Modelarea compactării cu role în industria farmaceutică folosind tehnici de inteligenţă computaţională

Hossam Mohammed Zawbaa Ismail ELSAYED

Facultatea de Matematică şi Informatică

Universitatea Babeş-Bolyai

Rezumatul tezei de doctorat

Doctor în Informatică

Mai 2016

Contents

Abstract

În această teză am dezvoltat variante noi ale unor algoritmi de optimizare inspiraţi din natură, precum leul furnicilor binar (binary antlion optimization BALO), lupul cenuşiu haotic (chaotic grey wolf optimization CGWO), lupul cenuşiu binar (binary grey wolf optimization BGWO), şi mulţi alţii. Toţi algoritmii propuşi de noi au fost comparaţi cu tehnici consacrate folosite în selectarea caracteristicilor, precum optimizarea cu roi de particule (particle swarm optimization PSO) şi algoritmii genetici (genetic algorithms GA). În industria farmaceutică, pentru dezvoltarea de noi medicamente sau pentru optimizarea proceselor de fabricaţie se folosesc frecvent cunoştinţele referitoare la relaţiile de cauzalitate dintre calitatea produsului şi atributele (caracteristicile) formulelor chimice. Folosind cantitatea imensă de date colectate de-a lungul practicii de dezvoltare de produse farmaceutice, modelele de inteligenţă computaţională (computational intelligence CI), bazate pe învăţarea automată şi pe algoritmi de optimizare inspiraţi din natură pot fi folosite eficient pentru a identifica atributele esenţiale de calitate (critical quality attributes CQA) şi parametrii esenţiali de proces (critical process parameters CPP), asociate formulelor chimice, respectiv proceselor de fabricaţie. Obiectivul principal al cercetării a fost evaluarea robusteţii tehnicilor de învăţare automată combinate cu algoritmii de optimizare inspiraţi din natură şi folosite la modelarea proceselor de fabricaţie a tabletelor. Mai precis, efortul a fost concentrat pe predicţia unor proprietă ti ale tabletelor precum porozitatea şi rezistenţa la rupere pe baza caracteristicilor pulberilor şi panglicilor de cauciuc. În acest scop, au fost efectuate experimente de compactare cu role pentru diverşi compuşi farmaceutici, din care au rezultat seturi de date ce conţinô gamă largă de caracteristici. Eficienţa modelării a fost evaluată pe baza caracteristicilor selectate (reducerea) şi folosind rădăcina

pătrată din eroarea medie (root mean square error RMSE). S-a observat că rezultatele prezise sunt într-o bună concordanţă cu datele experimentale.

Cuvinte-cheie:
Algoritmi de optimizare inspiraţi din biologie, Selecţia caracteristică, Compactarea cu role în industria farmaceutică, Optimizarea cu leul-furnicilor, Optimizarea cu fluture de lampă, Optimizarea cu lupul cenuşiu, Optimizarea cu păianjen social, Algoritmul de polenizare a florilor, Algoritmi genetici, Optimizarea cu roi de particule.

Chapter 1

Introducere

O *caracteristică* de intrare este o proprietate măsurabilă a problemei studiate. În ultimii ani, în aplicaţiile de învăţare automată şi recunoaştere de pattern-uri, domeniul caracteristicilor s-a expandat de la sute la mii de variabile. Volumele mari de date generate astăzi în biologie oferă informaţie mai detaliată şi mai utilă, dar complică analiza acestora, deoarece nu toate informaţiile sunt relevante. Selectarea caracteristicilor (atributelor) (feature selection, FS) relevante pentru un set de date este o problemă complexă: *selectarea caracteristicilor* este o tehnică de rezolvare a problemelor de clasificare şi regresie, identificând o submulţime a caracteristicilor şi eliminând pe cele redundante. Acest mecanism este cu atât mai util cu cât numărul caracteristicilor este mai mare, şi nu toate sunt necesare pentru a descrie datele experimentale (1).

Multe lucrări formulează problema selectării caracteristicilor (feature selection problem, FSP)ca fiind o problemă de *optimizare combinatorială*, în care submulţimea rezultată de caracteristici produce cea mai bună potrivire a datelor (2). În aplicaţiile din lumea reală, selectarea caracteristicilor este obligatorie, deoarece avem de-a face cu zgomote, cu caracteristici nerelevante sau greşite (3). Aceşti factori pot avea un impact negativ asupra performanţei clasificării. Metodele de selectare a caracteristicilor se clasifică în raport cu două criterii:

1. *strategia de căutare*: metoda folosită pentru a genera submulţimi sau combinaţii de caracteristici.

2. *calitatea selecţiei (fitness)*: criteriile folosite pentru a caracteriza calitatea unei submulţimi de caracteristici.

În general, FSP se formulează ca o problemă de optimizare *multi-criterială*: *minimizarea* dimensiunii submulţimii selectate de caracteristici şi *maximizarea* performanţei predicÅčiei. De regulă, aceste obiective sunt contradictorii, iar soluţia optimală este un compromis (4).

Mulţi dintre noii algoritmi de optimizare sunt *inspiraţi din natură* (5). Sunt trei mari surse se inspiraţie: *biologia*, *fizica*, şi *chimia*. Toţi noii algoritmi de optimizare bazaţi pe *biologie* se vor numi *bio-inspiraţi* (bio-inspired optimization algorithms, BIOA) (5).

