UNIVERSITATEA ”LUCIAN BLAGA” DIN SIBIU

Contract UEFISCDI nr. 27 / 04.08.2010

Programul: Resurse umane
Tipul proiectului: Proiect de cercetare post-doctorală
Cod proiect: PD_670 / 28.07.2010

SINTEZA DE CERCETARE

SISTEM DE CLASIFICARE AUTOMATA A DATELOR NESTRUCTURATE FOLOSIND METACLASIFICATOARE BAZATE PE METODE DE TIP SUPPORT VECTOR MACHINE ŞI NAIVE BAYES

Etapa unica 2011
Anul II

DIRECTOR PROIECT,

Sef lucrari dr. Ing. Daniel MORARIU

decembrie, 2011

1 Rezumatul proiectului

Acest proiect se încadrează în domeniul clasificării automate a documentelor text şi îşi propune îmbunătăţirea rezultatelor clasificării prin realizare unor strategi de combinare a rezultatelor metodelor de clasificare folosite.

Un număr impresionant de documente se găsesc în format electronic, iar utilizatorul are nevoie de componente de clasificare automată a acestora pentru a le putea gestiona. Gestiunea lor a devenit o problema foarte importanta în ultima perioada. Devine esenţială existenta unor programe inteligente de organizare automată a documentelor în categorii pentru a facilita analiza şi prelucrarea acestor documente. Datorită domeniului foarte vas în care ar trebui să lucreze acestea, devine dificil de realizat un singur clasificator cu performanţe foarte bune. Abordarea actuala este de a utiliza mai mulţi clasificatori de diferite tipuri combinaţi într-un meta-clasificator, sau realizarea unei clasificări hibride care se bazează pe predicţia clasificatorului cel mai bun pentru o problema particulară folosind caracteristicile vectorilor de intrare ale documentelor și istoria clasificărilor. Având mai mulţi clasificatori de baza, de tipuri diferite (SVM - Support Vector Machine, Bayes, reţele neurale, etc), idea este de a învăţa un meta-clasificator care prezice gradul de corectitudine pentru fiecare dintre clasificatorii de bază. Meta-etichetarea unei instanţe indică încrederea în clasificarea făcută de acesta, dacă instanţa este clasificata corect de către acel clasificator dintre toţi ceilalţi clasificatori utilizaţi. Regula de clasificare a meta-clasificatorului este ca fiecare clasificator de baza să atribuie o clasa la instanţa curentă şi apoi meta-clasificatorul să decidă dacă clasificarea este demnă de încredere sau nu. Pe lângă creşterea acurateţei de clasificare, prin exploatarea sinergismului mai multor clasificatoare, un alt avantaj al meta-clasificării constă în posibilitatea de exploatare a paralelismelor funcţionale (multiprocesor).

2 Obiectivele proiectului pentru anul 2011

Etapa	Obiective	Activităţi	Categorii de buget (Valoare lei)
Unică	1. Design Space Exploration pentru meta-clasificatorul neadaptiv	1.1 Realizarea unei explorări a spaţiului problemei folosind algoritmi de căutare euristica.	16500,00
		1.2 Implementarea algoritmilor euristici şi găsirea celor mai bune valori de ponderare. Realizare practica	14540,00
	2. Implementarea unui meta-clasificator adaptiv bazat pe reţele neurale	Reprezentarea vectorului documentului de intrare prin agregarea vectorilor de clase generaţi de către fiecare clasificator in parte.	15000,00
		Dezvoltarea unui meta-clasificator neuronal adaptiv. Realizare practica.	17000,00
			Total etapa: 63040,00

3 Sinteza activităţilor de cercetare realizate în 2011

3.1 Setul de date utilizat în experimente

Experimentele prezentate sunt efectuate folosind colecţia de date Reuters-2000 [Reut00], pe un set de 7053 documente, care au fost împărţite în mod aleatoriu în setul de antrenament (4702 eşantioane), numit în continuare setul A1, şi setul de test (2351 eşantioane), numit în continuare setul T1, care a fost prezentat mai în detaliu în sinteza pe anul 2010.

3.2 Meta-clasificator - Arhitecturi neadaptive propuse şi dezvoltate

Meta-clasificatorul propus în continuare conţine cei 9 clasificatori prezentaţi în sinteza din anul 2010 şi pleacă de la premisa că nu contează doar clasa învingătoare ci toate clasele întoarse de

către meta-clasificator şi locul pe care apare fiecare clasă în parte. De exemplu, în cazul a doi clasificatori şi 3 clase, dacă o clasă apare o dată pe locul 1 şi o dată pe locul 4 şi o altă clasă apare de 2 ori pe locul 2, este posibil ca cea de-a doua clasă să fie mai valoroasă, chiar dacă nu a obţinut niciodată locul 1.

