponovno u logaritamskoj ovisnosti, dok je vrijeme u linearnoj. Dodavanjem više pčela u koloniju produžuje se trajanje pojedine iteracije te se koloniji daje više mogućnosti da pronađe kvalitetnija rješenja. Promatranjem (a) dijela slike može se vidjeti da se već za 10 pčela (5 istovremeno promatranih rješenja) postižu dobri rezultati. U nastavku mjerenja se koristi 16 pčela.

6.1.3. Ograničenje za napuštanje dobivenog rješenja

Sljedeći na redu je parametar ograničenja za napuštanje dobivenog rješenja. Ovdje se postavlja pitanje – nakon koliko iteracija bez da su uspješno unaprijedile već postojeće rješenje trebaju pčele odustati i potražiti u potpunosti novo rješenje. Mjerene su vrijednosti od 500 do 5000 (korak 500). Mjerenja su provedena tri puta nakon čega su se koristile srednje vrijednosti zbog oscilacija unutar pojedinih mjerenja.



(a) (b)

Slika 6.3 Utjecaj ograničenja za napuštanje rješenja na grešku (a) i trajanje izvođenja (b)

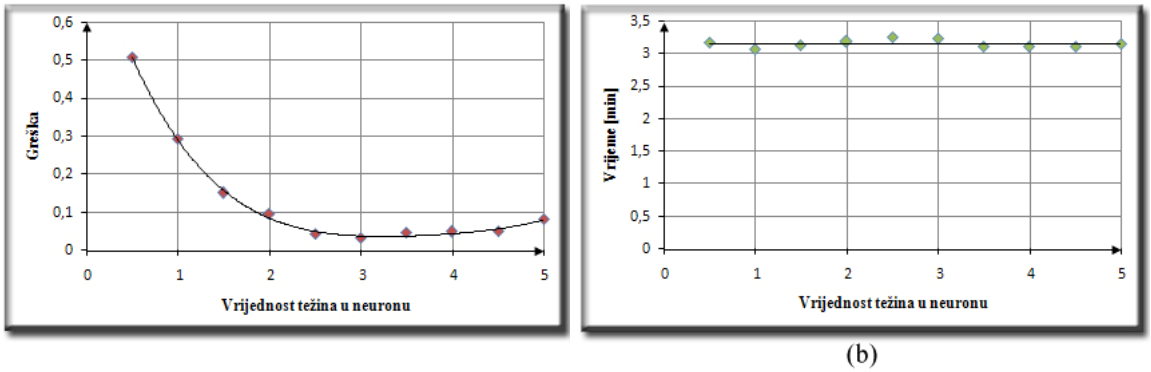
Slika 6.3 prikazuje dobivene rezultate. U promatranim mjerenjima greške nije primijećen uočljivi trend. Mjerenje trajanja izvođenja algoritma ukazuje na neovisnost vremenske složenosti algoritma o promatranom parametru. Pri malim vrijednostima ograničenja za napuštanje dobivenog rješenja postoji linearni pad vremenske složenosti kako se parametar povećava. To se objašnjava učestalošću stvaranja izviđača. Što je ograničenje manje, to se češće pozivaju izviđači da generiraju nova rješenja, odnosno – za manja ograničenja napuštanja potrebno je više ukupnog vremena. Pri većim vrijednostima ograničenja za napuštanje rješenja stvaranje izviđača postaje zanemarivo.

No, kod promatranja ovakvih rezultata treba uzimati u obzir i ukupni broj težinskih faktora koje neuronska mreža pokušava optimirati. U ovim mjerenjima je korištena 220x10x8 neuronska mreža, odnosno 2298 težinskih faktora. Kada bi se već nakon 100 iteracija dopustilo da se rješenje napusti, to bi značilo da se promatra svega nasumičnih 4%

mreže kako bi se pokušalo popraviti promatrano rješenje. U nastavku se uzelo ograničenje za napuštanje dobivenog rješenja od 500 iteracija.

6.1.4. Raspon vrijednosti težina u neuronima

Slijedi pitanje na koje vrijednosti trebaju pčele postavljati vrijednost težina veza između neurona unutar mreže. Obično postoje donja i gornja granica spomenutih vrijednosti, ali se ovdje radi jednostavnosti uzeo simetričan interval oko nule. Tako su vrijednosti iz intervala [-vrijednost, +vrijednost]. U obzir su došle vrijednosti od 0,5 do 5 (korak 0,5). Mjerenja su provedena tri puta te su u konačnici analizirane srednje vrijednosti.