În general, dimensiunea spaţiului de căutare creşte exponenţial cu numărul de caracteristici ale setului de date studiat (6). Prin urmare, o căutare exhaustivă pentru soluţia optimală sau aproape optimală poate fi *impracticabilă*; de asemenea, tehnicile de căutare exhaustivă suferă de *stagnare* în optime locale (2), (7). În toţi algoritmii de optimizare bio-inspiraţi este nevoie de un echilibru convenabil între *explorare* (diversificare, căutare globală) şi *exploatare* (intensificare, căutare locală) (8).

Obiectivele cercetării noastre sunt:

- O_1: utilizarea BIOA la FSP pentru a rezolva problema de clasificare care minimizează numărul de caracteristici selectate şi maximizează precizia clasificării.
- O_2: în problema de regresie, folosirea BIOA pentru a reduce numărul de caracteristici selectate şi a minimiza eroarea predicţiei.
- O_3: aplicarea BIOA în domeniul farmaceutic, pentru minimizarea numărului de caracteristici selectate şi minimizarea erorii predicţiei. De asemenea, studierea importanţei fiecărei variabile de intrare pentru un set de date dat.

În industria farmaceutică, pentru dezvoltarea de produse noi şi optimizarea proceselor de fabricaţie, este necesară o bună întelegere a relaţiilor de cauzalitate dintre calitatea produsului şi atributele formulelor chimice. Producţia industrială de medicamente are patru procese: amestecare, compactare cu role, măcinare şi ştanţare. Compactarea cu role este o metodă de preparare a granulelor de medicament pentru capsule sau tablete cu o densitate adecvată. Cei mai folosiţi adjuvanţi (excipienţi) sunt celuloza microcristalină (MCC), fosfatul de calciu dibasic (DCP) şi lactoza. Compactarea cu role este şi o tehnică de mărire a dimensiunii particulelor, care granulează materialele sub formă de pulbere pentru a obţine subproduse de dimensiuni intermediare, rezultând o granule de calitate înaltă. Este esenţială alegerea optimă a parametrilor critici pentru

compactarea cu role, precum presiunea constantă de compactare, distanţa dintre role, etc.

Teza are 9 capitole, inclusiv cel de faţă. Introducerea rezumă caracteristicile optimizării inspirate din biologie, importanţa selectării caracteristicilor în problemele de clasificare şi regresie, precum şi impactul selectării caracteristicilor asupra domeniului industriei farmaceutice. Capitolul (2) face o trecere în revistă a cercetărilor existente în învăţarea automată, optimizarea inspirată din biologie şi aplicaţiile acestora. Capitolul (3) prezintă informaţie de bază referitoare la algoritmii de învăţare automată şi optimizare inspirată din biologie. Capitolul (4) descrie câteva dimensiuni ale spaţiului experimentelor: diferiţi generatori de numere aleatoare, diferiţi algoritmi propuşi de optimizare inspiraţi din biologie şi metrici de evaluare a performanţelor acestora. Următoarele patru capitole descriu contribuţiile originale: cercetări experimentale de folosire a BIOA în clasificare (Capitolul (5)); rezultatele experimentale ale folosirii BIOA în regresie (Capitolul (6)), rezultatele experimentale ale folosirii BIOA în domeniul farmaceutic (Capitolul (7)). Capitolul (8) descrie alte trei studii de caz folosind BIOA. Studiile descrise conţin caracteristicile seturilor de date folosite şi analiza rezultatelor obţinute, prin compararea cu metode consacrate. În sfârşit, capitolul (9) rezumă concluziile acestui studiu şi direcţiile de cercetare viitoare.

Chapter 2

Starea actuală a cercetărilor în domeniu

2.1 Învăţarea automată

Tehnicile de învăţare automată (machine learning, ML) joacă un rol important în rezolvarea multor probleme complicate de clasificare şi regresie. Tehnicile ML se folosesc pentru construirea de modele de predicţie din date observaţionale. În problemele de clasificare, metodele ML sunt adecvate pentru variabile dependente discrete, având un număr finit de valori neordonate. Pe de altă parte, în cazul problemelor de regresie, variabilele dependente pentru care se folosesc metodele ML au valori continue şi ordonate. Tehnicile bazate pe nucleu (kernel-based techniques), precum maşinile cu suport vectorial (support vector machines, SVM), analiza componentei principale, procesele Gaussiene ş.a. reprezintă un important pas înainte în dezvoltarea ML. SVM au fost propuse de Vapnik în anii 1960 pentru clasificare şi au devenit recent subiectul unor eforturi intense de cercetare datorită dezvoltării tehnicilor şi teoriilor legate de aplicaţii la probleme de regresie şi estimare a densităţii (9). Reţelele neurale, în special cele cu un singur strat ascuns şi feed-forward (single hidden layer feed-forward neural networks, SLFN) sunt considerate unul dintre cele mai utilizate modele de ML în problemele de regresie şi clasificare (10). Modelul învăţării extreme (extreme learning machine, ELM) a fost propus pentru SLFN, unde conexiunile dintre stratul de intrare şi neuronii din stratul ascuns se selectează aleator şi rămân nemodificate pe tot parcursul procesului de învăţare (11).