3.2.1 Meta-clasificator neadaptiv bazat pe sumă

În meta-clasificatorul dezvoltat sunt incluşi 8 clasificatori de tip SVM (Support Vector Machine) şi unul de tip Naive Bayes. Fiecare dintre aceştia produce la ieşire un vector care conţine 16 scalari, vezi Fig. 3.1. Fiecare scalar reprezintă valoarea funcţiei de decizie a clasificatorului pentru clasa respectivă. Până acum, în [Mora08] se alegea întotdeauna valoarea cea mai mare şi clasa corespunzătoare a acestei valori era considerată clasa pe care o prezice clasificatorul respectiv. Pentru fiecare document în parte vom obţine acum 9 astfel de vectori fiecare cu 16 valori.

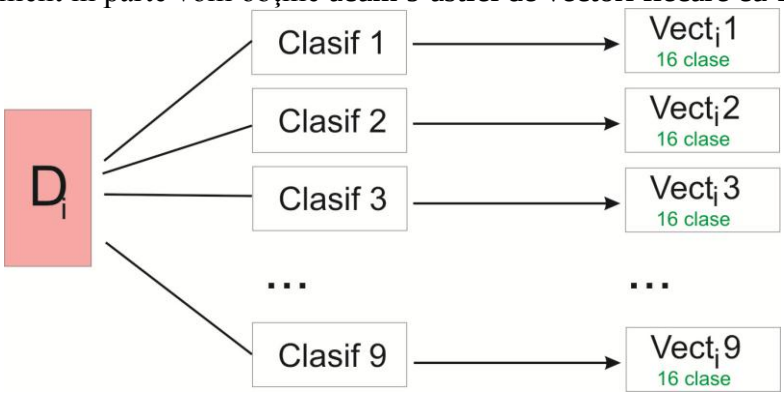

Fig. 3.1 Meta-clasificator neadaptiv bazat pe sumă

Valorile funcţiilor de decizie pentru clasificatorii de tip SVM se află în intervalul (-∞,∞) dar în apropierea valorii 0, iar pentru clasificatorul de tip Bayes valorile se află în intervalul (-∞, 0). Având în vedere aceste diferenţe şi pentru a putea realiza însumarea valorilor vectorilor, am transpus valorile vectorilor în intervalul [1, ∞).

$$V_i = V_i + \left|\min(\vec{V})\right| + 1 \qquad (3.1)$$

Astfel, pentru fiecare vector, diferenţele între valorile vectorilor de ieşire ai clasificatorilor de tip SVM se păstrează. La fel şi pentru clasificatorul de tip Bayes. Pentru a putea realiza însumarea acestor vectori, în următorul pas am normalizat vectorii, aducând valorile acestora în intervalul (0,1].

În cazul meta-clasificatorului care realizează doar sumele (numit in continuare M-SUM), am însumat cele 16 valori ale acestor 9 vectori, vezi Fig. 3.2, clasa câştigătoare fiind clasa cu valoarea cea mai mare obţinută.

$$Class = \max_{c_i, i=1,16} \sum_{k=1}^{9} V_i[k] \qquad (3.2)$$

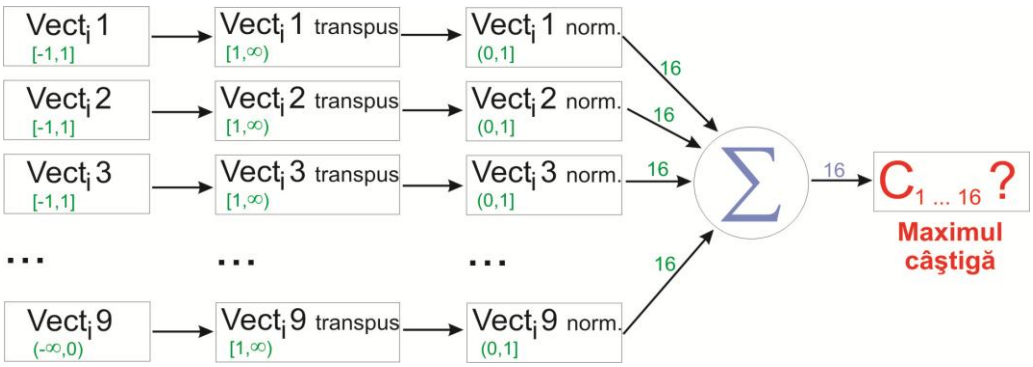

Fig. 3.2 Meta-clasificator neadaptiv bazat pe sumă (M-SUM)

Acest meta-clasificator bazat pe sumă, fiind unul neadaptiv, va obţine întotdeauna acelaşi rezultat pentru o anumită instanţă de intrare. În cazul rulării pe cele 2351 documente de test (setul T1), am obţinut un număr de **313** documente clasificate eronat, care reprezintă o acurateţe a

clasificării de **86.68%,** cu **0.59%** mai mare decât valoarea obţinută, folosind votul majoritar şi toţi cei 9 clasificatori. Astfel putem concluziona că metoda bazată pe luarea în considerare doar a clasei învingătoare (*majority vote*), are dezavantaje faţă de metoda prezentată mai sus. În acest caz există şansa ca o clasă care poate niciodată nu a obţinut locul 1, dar a obţinut valori apropiate de maxim, să fie în final clasa corectă.