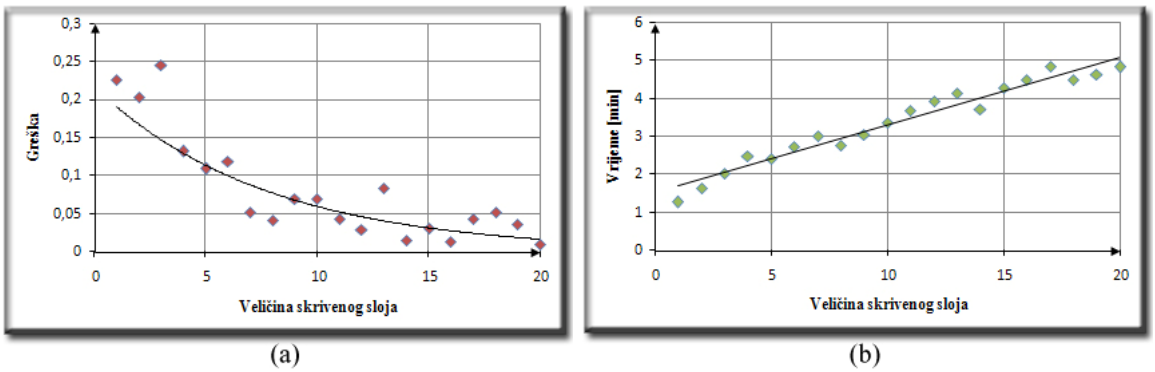
(b)

Slika 6.4 Utjecaj raspona vrijednosti težina na grešku (a) i trajanje izvođenja (b)

Slika 6.4 prikazuje dobivene rezultate. Porastom dozvoljenog raspona vrijednosti težina neuronske mreže greška se prvo smanjuje, a zatim lagano povećava. Posebno su loši rezultati za premale vrijednosti što se objašnjava nemogućnošću neuronske mreže da naglasi važnost određenih veza jer su na raspolaganje pružene premale vrijednosti. Vrijednosti težina ne utječu na dodatnu vremensku složenost unutar algoritma, što je i vidljivo na b dijelu slike 6.4. U nastavku se uzimao interval [-3, +3].

6.1.5. Broj neurona u skrivenom sloju

Konačno, preostaje parametar koji nema veze sa samim ABC algoritmom, već pitanjem dizajna neuronske mreže. Ulazni sloj neuronske mreže ovisi o broju piksela promatrane regije. Izlazni sloj neuronske mreže ovisi o broju klasa koje mreža treba prepoznati. Ostaje pitanje kojih dimenzija treba biti skriveni sloj. Uzme li se premali broj neurona, tada se ne dobiva dovoljno složena neuronska mreža. Uzme li se previše neurona, tada neuronska mreža postaje nepotrebno kompliciranа pa je potrebno više vremena da se njezini težinski faktori usavrše. Promatrale su se vrijednosti između 1 i 20 neurona (uz korak 1).



Slika 6.5 Utjecaj veličine skrivenog sloja na grešku (a) i trajanje izvođenja (b)

Slika 6.5 potvrđuje prethodne pretpostavke. Greška pada eksponencijalno s porastom veličine skrivenog sloja jer se pruža više mogućnosti neuronskoj mreži da izrazi pripadnost novčanice određenoj klasi. S druge strane, veći broj neurona povlači više težinskih faktora. Iz tog razloga vrijeme izvođenja linearno raste s porastom veličine skrivenog sloja. Broj neurona se preporučuje iznad 15.

6.2. Rezultati skeniranih primjeraka

Prilikom ispitivanja rada sustava korišteni su sljedeći parametri:

- Broj iteracija [BI] = 25000
- Veličina kolonije [VK] = 16
- Ograničenje napuštanja rješenja [ONR] = 100
- Raspon vrijednosti težina [RVT] = 3 (od -3 do +3)
- Veličina skrivenog sloja [VSS] = 15

Primjerci novčanica skenirali su se na uređaju Epson Stylus SX100 uz rezoluciju 100 DPI.