2.2 Optimizarea inspirată din biologie

Numeroase tehnici euristice reproduc comportamentul sistemelor din natură, biologice sau fizice, fiind considerate metode robuste de optimizare. Algoritmii genetici (genetic algorithms, GA) au fost primii algoritmi evolutivi introduşi în literatură, bazaţi pe procesul natural de evoluţie prin reproducere (12). Un algoritm de FS bazat pe GA şi folosind o mulţime fuzzy ca funcţie de fitness a fost propus în (13). În PSO, fiecare soluţie este o particulă definită de poziţie, fitness şi un vector de viteză ce reprezintă direcţia de mişcare a acesteia (14). Algoritmul de FS folosind optimizarea cu colonia de furnici (ant colony optimization, ACO) a fost aplicat la detectarea intruziunii în reţele (15). ACO foloseşte rata de discriminare a lui Fisher pentru adoptarea informaţiei euristice şi teoria mulţimilor pentru FS (16). Colonia artificială de albine (artificial bee colony, ABC) este un algoritm de optimizare numerică bazat pe comportamentul albinelor la căutarea hranei. În ABC, albinele lucrătoare caută sursa de hrană şi le înştiinţează pe celelalte albine (17). Algoritmul albinei virtuale (virtual bee algorithm, VBA) se aplică la optimizarea funcţiilor numerice în 2-D folosind un roi de albine virtuale care se mişcă aleator în spaţiul caracteristicilor şi interacţionează pentru detectarea surselor de hrană, conducând astfel la soluţiile posibile ale problemei de optimizare (18). ALO este un algoritm evolutiv recent care reproduce mecanismele de vânătoare ale leului furnicilor (19).

Chapter 3

Preliminarii şi fundamente

3.1 Învăţarea automată

Clasificarea şi regresia sunt tehnici de ML pentru construirea de modele de predicţie din date observaţionale. Tehnicile de clasificare se folosesc pentru variabile dependente cu valori discrete şi neordonate, eroarea de predicţie fiind măsurată în funcţie de costul clasificărilor greşite. Tehnicile de regresie se aplică la variabile dependente cu valori continue şi ordonate, eroarea de predicţie (eroarea medie pătratică) calculându-se prin diferenţa pătratelor dintre valorile observate şi cele prezise. Această secţiune prezintă pe scurt tehnicile de clasificare şi regresie folosite în teză.

3.1.1 Tehnici de clasificare

Metodele de ML joacă un rol important la rezolvarea problemelor complicate de clasificare. În cele ce urmează facem o prezentare succintă tehnicilor de clasificare folosite.

3.1.1.1 Vecinul cel mai K-apropiat

Vecinul cel mai K-apropiat (K-nearest neighbor, KNN) este un clasificator foarte simplu. În faza de clasificare, un eşantion nou este clasificat pe baza majorităţii categoriei KNN (K este un întreg predefinit): fiind dat un punct de interogare, algoritmul determină primele K obiecte sau puncte de antrenament cele mai apropiate de el. Se foloseşte distanţa dintre puncte ca măsură de apropiere. După ce sunt găsite clasele KNN, noul eşantion urmează (prezice) clasa majoră a KNN (20).

3.1.1.2 Pădurea aleatoare

Pădurea aleatoare (random forest RF) este considerată una dintre cele mai bune tehnici ML de clasificare şi regresie. Ea poate clasifica cu precizie seturi mari de date (21) şi constă dintr-o colecţie de clasificatori, cu structură arborescentă. Fiecare arbore depinde de valori vectoriale aleatoare eşantionate independent (22). Datele de intrare trec prin rădăcina arborelui şi apoi îl traversează. Datele de intrare sunt eşantionate aleator, fiind înlocuite de mulţimi din ce în ce mai mici. Clasa eşantionului se determină folosind arborii RF bazaţi pe un generator de numere aleatoare (21). Variabila aleatoare specifică cum se efectuează tăierile succesive în timpul construirii arborelui, prin selectarea nodului şi a coordonatei care este punctul de diviziune, ca şi a poziţiei acesteia (23).

3.1.2 Tehnici de regresie

3.1.2.1 Reţelele neurale artificiale

Reţelele neurale artificiale (artificial neural networks ANN) au fost create ca generalizări ale modelelor matematice reprezentând sistemele nervoase biologice. În modelul matematic simplificat al neuronului, efectele sinapselor se reprezintă prin ponderi ale conexiunilor care modulează efectul asupra semnalelor de intrare asociate, iar caracteristicile nonlineare ale neuronilor se reprezintă printr-o funcţie de transfer. Există multe funcţii de transfer pentru prelucrarea intrărilor ponderate şi perturbate, dintre care remarcăm patru dintre ele, folosite în mod frecvent la prelucrarea multimedia. Comportamentul reţelei neurale depinde în mare măsură de interacţiunile dintre neuroni (24).

3.1.2.2 Învăţarea automată extremă

Modelul învăţării automate extreme (extreme learning machine ELM) a fost propus pentru SLFN. În acest model, conexiunile dintre stratul de intrare şi neuronii ascunşi se selectează aleator şi rămân nemodificate pe tot parcursul procesului de învăţare. Conexiunile de ieşire se ajustează prin minimizarea funcţiei de cost, folosind un sistem liniar (11). Caracteristicile proprii ELM sunt alegerea aleatorie a ponderilor neuronilor din stratul ascuns şi funţia de bias. Ambele operaţii ale ELM (antrenarea şi predicţia) sunt mult mai rapide decât cele ale altor tehnici non-liniare. Prin urmare, ELM tinde să ofere o performanţă de generalizare bună cu o viteză de învăţare ridicată (25).

3.1.3 Modele predictive

1. *Cubist* este un pachet ce implementează arborii de decizie predictivi bazaţi pe reguli propuşi de Quinlan (26). Modelele Cubist introduc ecuaţii liniare în nodurile lor terminale, fiind capabili să prezică valori numerice.