Am realizat o serie de cercetări privind diferite ponderări a elementelor vectorilor. În cazul primei ponderări testate vom asigna în fiecare vector pentru clasa de pe locul 1, valoarea 12, pentru clasa de pe locul 2, valoarea 10, iar în continuare, pentru fiecare clasă de pe următoarele locuri, valori descrescătoare până la valoarea 1 (abordarea implementează soluţia consacrată în concursul european de muzică uşoară EUROVISION). În urma acestei ponderări am obţinut un număr de 316 erori de clasificare, ceea ce reprezintă o acurateţe a clasificării pe setul T1 de **86,55%** pentru această metodă. Rezultatele obţinute sunt cu 0,12% mai slabe decât cele obţinute direct pe sumă.

O altă schemă testată este cea în care în pasul în care se realizează ponderarea, am decis că în fiecare vector reprezentat ca în secţiunea anterioară, pentru clasa de pe primul loc, noua valoare să fie vechea valoare **înmulţită** cu 12. Pentru clasa de pe locul 2 va fi vechea valoare înmulţită cu 10...

În acest caz am obţinut un număr de **305** documente clasificate eronat pe setul T1, acurateţea clasificării pentru acest meta-clasificator fiind de **87.02%.** Această acurateţe de clasificare obţinută este cea mai mare obţinută prin utilizarea unui meta-clasificator neadaptiv, dar evident mai mică decât limita maximă de 98.63%, la care poate ajunge teoretic meta-clasificatorul.

Am realizat o serie de cercetări privind şi alte variante de ponderare a elementelor vectorilor, cum ar fi înjumătăţirea ponderilor, ponderi mici liniar descrescătoare iar rezultatele comparative sunt prezentate sintetic în Fig. 3.3. Totuşi, cele mai bune rezultate le-am obţinut în cazul în care valorile ponderilor scad liniar cu un pas egal cu valoarea 0,5. Valoarea de prima poziţie va fi ponderată cu valoarea 8.5 ş.a.m.d., descrescător până la ultima poziţie, unde valoare ponderii este 1.0. Astfel, numărul de documente incorect clasificate de către meta-clasificator a scăzut la 301, rezultând o acurateţe a clasificării de **87.20%.**

Rezultatele prezentate în această secţiune au fost de asemenea publicate în [Cret09].

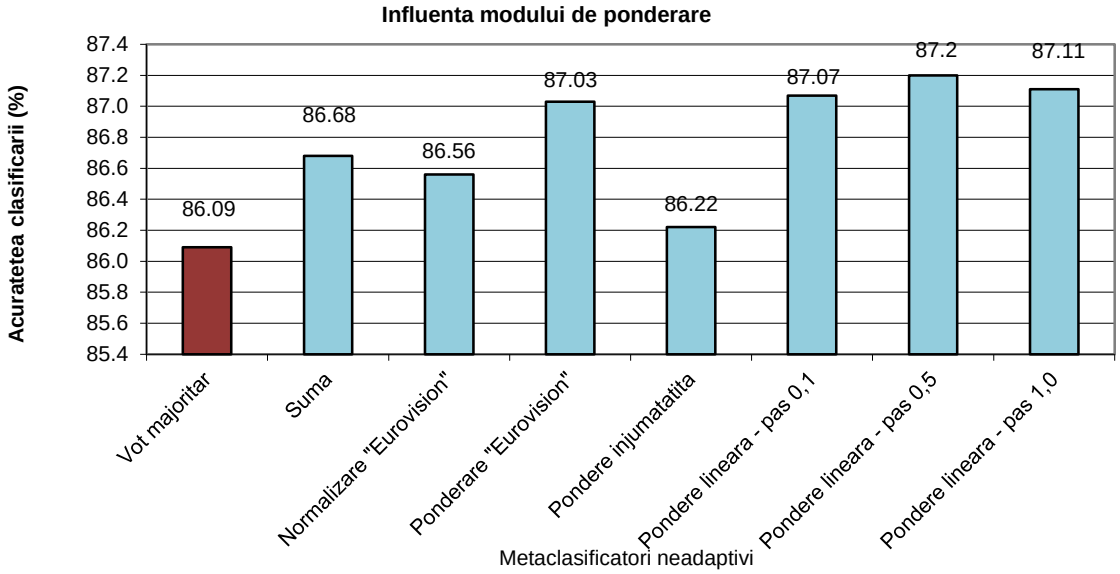

Fig. 3.3 - Comparaţie rezultate metaclasificatori neadaptivi

3.2.2 Meta-clasificator cu ponderi calculate. Design Space Exploration utilizând algoritmi genetici. Rezultate obţinute.

Întrucât alegerea valorilor optime pentru ponderarea fiecărei poziţii de clasă este o sarcină dificilă, în această secţiune prezint o metodă bazată pe un algoritm genetic pentru calculul ponderilor optime. Pornind de la setul de date de test, am creat un fişier de testare adecvat, care este folosit pentru algoritmul genetic. Astfel, pentru fiecare document din setul de testare am salvat toate