6.2.1. Skup za učenje

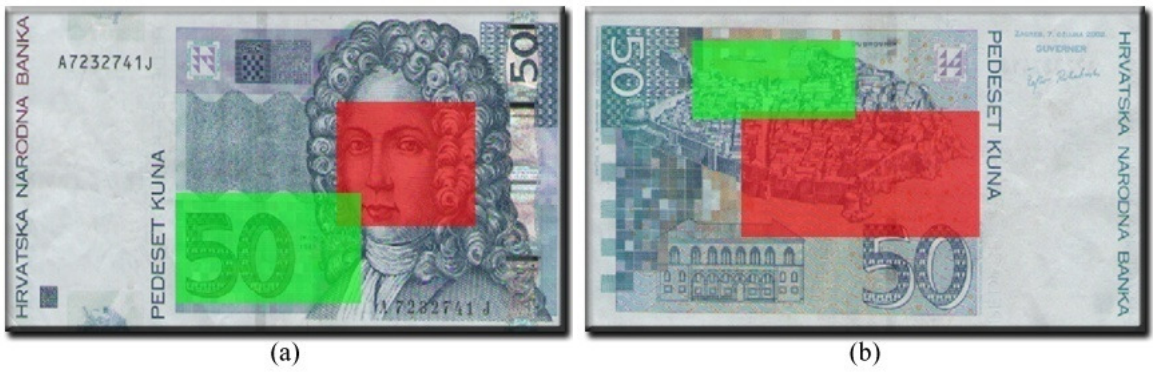
Za učenje neuronske mreže korištena su dva primjerka svake hrvatske novčanice (10, 20, 50, 100, 200, 500 i 1000), prikazane iz četiri različite pozicije (prednja ispravna, prednja rotirana, stražnja ispravna i stražnja rotirana). Što znači da ukupna veličina skupa za učenje iznosi 2*7*4 = 56 primjera.

6.2.2. Skup za ispitivanje

Za ispitivanje neuronske mreže korištena su pet primjerka svake hrvatske novčanice, što daje ukupno 5*7*4 = 140 primjera (u koje su uključeni i primjeri iz skupa za učenje). Nad ovim se skupom provjerila uspješnost neuronske mreže koja tijekom učenja nije vidjela 60% primjera iz ovog skupa.

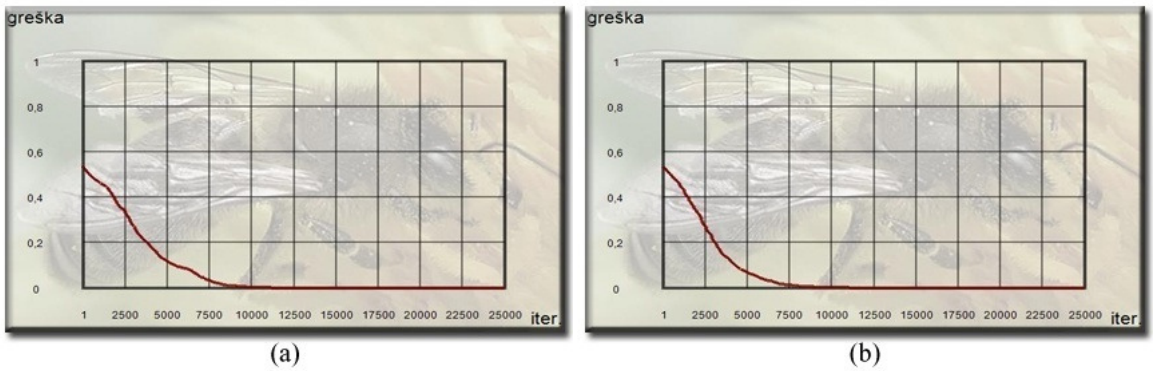
6.2.3. Učenje neuronskih mreža

Korištene su po dvije regije za svaku stranu novčanica. Regije se mogu vidjeti na slici 6.6. Prednje strane su koristile regije koje sadrže novčanu vrijednost (zelena regija) te lice osobe (crvena regija). Stražnje strane su koristile regije u oba slučaja vezane za glavnu pozadinsku sliku. Jednom je korištena veća (crvena regija), a drugi put manja površina (zelena regija).