2. *Pădurea aleatoare Random Forest (RF)* crează o mulţime de arbori de decizie folosind date de intrare aleatoare. Pachetul `randomForest` din mediul `R` a fost folosit în (27).

3. *Perceptronul multistrat monoton Monmlp (monotonic multilayer perceptron)* (28, 29) a fost folosit pentru a exploata avantajele învăţării fără propagare înapoi.

4. *Reţelele neurale cu învăţare profundă Deep learning neural networks* sunt folosite pentru rezolvarea unor probleme complexe, pe baza combinării unor soluţii mai simple. Aceste sisteme pot fi exploatate în medii reale (30).

5. *FugeR* este destinat antrenării sistemelor fuzzy bazate pe algoritmi evolutivi.

3.2 Algoritmi de optimizare inspiraţi din biologie

3.2.1 Algoritmi genetici

Algoritmii genetici (genetic algorithms, GA) au fost primii algoritmi evolutivi cunoscuţi în literatură, fiind dezvoltaţi de Holland în deceniile 7 şi 8 ale secolului trecut. GA pot rezolva probleme de optimizare complexe şi neliniare.

3.2.2 Optimizarea cu roi de particule

Optimizarea cu roi de particule (particle swarm optimization, PSO) este o metodă euristică de optimizare globală dezvoltată de Kennedy and Eberhart in 1995 (14). Algoritmul PSO face parte din categoria algoritmilor de inteligenţă a roiurilor şi se bazează pe comportamentul de mişcare al păsărilor (31). El este utilizat pe scară largă pentru rezolvarea problemelor de optimizare şi de FS (32).

3.2.3 Colonia artificială de albine

Algoritmul de optimizare bazat pe comportamentul albinelor pentru căutarea hranei se numeşte colonia artificială de albine (artificial bee colony, ABC) şi a fost propus de Karaboga in 2007 (17). În ABC, albinele lucrătoare caută o sursă de hrană şi apoi le înştiinţează pe celelalte. Albinele culegătoare le urmează pe cele lucrătoare iar albinele cercetaş zboară spontan pentru a descoperi surse de hrană mai bune (8).

3.2.4 Algoritmul licuricilor

Algoritmul licuricilor (firefly algorithm, FFA) este o metodă stohastică de optimizare globală dezvoltată de Yang în 2008 (33). FFA imită mecanismele de împerechere şi de schimb de informaţie ale licuricilor care folosesc sclipiri luminoase. În FFA, mişcarea unui licurici este determinată în principal de atractivitatea altor licurici.

3.2.5 Căutarea cucului

Algoritmul căutarea cucului (cuckoo search, CS) este un algoritm euristic de căutare propus de Yang în 2009 (34) şi folosit la rezolvarea problemelor de optimizare continuă. Cucii au o strategie agresivă de reproducere şi-şi depun ouăle în cuiburile altor păsări gazdă, care pot fi de altă specie. Pasărea gazdă poate descoperi că oul nu este al ei, caz în care fie că aruncă oul din cuib, fie abandonează cuibul şi-şi construieşte altul în alt loc.

3.2.6 Algoritmul liliacului

Yang a dezvoltat algoritmul liliacului (bat algorithm, BA) în 2010 (35). BA este o tehnică a meta-heuristică care foloseşte comportamentul de ecolocaţie pentru căutarea prăzii şi detectarea sau evitarea obstacolelor. Pentru navigare, liliecii emit pulsuri sonore puternice şi ascultă ecoul reflectat de obiectele înconjurătoare.

3.2.7 Algoritmul de polenizare a florilor

Algoritmul de polenizare a florilor (flower pollination algorithm, FPA) este o metodă meta-heuristică de optimizare bazat pe operaţia de polenizare a florilor plantelor şi a fost dezvoltat de Yang în 2012 (36). Obiectivul principal al florii este reproducerea prin polenizare. Polenizarea florilor este asociată de obicei cu transferul de polen, realizat de

polenizatori precum insectele, păsările, liliecii ş.a. (37). Polenizarea este de două tipuri: *auto-polenizare (abiotică)* şi *polenizare încrucişată (biotică)*, care corespund căutării locale, respectiv căutării globale în algoritm.

3.2.8 Optimizarea cu păianjen social

Optimizarea cu păianjen social (social spider optimization, SSO) este unul dintre algoritmii recenţi din categoria inteligenţei roiurilor şi a fost propus de Cuevas în 2013 (38). Algoritmul SSO mimează comportamentul coloniei de păianjeni din natură şi are două componente: membrii sociali şi pânza comună. Păianjenul cooperează interactiv cu ceilalţi membri ai coloniei (39).

3.2.9 Optimizarea cu lupul cenuşiu

Algoritmul de optimizare cu lupul cenuşiu (grey wolf optimization, GWO) a fost dezvoltat recent de Mirjalili in 2014. GWO imită modul în care haita de lupi caută hrana şi supravieţuieşte prin evitarea inamicilor (40). Fiecare lup cenuşiu din haită îşi alege poziţia, mişcându-se continuu spre un loc mai bun şi fiind atent la ameninţările potenţiale. GWO are un parametru ce semnifică probabilitatea de ameninţare, imitând incidentele când lupii se ciocnesc de inamicii lor.