ieşirile din toţi clasificatorii separat. Astfel, pentru un document, am obţinut 9 vectori de ieşire (pentru că avem 9 clasificatori), fiecare vector cu un număr de 16 scalari (pentru că avem 16 clase). Elementele fiecăruia dintre cei 9 vectori sunt în prealabil ordonate descrescător. În algoritmul genetic folosit în experimente, un cromozom reprezintă valoarea ponderilor pentru fiecare clasă separat, în funcţie de poziţia clasei. Prima valoare din vector reprezintă ponderea clasei de pe prima poziţie din clasificator, a doua valoare din vector reprezintă ponderea clasei de pe poziţia următoare, şi aşa mai departe. Astfel cromozomul va avea forma:

$$c = \left(w_1, w_2, ..., w_j\right), \text{ cu } j = \overline{1,16} \qquad (3.3)$$

unde w_j reprezintă ponderea pentru fiecare poziţie a clasei, returnată de fiecare clasificator utilizat în meta-clasificator. Setul de antrenament este de forma $\left\{\left\langle \vec{x}_{ij}, y\right\rangle, i = \overline{1,m} \text{ and } j = \overline{1,n}\right\}$, unde y reprezintă clasa corectă pentru documentul $\vec{x}_{ij}$, n reprezintă numărul claselor iar m reprezintă numărul clasificatoarelor utilizate în meta-clasificator. Pentru algoritmul genetic, setul de antrenament este alcătuit din ieşirile tuturor clasificatorilor utilizaţi în meta-clasificator. Pentru a simplifica calcularea valorii funcţiei de fitness, voi folosi o reprezentare a cromozomului care utilizează toate ieşirile clasificatorilor utilizaţi în meta-clasificator. Prin această abordare se va ţine cont şi de poziţia fiecărei clase în ieşirea fiecărui clasificator. În cromozom sunt salvate, pentru fiecare poziţie a clasei, ponderile care se vor utiliza în calcularea acurateţei meta-clasificatorului. Astfel, decizia meta-clasificatorului include şi ponderile pentru fiecare clasă.

Pentru fiecare experiment am pornit cu o generaţie de 100 de cromozomi, fiecare dintre ei având iniţial valori aleatoare cuprinse în intervalul [-1, 1]. Pentru a genera o populaţie nouă, am folosit operatorii genetici ca selecţia, crossover şi mutaţia. Procesul se opreşte după un număr predefinit de 1000 de generaţii, sau dacă nu apar modificări ale celui mai bun cromozom în ultimele 20 de generaţii. Pentru fiecare cromozom, funcţia de fitness se va calcula ca fiind acurateţea clasificării obţinută de meta-clasificator pe setul de test.

Deoarece ordinea claselor este importantă, în cromozom am respectat condiţia $w_1 > w_2 > ... w_{16}$ pentru formula (3.3) şi această condiţie o considerăm cunoştinţe de domeniu. Pentru calcularea funcţiei de fitness, am efectuat următorii paşi:

1. pentru fiecare document din setul de test, algoritmul obţine un vector de ieşire corespunzător pentru fiecare clasificator inclus în meta-clasificator;
2. fiecare vector de ieşire este ponderat cu valorile corespunzătoare din cromozom;
3. clasa învingătoare se stabileşte însumând rezultatele pentru fiecare clasă în parte şi alegând-o pe cea cu valoare maximă conform formulei :

$$Class_j = \sum_{i=1}^{9} w_j c_{ij}, \text{ for } j = \overline{1,16} \quad si \quad WinClass = \arg\max_{j=\overline{1,16}}(Class_j) \qquad (3.4)$$

unde w_j reprezintă valoarea cromozomului pentru ponderea clasei situată pe poziţia j, c_{ij} reprezintă valoarea clasei calculată de clasificatorul i de pe poziţia j, $Class_j$ reprezintă valoarea calculată pentru clasa de pe poziţia j.

4. clasa declarată învingătoare (WinClass) de către meta-clasificator va fi comparată cu clasa reală propusă de Reuters iar dacă clasele pentru acelaşi document sunt identice, atunci am considerat că documentul a fost clasificat corect;
5. după procesarea tuturor documentelor, valoarea funcţiei de fitness se calculează ca fiind acurateţea obţinută pe setul de testare.

Am considerat ca fiind cel mai bun cromozom acela cu care s-a obţinut cea mai mare valoare a funcţiei de fitness. Ca şi metode de selecţie a cromozomilor din cadrul unei populaţii, am folosit două metode: metoda „Roulette Wheel” şi metoda lui Gauss. Pentru crearea unei populaţii am folosit toţi cei 3 operatori genetici astfel: în 30% din cazuri am folosit operatorul de selecţie (selectând întotdeauna cel mai bun cromozom), în 40% din cazuri am folosit operatorul de mutaţie iar operatorul de crossover l-am folosit în 30% din cazuri. Totodată am respectat condiţia iniţială pentru valoarea ponderilor ca $w_1 > w_2 > ... w_{16}$. Pentru operatorul de crossover am căutat punctul de secţionare (s), astfel încât condiţia $w_s^{primul\_părinte} > w_{s+1}^{al\_doilea\_părinte}$ este satisfăcută. Pentru operatorul de

mutaţie, după alegerea aleatoare a punctului de mutaţie (m), noua valoare este aleasă aleator din intervalul $w \in (w_{m-1}, w_{m+1})$. Aceste condiţii au fost impuse în algoritmul genetic, deoarece considerăm că un cromozom care nu le respectă nu reprezintă un cromozom valid şi astfel el nu ar trebui să fie generat.