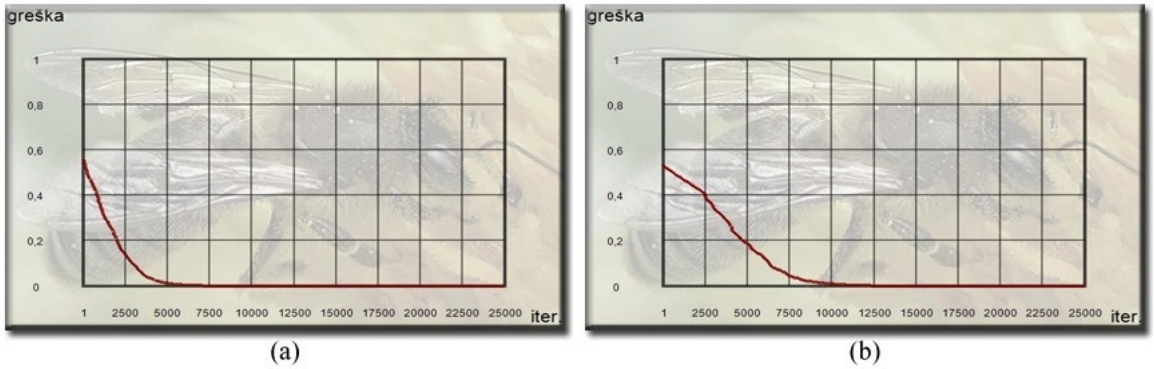


Slika 6.6 Odabrane regije za prednje strane (a) i stražnje strane novčanica (b)

Svaka od četiri regija trebala je istrenirati vlastitu neuronsku mrežu. Prikaz učenja može se vidjeti na slikama 6.7 (prednja strana novčanice) i 6.8 (stražnja strana novčanice). Sve četiri regije su bez većih problema uspješno istrenirale neuronsku mrežu nad skupom za učenje unutar 10000 (od ukupnih 25000) iteracija.



Slika 6.7 Proces učenja neuronskih mreža prednjih novčanica za prvu (a) i drugu regiju (b)



Slika 6.8 Proces učenja neuronskih mreža stražnjih novčanica za prvu (a) i drugu regiju (b)

Na kraju su kao sredstvo provjere prije konačne ocjene ispitane sve četiri neuronske mreže nad skupom za ispitivanje. Rezultati koji su pohranjeni u datoteku "mjerenja.txt" prilikom učenja mogu se vidjeti u tablici 6.1. Sve su neuronske mreže uspjele 100% klasificirati skup za ispitivanje, osim stražnje (a) koja je jedan primjerak krivo klasificirala. Sve neuronske mreže su ukupno učile na primjerima 42 minute.

Tablica 6.1 Zapisi o procesu učenja regija i njihovim trajanjima i greškama

Regija	BI	VK	ONR	RVT -	RVT +	VSS	Trajanje	Greška (gruba)	Greška (složena)
Prednja (a)	25000	16	100	-3	3	15	0:10:35	0,0000	0,0001
Prednja (b)	25000	16	100	-3	3	15	0:09:40	0,0000	0,0001
Stražnja (a)	25000	16	100	-3	3	15	0:08:20	0,0071	0,0066
Stražnja (b)	25000	16	100	-3	3	15	0:13:02	0,0000	0,0025

6.2.4. Uspješnost sustava

Konačno, sve su četiri regije objedinjene unutar upravljačke strukture kako bi se donijela konačna odluka na temelju skupine od četiri neuronskih mreža. Rezultati se mogu vidjeti u tablici 6.2, u prvom retku koji se odnosi na ovaj tip slika (skenirane). Sustav je postigao 100% uspješnosti kada se sustavu pomoglo tako da se isključi "ništa od navedenog" odgovor. Naime, kako je svaka regija imala vlastiti "ništa od navedenog" odgovor, tako unutar sustava postoji i zajednički "ništa od navedenog" odgovor. No, kako se zna da su se prilikom ovih mjerenja sustavu davale samo hrvatske novčanice, tada se sustavu moglo pomoći uz pretpostavku da se svi primjerci uistinu nalaze u jednoj od 14 postojećih klasa. Ako se od sustava traži da se uvede i petnaesta klasa (za sve preostale), onda sustav za tri hrvatske novčanice pogrešno kaže da ne pripadaju niti jednoj klasi hrvatskih novčanica. U tom je slučaju uspješnost sustava 98%. Naravno, u takvom

scenariju, sve tri neispravno klasificirane novčanice imaju kao najvjerojatniji odgovor "ništa od navedenog", dok je ispravni odgovor u sva tri slučaja drugi po redu dobiven kao rezultat (što objašnjava 100% uspješnost kada se preostala klasa isključi).