3.2.10 Algoritmul hibrid maimuţă - banc de krill

Algoritmul hibrid maimuţă - banc de krill (monkey and krill herd algorithm, MAKHA) a fost dezvoltat de Khalil in 2015 (41). În general, algoritmii hibrizi folosesc operatori dintr-un algoritm combinaţi cu operatori din alt algoritm în ideea de a folosi ce e mai bun din fiecare algoritm pentru a îmbunătăţi performanţa. Algoritmul hibrid MAKHA foloseşte cei mai performanţi operatori din algoritmul maimuţă (monkey algorithm, MA) şi algoritmul bancului de krill (krill herd algorithm, KHA), evitând folosirea operatorilor mai puţin performanţi sau care necesită putere mare de calcul din respectivii algoritmi (41).

3.2.11 Algoritmul libelulă

Mirjalili a propus anul trecut algoritmul libelulă (dragonfly algorithm, DA) (42). Libelulele sunt insecte uimitoare, existând peste 3000 de specii diferite (43). Ciclul de

viaţă al libelulei are două faze principale: nimfă şi adult. Cea mai mare parte a vieţii lor, libelulele sunt în faza nimfă. La terminarea acesteia, ele devin adulte şi se comportă ca nişte prădători mici, vânând aproape toate celelalte insecte mici din mediul lor. DA imită comportamentele de roi static (vânătoare) şi dinamic (migrare) ale libelulelor (44).

3.2.12 Optimizarea cu fluture de lampă

Tot anul trecut, Mirjalili a dezvoltat şi algorithmul de optimizare cu fluture de lampă (moth-flame optimization, MFO). Prin evoluţie, fluturii au ajuns să zboare noaptea orientându-se după lună şi folosind pentru navigare o metodă numită orientare transversală: direcţia de zbor a fluturelui menţine un unghi constant în raport cu luna (adică sursa de lumină) (45). Această metodă este considerată o tehnică foarte eficientă pentru zborul în linie dreaptă pe distanţe lungi (46).

3.2.13 Optimizarea cu leul-furnicilor

Optimizarea cu leul-furnicilor (antlion optimization, ALO) este un algoritm propus tot anul trecut şi tot de Mirjalili (19). ALO imită mecanismele de vânătoare folosite de leul furnicilor în natură. Leii-furnicilor (antlions, doodlebugs) aparţin familiei Myrmeleontidae din ordinul Neuroptera. Ei vânează în stadiul de larvă, iar în stadiul de adult se reproduc (19).

Chapter 4

Sistemul propus şi metodologia de cercetare

O caracteristică comună a algoritmilor de inteligenţa roiurilor este că informaţia este partajată între mai mulţi agenţi. La fiecare iteraţie a procesului de optimizare, toţi agenţii/o parte dintre agenţi îşi actualizează/modifică poziţia pe baza informaţie de poziţie a celorlalţi agenţi sau a lor.

Explorarea sau *căutarea globală* se poate defini ca achiziţie de informaţie nouă prin căutare (33). Explorarea este preocuparea principală a tuturor algoritmilor de optimizare deoarece ea permite determinarea de noi regiuni de căutare care ar putea conţine soluţii mai bune. *Exploatarea* sau *căutarea locală* se poate defini ca folosire a informaţiei cunoscute. Regiunile bune se exploatează prin căutare locală. Procesul de selecţie trebuie să fie echilibrat între selectarea aleatoare şi selectarea greedy, pentru a dirija căutarea către soluţii candidat mai potrivite (exploatare) şi în acelaşi timp prin promovarea diversităţii utile în populaţie (explorare) (33).

Algoritmii BIOA propuşi sunt folosiţi pentru a determina un număr *minim* de caracteristici care *maximizează* performanţa de predicţie. Spaţiul de căutare reprezintă fiecare caracteristică ca dimensiune individuală, amplitudinea fiecărei dimensiuni fiind între 0 şi 1; este nevoie de o metodă inteligentă de căutare pentru determinarea mulţimii optime de caracteristici în spaţiul de căutare imens care maximizează o funcţie de potrivire dată. În cazul clasificării, funcţia generală de potrivire pentru BIOA propuşi este maximizarea performanţei de clasificare (4.1) peste mulţimea de validare, fiind

cunoscută mulţimea de antrenare, cu condiţia menţinerii unui număr minim de caracteristici selectate:

$$\downarrow Fitness = \alpha(1 - P) + \beta\frac{\mid R \mid}{\mid C \mid}, \tag{4.1}$$

unde R este lungimea submulţimii de caracteristici selectate, C este numărul total de caracteristici în setul de date, α şi β sunt doi parametri caracterizând importanţa performanţei clasificatorului şi lungimea submulţimii, $\alpha \in [0, 1]$ and $\beta = 1 - \alpha$, iar P este performanţa clasificării măsurată ca în relaţia (4.2):

$$P = \frac{N_c}{N}, \tag{4.2}$$

unde N_c este numărul instanţelor de date clasificate corect, iar N este numărul total de date din setul de date.

În cazul regresiei, funcţia generală de potrivire pentru BIOA propuşi este minimizarea erorii de predicţie (4.3) peste mulţimea de validare, fiind cunoscută mulţimea de antrenare, cu condiţia menţinerii unui număr minim de caracteristici selectate:

$$\downarrow Fitness = \alpha * E + \beta\frac{\sum_i \theta_i}{N}, \tag{4.3}$$

unde E este eroarea predicţiei, θ reprezintă un vector N-dimensional cu elemente 0 şi 1 reprezentând caracteristici neselectate/selectate, iar N este numărul total de caracteristici din setul de date.