Testele au fost efectuate pe seturile de date utilizate şi de către meta-clasificatorii propuşi. Pentru fiecare metodă de selecţie a cromozomilor, am efectuat câte 4 teste, având în vedere faptul că rezultatul meta-clasificatorului depinde de valorile cu care se iniţializează ponderile pentru cele 16 clase. Rezultatele obţinute pentru metoda „Roulette Wheel" le prezint în Fig.3.4.

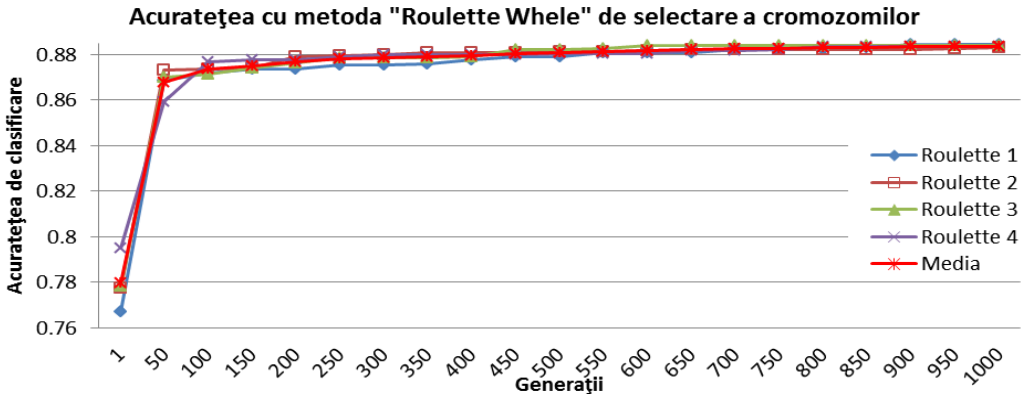

Fig. 3.4 Acurateţea obţinută de metaclasificator utilizând metoda ruletei de selectare a cromozomilor

Rezultatele obţinute de meta-clasificatorul cu ponderi calculate s-au îmbunătăţit în medie faţă de rezultatele meta-clasificatorului neadaptiv prezentat anterior cu 1.16%; în cazul utilizării metodei „Roulette Wheele" de selecţie a cromozomilor, acurateţea de clasificare ajunge la 88.36% iar în cazul utilizării metodei de selecţie Gauss, acurateţea de clasificare creşte cu 1.17% şi ajungând la 88.37%. Cel mai bun rezultat al acestui meta-clasificator a fost obţinut în cazul metodei Gauss de selecţie a cromozomilor, acurateţea de clasificare ajungând la **88.55%** în acest caz. Rezultatele prezentate în această secţiune sunt prezentate detaliat în [Cret11].

3.3 Meta-clasificator adaptiv bazat pe reţea neuronală

Meta-clasificatorul dezvoltat în continuare se bazează pe o preclasificare şi o reţea neuronală de tip feed-forward cu învăţare online. Am dorit să includ în meta-clasificatorul numit în continuare **MC-BP (Meta-clasificator cu reţea Backpropagation)** o reţea neuronală, deoarece am considerat că un meta-clasificator adaptiv poate va reuşi să se „adapteze" şi la datele cu probleme, care există în setul de antrenare/testare. Reţelele neuronale sunt sisteme care se adaptează la schimbările survenite în seturile de date, astfel că meta-clasificatorul MC-BP devine unul mult mai adaptiv decât metodele SBDE şi SBCOS dezvoltate şi prezentate în [Mora06].

Pentru antrenarea şi testarea reţelei neuronale cu învăţare Backpropagation, am plecat de la setul de vectori obţinut de meta-clasificatorul neadaptiv prezentat anterior. Am antrenat acel meta-clasificator cu setul de antrenament A1 (4702 documente) şi l-am testat folosind setul de test T1 (2351 documente) . Ca şi intrare în acest meta-clasificator, avem setul de date iar la ieşire obţinem un set de vectori, câte un vector pentru fiecare document de intrare, de 16 elemente fiecare. Setul de vectori obţinut, pornind de la setul de documente de antrenare A1, pe care îl vom numi în continuare setul AV1 este un vector agregat conţinând 16 elemente (scalari). Acesta va fi folosit în etapa de antrenare a reţelei. Setul de vectori obţinut pornind de la setul de documente de testare T1, numit în continuare TV1, va fi folosit atât in etapa de testare cât şi în etapa de determinare a configuraţiei reţelei.