Postupak klasifikacije je brz u usporedbi s vremenom pripreme ispitnih primjera, na koji odlazi većina resursa tijekom ispitivanja. Tijekom provjere rada sustava izmjereno je prosječno vrijeme klasifikacije od 0,41 sekunde po jednoj novčanici.

Tablica 6.2 Rezultati uspješnosti sustava za skenirane i fotografirane primjere

	Uključujući "ništa od navedenog" odgovor			**Isključenjem "ništa od navedenog" odgovora**		
Tip slika	**Uspješna klasifikacija**	**Veličina skupa**	**Postotak uspješnosti**	**Uspješna klasifikacija**	**Veličina skupa**	**Postotak uspješnosti**
Skenirane	137	140	**98%**	140	140	**100%**
Fotografirane	78	112	**70%**	105	112	**94%**

6.3. Rezultati fotografiranih primjeraka

U ovoj skupini mjerenja uzeo se skup istreniranih neuronskih mreža iz prethodnog mjerenja te se ovdje ispitao nad fotografiranim novčanicama. Novčanice su se slikale digitalnim fotoaparatom Panasonic DMC-LZ3 na rezoluciji od 72 DPI. Skup za učenje preuzet je iz prethodnog mjerenja, odnosno – za ispitivanje fotografija koristio se skup neuronskih mreža koje su učile nad skeniranim slikama.

6.3.1. Skup za ispitivanje

Za ispitivanje korištena su četiri primjerka svake hrvatske novčanice, što daje ukupno 4x7x4 = 112 primjera. Svih 112 primjera neuronske mreže nisu vidjele tijekom učenja. Valja istaknuti kako se skenirane novčanice razlikuju od fotografiranih jer skener pritisne novčanicu te ju izravna, što nije slučaj kod fotografija. Fotografije su se prije predaje na ispitivanje obradile u smislu da su označeni rubovi novčanice pomoću opcije opisane u poglavlju 5.4.1.

6.3.2. Uspješnost sustava

Rezultati se mogu vidjeti u tablici 6.2, u drugom retku (tip slika: fotografirane). Očekivano, sustav se ponaša lošije u ovom slučaju iz nekoliko razloga: ispituju se fotografirane novčanice nad neuronskih mrežama koje su učene nad skeniranim novčanicama. S druge strane, zgužvanost novčanica uvelike odmaže ovom procesu.

Dobiveni rezultati iznose 94% uspješnosti ako se promatra samo skup hrvatskih novčanica (sedam pogrešno klasificiranih primjera). U slučaju kada se od sustava traži da se izjasni i po pitanju preostale (petnaeste) klase, tada u 30% slučajeva daje pogrešan odgovor. Takav rezultat je poprilično loš, ali se treba naglasiti kako raspoznavanje uzoraka putem neuronskih mreža nije predviđeno za veće oscilacije na ulaznim slikama. Neuronska mreža očekuje određene elemente na točnim mjestima, a zgužvanost novčanica kod fotografija očito utječe na sposobnost mreže da donese ispravan odgovor. No, uspješnost od 94% unutar skupa hrvatskih novčanica u takvim uvjetima je zadovoljavajuća. Iz tog se razloga preporučuje da, ako se ispituje klasifikacija fotografiranih primjera, da se obavezno isključi opcija "ništa od navedenog".

7. Zaključak

Algoritam umjetne kolonije pčela pokazao se uspješnim pri rješavanju problema raspoznavanja novčanica. Iako algoritam nema kompliciranu matematičku podlogu, to ga ne zaustavlja da uspješno optimira težinske faktore neuronske mreže.

Klasično bi se problem raspoznavanja uzoraka rješavao neuronskom mrežom koju uči postupak poput algoritma propagacije greške unatrag. Umjetne pčele nemaju potrebe propagirati greške unatrag, već svojom brojnošću (veličina kolonije) i upornošću (broj iteracija) pronalaze optimalno rješenje kombinacijom nasumičnog istraživanja te intenzivne pretrage okoline u područjima veće kvalitete rješenja.