Toate valorile asociate caracteristicilor sunt în intervalul [0, 1], acolo unde valoarea asociată unei caracteristici se apropie de 1; respectiva caracteristică este candidată a fi selectată în predicţie. În calculelele individuale de potrivire, există un prag în raport cu care se decide dacă o caracteristică se selectează în etapa de evaluare.

Termenul aleator de ponderare α este folosit cu o valoare mare astfel ca el să fie adecvat unui spaţiu de caracteristici cu mai multe minime locale. El este folosit pentru a echilibra compromisul între explorare şi exploatare şi trebuie tratat cu atenţie.

În timpul fazei de antrenare, poziţia fiecărui agent (fluture, leul-furnicilor, lup cenuşiu, furnică, albina, etc.) reprezintă o submulţime de caracteristici. Mulţimea de antrenare se foloseşte pentru a evalua modelele de clasificare (KNN) şi de regresie (ELM sau ANN) pe mulţimea de validare în timpul optimizării pentru a ghida procesul

de selectare a caracteristicilor. Fiecare set de date se împarte în trei părţi egale pentru *antrenare*, *validare* şi *testare*. Mulţimea de *antrenare* se foloseşte pentru a antrena modelul de predicţie prin optimizare şi la evaluarea finală. Mulţimea de *validare* se foloseşte pentru evaluarea performanţei modelului de predicţie în timpul procesului de optimizare. Mulţimea de *testare* se foloseşte pentru evaluarea caracteristicilor selectate la evaluarea finală. Modelele de clasificare şi de regresie (KNN sau ELM sau ANN) se folosesc pentru a garanta calitatea caracteristicilor selectate şi sunt evaluate pe mulţimea de validare în interiorul funţiei de validare, pe parcursul procesului de optimizare (31).

4.1 Distribuţii aleatoare

În această secţiune prezentăm succint diverse modele de distribuţii aleatoare (distribuţia Gauss, lèvy flight şi haotică) precum şi modul în care se aplică acestea în diferiţi BIOA. O *variabilă aleatoare* se poate considera ca o expresie a cărei valoare este realizarea sau rezultatul unor evenimente asociate unui proces aleator (33). Variabila aleatoare este o funcţie reprezentată printr-o funcţie care transformă evenimentele în numere reale. Domeniul acestei transformări se numeşte spaţiu de eşantionare. Fiecare variabilă aleatoare se reprezintă printr-o funcţie densitate de probabilitate care exprimă distribuţia sa de probabilitate. În studiul nostru folosim trei distribuţii diferite; detalii despre acestea se găsesc în subsecţiunile următoare şi în (47).

4.1.1 Distribuţia Gauss

Distribuţia Gaussiană (normală) este cea mai populară şi se aplică la multe variabile fizice, precum intensitatea luminii, eroarea/incertitudinea măsurătorilor, ş.a.

4.1.2 Distribuţia Lèvy flight

Mulţi cercetători au studiat comportamentul de zbor al păsărilor şi insectelor care are caracteristicile tipice zborurilor lèvy (48). Distribuţia lèvy flight a fost aplicată la problemele de optimizare şi rezultatele preliminare sunt promiţătoare (49).

4.1.3 Distribuţia haotică

Haosul înseamnă o condiţie sau un loc cu o mare dezordine sau confuzie (50). Sistemele haotice sunt sisteme deterministe care expun comportament neregulat (sau chiar

aleator) şi o dependenţă majoră de condiţiile iniţiale. Haosul este unul dintre cele mai populare fenomene care se petrec în sistemele neliniare, a căror acţiune este complexă şi prezintă caracteristici de imprevizibilitate (51). Teoria haosului studiază comportamentul sistemelor care respectă legi deterministe dar care este aleator şi imprevizibil, adică sistemele dinamice. Variabilele haotice pot trece prin toate stările în anumite intervale, în funcţie de regularitatea lor, fără repetiţie (51). O *transformare haotică* este o transformare care prezintă un anumit tip de comportament haotic (50). În lucrarea de faţă, am folosit trei transformări haotice: logistică, Tent, şi Singer.

4.2 BIOA cu diverse distribuţii aleatoare

4.2.1 Versiunea haotică a BIOA

Datorită unor proprietăţi interesante precum mixingul topologic, orbitele periodice dense, ergodicitatea şi stochasticitatea intrinsecă, sistemele haotice se pot folosi pentru a modela parametrul care caracterizează comutarea între explorare şi exploatare. În FS, căutarea haotică este mai aptă să evadeze din optimele locale decât căutarea aleatoare.

4.2.1.1 Optimizarea haotică cu leul furnicilor

În optimizarea haotică cu leul furnicilor (chaotic antlion optimization, CALO) ALO selectează iterativ o furnică pentru vânare într-o manieră ruletă şi efectuează doi paşi aleatori: (a) pasul aleator al furnicilor în jurul celei selectate şi (b) pasul aleator în jurul leului furnicilor elită/cel mai bun. Furnica selectată îşi adaptează poziţia sa din cei doi paşi aleatori anterori. Parametrul I controlează raportul dintre explorare şi exploatare în ALO, descrescând liniar pentru a permite mai multă exploatare la începutul procesului de optimizare, explorarea devenind mai importantă spre sfârşitul optimizării. Prin urmare, în explorare se folosesc jumătate din resursele de optimizare, iar restul de timp este dedicat exploatării. Sistemele haotice se pot folosi pentru a adapta acest parametru, ce permite comutarea între explorare şi exploatare. Algoritmul CALO propus este prezentat schematic în figura (4.1). Strategia de căutare a abordării bazate pe wrapper explorează spaţiul caracteristicilor pentru a găsi o submulţime de caracteristici ghidat de performanţa individuală a submulţimilor de caracteristici.