În ceea ce priveşte arhitectura reţelei backpropagation, am ales una care conţine două straturi de unităţi cu funcţia sigmoidală de activare, iar fiecare unitate de pe fiecare strat este conectată cu toate unităţile de pe stratul precedent. Deoarece la intrare avem la dispoziţie vectori agregaţi de 16 elemente, reţeaua va avea pe stratul de intrare 16 neuroni. La ieşire, meta-

clasificatorul trebuie să „prezică” clasa în care se găseşte documentul curent. Atunci reţeaua Backpropagation va avea la ieşire tot un număr de 16 neuroni, deoarece avem 16 clase distincte, iar la un moment dat va fi activ doar un neuron. În stratul ascuns avem un număr variabil de neuroni, alegerea acestui număr va fi făcut în funcţie de rezultatele simulărilor care vor fi prezentate în secţiunea următoare.

În faza de antrenare, deoarece reţeaua este una cu învăţare supervizată, pentru setul de antrenare am creat un set cu răspunsurile corecte pentru fiecare document în parte. Un astfel de răspuns conţine valoarea „1” pe poziţia clasei corecte şi valoarea „0” în rest. Structura meta-clasificatorului adaptiv M-BP este prezentată în Fig. 3.5.

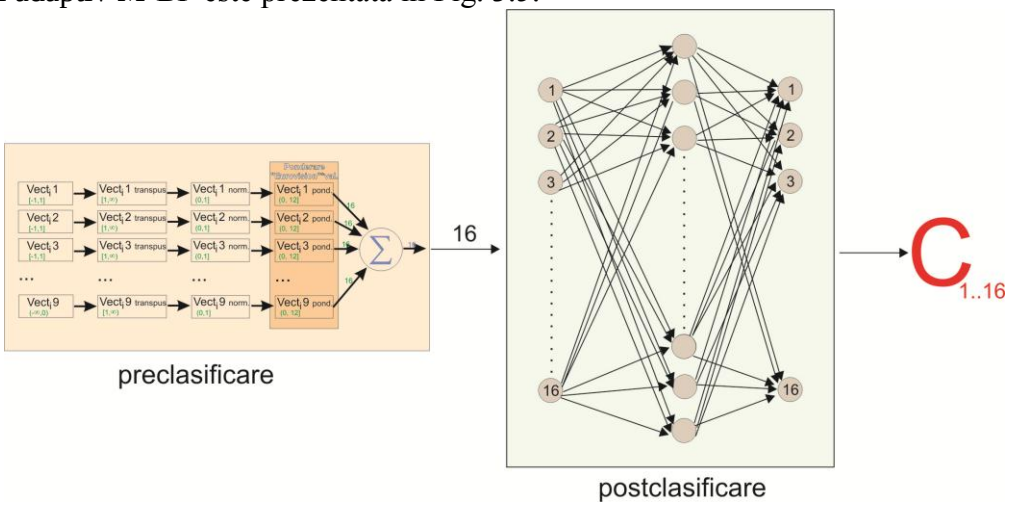

Fig.3.5 Metaclasificator adaptiv M-BP

Deoarece nu există o formulă matematică pentru calcularea numărului optim de neuroni necesari pe stratul ascuns, am realizat câteva experimente în vederea determinării numărului optim de neuroni pentru reţeaua prezentată în Fig. 3.5. Ca şi metodă de evaluare, am oprit antrenarea reţelei după ce aceasta a ajuns din punct de vedere al erorii de antrenare la anumită valoare, am evaluat reţeaua pe întreg setul, din punct de vedere al numărului de documente incorect clasificate, după care am continuat antrenarea. Valorile erorii la care am oprit antrenarea reţelei sunt calculate ca fiind sumă a tuturor erorilor obţinute pentru fiecare exemplu în parte din setul TV1. Evaluarea reţelei în acel punct se face prin contorizarea numărului de documente incorect clasificate de către reţea. Având în vedere că setul TV1 conţine 2351 vectori şi că eroarea pe fiecare vector reprezintă o sumă de 16 elemente, eroarea totală va avea valori supraunitare. Vom începe testarea pornind de la o valoare a erorii totale egală cu 500, ceea ce înseamnă o eroare medie pe fiecare vector de 0.21. Ideea este de ajunge cu eroarea de antrenare la o valoare cât mai mică, într-un timp cât mai scurt.

În cazurile în care reţeaua este mai simplă (are un număr mai mic de neuroni pe stratul ascuns), de la un moment dat eroarea totală a început să scadă foarte încet, moment în care am oprit antrenarea reţelei. De aceea, din acel punct nu vor mai fi valori în graficele pe care le prezint. Am modificat numărul de neuroni utilizând multipli ai lui 16 (Fig. 3.6). Pe măsură ce am crescut numărul neuronilor de pe stratul ascuns păstrând pas de învăţare η=1, eroarea a scăzut de la 0.2 la 0.04 per exemplu. Cea mai bună valoare a acurateţei de clasificare obţinută până în acest moment este de 94.26%, fiind deja superioară celei mai bune valori obţinute cu meta-clasificatorul de tip SBCOS cu 9 clasificatoare (93.32%) [Cret10].