Uvođenjem ekstrakcije regija sa slika prije prolaska kroz neuronsku mrežu osigurava se laka proširivost sustava, gdje se na postojeći sustav mogu lako dodati druge valute s vlastitim specifičnim regijama od interesa.

Zadatak postavljen u poglavlju 4.3.1 je obavljen – sustav klasificira jednu novčanicu za 0,41 sekunde, dok je postignuta uspješnost od 100% za skenirane i 94% za fotografirane hrvatske novčanice, ako se promatraju samo postojeći skupovi.

POTPIS:

_____

8. Literatura

[1] P. Lucic; D. Teodorovic (2001). "Bee system: Modeling combinatorial optimization transportation engineering problems by swarm intelligence". In Preprints of the Tristan IV Triennial Symposium on Transportation Analysis, SaoMiguel, Azores Islands, Portugal.

[2] D. Teodorovic; M. O. Dell (2005). "Bee colony optimization - a cooperative learning approach to complex transportation problems". In Proceedings of 10th EWGT Meeting and 16th Mini EURO Conference.

[3] H. S. S. Drias; S. Yahi (2005). "Cooperative bees swarm for solving the maximum weighted satisfiability problem". In Computational Intelligence and Bioinspired Systems, volume 3512/2005 of LNCS: 318 – 325, Springer, Berlin.

[4] D. T. Pham; A. Ghanbarzadeh; E. Koc; S. Otri; S. Rahim; M. Zaidi (2005). "The bees algorithm". Technical report, Manufacturing Engineering Centre, Cardiff University, UK.

[5] D. Karaboga (2005). "An idea based on honey bee swarm for numerical optimization". Technical Report TR06, Erciyes University, Engineering Faculty.

[6] K. V. Frisch (1973). "Nobel Lecture: Decoding the Language of the Bee". University of Munich, Federal Republic of Germany.

[7] "Detailed Pseudocode of the ABC Algorithm" (2008). http://mf.erciyes.edu.tr/abc/index.htm, 7.5.2010.

[8] W. McCulloch; W. Pitts (1943). "A logical calculus of the ideas immanent in nervous activity". Bulletin of Mathematical Biophysics, 7: 115 - 133.

[9] M. Minsky; S. Papert (1969). "Perceptrons". MIT Press, Cambridge.

[10] M. Čupić; J. Šnajder (2001). "Web-orijentirani simulator modela neuronskih mreža s primjerom raspoznavanja novčanica i generatorom uzoraka", FER.

[11] T. Mitchell (1997). "Machine Learning". McGraw Hill. 2. stranica.

[12] R. Duda; P. Hart; D. Stork (2001). "Pattern Classification". New York: John Wiley & Sons.

[13] J. Canny (1986). "A computational approach to edge detection". Pattern Analysis and Machine Intelligence.

RASPOZNAVANJE UZORAKA PRIMJENOM UMJETNE KOLONIJE PČELA

Sažetak

U ovom diplomskom radu je istražen problem raspoznavanja uzoraka, odnosno novčanica. Zadatak se rješava uz pomoć umjetne kolonije pčela koja optimira težinske faktore umjetne neuronske mreže koja služi za klasifikaciju novčanica. U praktičnom dijelu rada ostvarena su mjerenja ovisnosti uspješnosti sustava o parametrima učenja. Ti su se parametri dalje iskoristili za ispitivanje nad skupom hrvatskih novčanica koje su prikupljene na dva načina. Skenirane novčanice postigle su uspješnost od 100% (98%), dok su fotografirane novčanice postigle uspješnost od 94% (70%).

Ključne riječi: umjetna kolonija pčela, raspoznavanje uzoraka, umjetna neuronska mreža

Abstract

This graduation thesis deals with pattern recognition problem, or recognition of banknotes to be precise. Given task is solved by using artificial bee colony to optimize weight factors in artificial neural network which is responsible for banknote classification. Practical part of this thesis includes analysis of dependence between system's success and parameters of training process. Parameters retrieved in such way were later used for system testing on a set of Croatian banknotes in two ways: scanned banknotes were 100% (98%) successfully classified, while photographed banknotes achieved 94% (70%) success rate.

Keywords: artificial bee colony, pattern recognition, artificial neural network