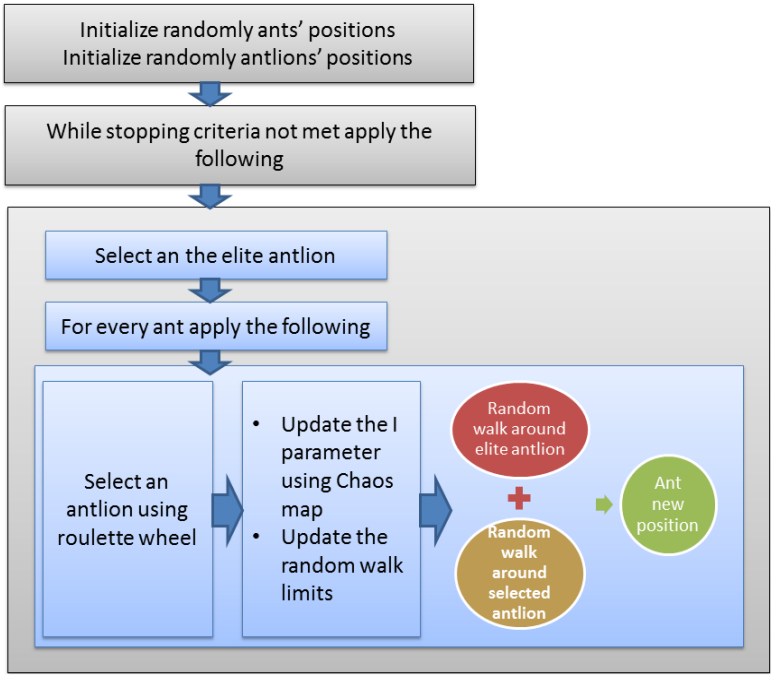


Figure 4.1: Optimizarea haotică cu leul furnicilor propusă (CALO)

4.2.1.2 Optimizarea haotică cu lupul cenuşiu

În GWO (40), un singur parametru, $\overrightarrow{a}$, a fost propus pentru a controla comutarea dintre explorare şi exploatare. Acest parametru a fost prevăzut să descrească liniar pentru a permite mai multă explorare la începutul optimizării, pe când exploatarea devine mai importantă la sfârşitul optimizării. Propunerea permite consumarea a jumătate din timpul de optimizare pentru explorare, în timp ce exploatarea ocupă cealaltă jumătate. În optimizarea haotică cu lupul cenuşiu (Chaotic Grey Wolf Optimization CGWO), parametrul $\overrightarrow{a}$ este astfel adaptat ca să permită perioade succesive de explorare să fie urmate de exploatare, prin distribuirea explorării pe toată perioada de optimizare, urmată, de fiecare dată, de exploatare.

4.2.2 Versiuni Lèvy ale BIOA

Lèvy flight este o modalitate de parcurgere eficientă în explorarea spaţiului de căutare imens, fiind caracterizat de salturi mari şi abrupte. O parcurgere bazată pe lèvy flight

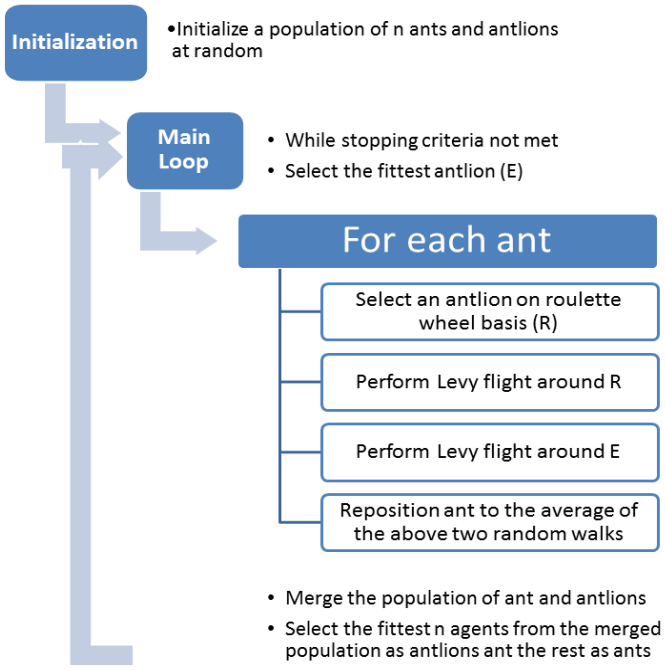


Figure 4.2: Algoritmul de optimizare lèvy cu leul furnicilor (LALO)

este preferată distribuţiei uniforme atunci când se doreşte creşterea vitezei de convergenţă şi convergenţa la optime globale.

4.2.2.1 Optimizarea Lèvy cu leul furnicilor

Optimizarea Lèvy cu leul furnicilor (lèvy antlion optimization LALO) este descrisă în figura (4.2). Randomizarea joacă un rol important atât în *explorare*, cât şi în *exploatare*, esenţa acesteia fiind parcurgerea aleatoare (33) - proces aleator constând din efectuarea unei secvenţe de paşi aleatori consecutivi.