Rezultate obţinute în cazul antrenării pe setul AV1 şi a testării pe TV1 sunt prezentate în Fig. 3.6. Prezint aici rezultate doar pentru arhitecturi ale reţelei cu un număr de neuroni mai mare de 96 pe stratul ascuns şi un coeficient de învăţare descrescător în timp. Şi în acest caz, pentru testare, oprim reţeaua în momentul în care atinge un anumit prag al erorii de antrenare, o testăm pentru a obţine numărul de documente incorect clasificate, după care continuăm cu antrenarea. În acest caz, eroarea totală de antrenare este obţinută ca o sumă a tuturor celor 4702 erori, ceea ce reprezintă o medie a erorii per exemplu de 0.11 în cazul erorii totale egale cu 500. În acest experiment am ajuns la o eroare totală egală cu 80, ceea ce înseamnă o eroare medie de 0.017 per exemplu.

Arhitectura cu 176 de neuroni pe stratul ascuns a obţinut cele mai multe valori minime pentru numărul de documente incorect clasificate, dar în momentul în care eroarea totală de antrenare a scăzut sub valoarea 100, rezultatele cele mai bune au fost obţinute de arhitectura cu 192 de neuroni pe stratul ascuns. În acest caz am obţinut un număr de 14 documente incorect clasificate, ceea ce reprezintă o acurateţe de clasificare a meta-clasificatorului de **99.40%.** Diferenţa faţă de cea mai bună valoare faţă de cea cu 176 de neuroni pe stratul ascuns este de doar **3** documente incorect clasificate.

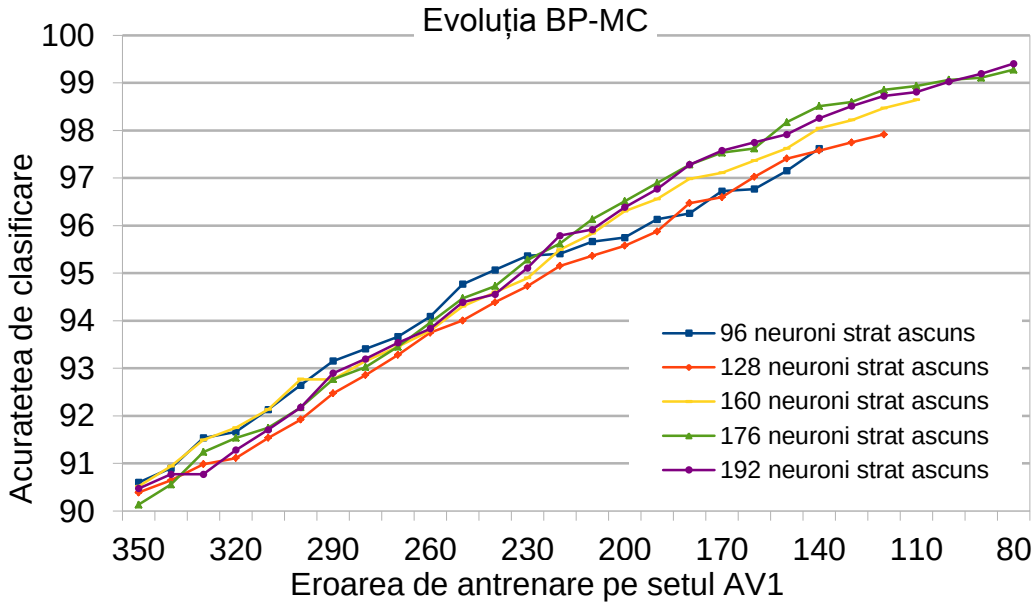

Fig. 3.6 Acurateţea de clasificare în cazul antrenării pe setul AV1 şi a testării pe setul TV1

Acest nou meta-clasificator cu o reţea neuronală cu număr suficient de mare de neuroni pe stratul ascuns a reuşit să depăşească şi limita maximă de 98.63% la care ar fi putut ajunge „teoretic" clasificatorii incluşi în cadrul meta-clasificatorului.

Foarte interesant, acest meta-clasificator neuronal cu învăţare supervizată a demonstrat faptul că acurateţea de 98.63% nu este de fapt limita maximă a meta-clasificării, aşa cum eu considerasem. Datorită procesului de învăţare supervizată, această limită poate fi depăşită. Spre exemplu, în cazul unui vector de intrare în reţea al cărui element maxim nu se află situat pe poziţia clasei corecte, acesta poate activa la ieşire celula corectă tocmai datorită unui proces de învăţare adecvat (în care reţelei i s-au mai livrat exemple asemănătoare). Această limită maximă a acurateţei clasificării este corectă doar pentru metode de agregare neadaptive (postclasificare triviala) ale clasei optimale. Cu acest adaus, limita teoretică rămâne corectă şi devine acum clar de ce algoritmul neuronal (adaptiv) o poate depăşi şi o chiar depăşeşte!

Rezultatul obţinut de acest meta-clasificator pentru acurateţea clasificării de **99.40%** este cel mai bun rezultat obţinut în toate experimentele efectuate în prezenta lucrare (pentru cazul antrenării şi testării pe seturi diferite).

Rezultatele prezentate în aceasta secţiune au fost prezentate şi în [Cret10].

4 Concluzii

Pornind de la meta-clasificatorul cu 9 clasificatori prezentat în sinteza din 2010 am creat o serie de meta-clasificatori neadaptivi care folosesc diferite procedee pentru ponderarea valorilor generate de către fiecare clasificator în parte, cu scopul de a îmbunătăţi acurateţea finală a clasificării. Astfel s-a creat un meta-clasificator care, în loc să contorizeze clasa prezisă de fiecare clasificator în parte, cum ar fi în cazul „vot majoritar" (MV), va însuma simplu toate valorile generate de către clasificatoare pentru fiecare clasă în parte. Am prezentat o serie de experimente care încearcă diferite valori pentru a pondera valorile fiecărei clase din vectorii generaţi de către clasificatori. Aceste valori ponderează vectorii, în funcţie de ordinea obţinută de fiecare clasă în cadrul vectorului. Cel mai bun rezultat obţinut a fost de **301** documente incorect clasificate, ceea ce

reprezintă o acurateţe a clasificării de **87,20%.** Acest rezultat s-a obţinut când s-a utilizat ponderarea liniară cu pasul de „0,5”.

O altă modificare a fost utilizarea de algoritmi genetici pentru găsirea acelor ponderi care, utilizate în meta-clasificatorul neadaptiv prezentat mai sus, să ducă la o îmbunătăţire substanţială a acurateţei de clasificare. Astfel, am creat un meta-clasificator original cu ponderi calculate, folosind algoritmi genetici. Rezultatele obţinute de meta-clasificatorul cu ponderi calculate s-au îmbunătăţit în medie, cu **1.16%** în cazul utilizării metodei „Roulette Wheel” de selecţie a cromozomilor, acurateţea de clasificare ajungând în medie, la **88,36%.** În cazul utilizării metodei de selecţie Gauss pentru selecţia cromozomilor dintr-o populaţie, îmbunătăţirea a fost de **1.17%,** ajungând în medie la **88.37%.** Cel mai bun rezultat al acestui meta-clasificator a fost obţinut într-un experiment folosind metoda Gauss de selecţie a cromozomilor, acurateţea de clasificare ajungând la **88.55%.**

În cazul meta-clasificatorului adaptiv ideea de la care am pornit a fost de a construi acest meta-clasificator format din două componente. O componentă, considerată ca fiind etapa de preclasificare, realizată din meta-clasificatorul (selector) neadaptiv şi o componentă nouă, considerată ca fiind etapa de postclasificare, adaptivă, realizată dintr-o reţea neuronală de tip backpropagation. Cele mai bune rezultate (**99.40%** acurateţe de clasificare) s-au obţinut folosind o reţea neuronală cu **192** de neuroni pe stratul ascuns. Totuşi, din punct de vedere al numărului de rezultate bune obţinute pe parcursul antrenării, optimul a fost atins pentru cazul utilizării unei reţele cu **176** de neuroni pe stratul ascuns (chiar dacă în final doar s-a apropiat de eroarea de antrenare minimă). În urma experimentelor efectuate s-a putut observa că introducerea unei reţele neuronale în cadrul meta-clasificatorului face ca acesta să se adapteze mult mai bine la documentele care trebuie clasificate, reuşind astfel să clasifice şi documentele cu problemă pe care meta-clasificatorii adaptivi şi neadaptivi prezentaţi anterior nu au reuşit să le „înveţe”. Acest nou meta-clasificator a reuşit să depăşească şi limita maximă de **98.63%** la care ar fi putut ajunge „teoretic” (din cauza algoritmului neadaptiv de agregare finală a clasei optime) clasificatorii incluşi în cadrul meta-clasificatorului.

5 Referinţe bibliografice

[Cret09] Creţulescu R., Morariu, D., Vintan, L., *Eurovision-like weighted Non-Adaptive Metaclassifier for Text Documents*, The 8th RoEduNet International Conference, Galaţi, Romania, 2009, **indexată (ISI) Thomson Reuters**

[Cret10] Creţulescu, R., Morariu, D, Vinţan, L, Coman, I. – *An Adaptive Metaclassifier for Text document*, 16th International Conference on Information Systems Analysis, pp. 372-377, ISBN-13: 978-1-934272-86-2(Collection), ISBN-13: 978-1-934272-88-6(Volume II) ,Florida, USA, 2010 **indexată (ISI) Thomson Reuters**

[Cret11] Creţulescu, R., Morariu, D., Breazu, M., Vintan L. – *Using Genetic Algorithms for Weights Space Exploration in an Eurovision-like weighted Metaclassifier*, The second international conference in Romania of Information Science and Information Literacy, ISSN 2067-9882, April 2011.

[Mora06] Morariu, D., Vintan, L., Tresp, V., Feature Selection Method for an Improved SVM Classifier, Proceedings of the 3rd International Conference of Intelligent Systems (ICIS’06), ISSN 1503-5313, vol. 14, pp. 83-89, Prague, August, 2006.

[Mora08] Morariu, D., *Text Mining Methods based on Support Vector Machine*, Ed. MatrixRom, Bucureşti, 2008.

[Reut00] Misha Wolf and Charles Wicksteed - Reuters Corpus: http://www.reuters.com/researchandstandards/corpus/ lansat în noiembrie 2000, accesat în septembrie 2009