4.2.2.2 Optimizarea Lèvy cu păianjenul social

Optimizarea Lèvy cu păianjenul social (Lèvy social spider optimization LSSO) este o variantă a algoritmului SSO în care lungimea pasului urmează distribuţia lèvy; o astfel de parcurgere aleatoare se numeşte zbor lèvy (lèvy flight) sau parcurgere lèvy (lèvy walk). Din punct de vedere matematic, versiunile simple de zboruri lèvy sunt mai

eficiente decât parcurgerile aleatoare Browniene la explorarea necunoscutului, căutarea în spaţiile mari fiind caracterizată de salturi abrupte şi mari .

4.3 Versiuni binare ale BIOA

În unele probleme specifice, precum FS, soluţiile au doar valori binare {0, 1}, fapt ce sugerează folosirea unor versiuni binare ale BIOA. Aceste versiuni prezintă avantajul reducerii dimensiunii spaţiului de căutare, simplificând sarcina determinării soluţiilor optimale.

4.3.1 Optimizarea binară cu lupul cenuşiu

În optimizarea continuă GWO, lupii îşi modifică continuu poziţiile în spaţiu. Versiunea binară a optimizării cu lupul cenuşiu (BGWO) a fost propusă pentru realizarea FS (52). Formula de actualizare a poziţiei lupilor este funcţie de trei vectori de poziţie, $x_\alpha, x_\beta, x_\delta$, care atrag fiecare lup spre primele trei cele mai bune soluţii. În algoritmul BGWO, mulţimea soluţiilor este în formă binară la orice moment: toate soluţiile sunt vârfuri ale unui hipercub.

4.3.2 Optimizarea binară cu leul furnicilor

Algoritmul ALO are numeroase avantaje precum *explorarea îmbunătăţită*, *evitarea optimelor locale*, *exploatarea*, şi *convergenţa* (19), fapt ce sugerează folosirea lui şi în alte aplicaţii. Algotitmii de optimizare continuă se folosesc pentru determinarea de combinaţii de caracteristici care maximizează performanţa clasificatorului iar agenţii de căutare sunt poziţionaţi într-un spaţiu d-dimensional de căutare pe poziţii în intervalul [0, 1]. În cazul algoritmilor binari de optimizare, spaţiul de căutare este limitat la doar două valori {0, 1} pe dimensiune, deci algoritmul va funţiona mai eficient. De asemenea, operatorul binar este mult mai simplu decât cel continuu.

4.3.3 Optimizarea binară cu fluturele de lampă

În algoritmul MFO, fluturii îşi modifică continuu poziţia în spaţiu pe o spirală (mişcare în spirală, spiral moving SM). SM este componenta computaţională principală a algoritmului, deoarece ea decide cum se repoziţionează fluturii în jurul surselor de lumină,

fapt ce permite fluturelul să zboare în jurul sursei sale de lumină şi nu în spaţiul dintre surse.

4.4 Algoritmi folosiţi pentru comparare

Algoritmii folosiţi pentru comparare ñ acest studiu sunt de dată recentă: ALO, MFO, GWO, SSO, DA, MAKHA, FPA, BAT, CS şi FFA. Toţi algoritmii de optimizare au fost comparaţi cu doi algoritmi consacraţi pentru FS: PSO and GA, ca şi cu variantele fiecăruia dintre algoritmii propuşi.

4.5 Metode de iniţializare

În studiile noastre, am folosit patru metode de iniţializare a parametrilor, capabile să asigure convergenţa din diverse poziţii iniţiale: *iniţializarea mică*, *iniţializarea uniformă*, *iniţializarea mare* şi *iniţializarea MRMR*. *Iniţializarea mică* se foloseşte pentru a testa capabilitatea de căutare globală a unui algoritm de optimizare dat atunci când poziţiile iniţiale ale agenţilor de căutare sunt departe de cele optime. *Iniţializarea mare* evaluează capabilitatea de căutare locală a unui algoritm de optimizare atunci când poziţiile agenţilor de căutare sunt în vecinătatea soluţiei optimale şi este necesară doar căutarea locală pentru a ajunge la soluţia optimală. Modelele de *iniţializare uniformă* se folosesc atunci când poziţiile agenţilor de căutare se setează aleator în spaţiul de căutare folosind distribuţia uniformă pentru fiecare dimensiune a spaţiului. În sfârşit, *iniţializarea MRMR* este o metodă de iniţializare bazată pe filtre care exploatează datele indiferent ce clasificator se foloseşte pentru a selecta o submulţime de caracteristici.

4.6 Metrici de performanţă

Fiecare algoritm a fost rulat de $K * M$ ori cu o poziţionare aleatoare a agenţilor de căutare, cu excepţia soluţiei care includea toate caracteristicile, care a fost forţată să fie pe poziţia unuia dintre agenţii de căutare. Forţarea soluţiei complete garantează că toate submulţimile ulterioare de caracteristici, dacă sunt selectate ca soluţie globală, sunt mai potrivite decât aceasta. Execuţiile repetate ale algoritmilor de optimizare au fost folosite pentru testarea convergenţei. Indicatorii (măsurile) folosite pentru a compara algoritmii, precum media valorilor de potrivire, cea mai bună (cea mai slabă) valoare

de potrivire, abaterea standard a valorilor de potrivire, eroarea prediţiei şi numărul (media) caracteristicilor selectate.

Chapter 5

Aplicarea optimizării inspirate din biologie în procesele farmaceutice

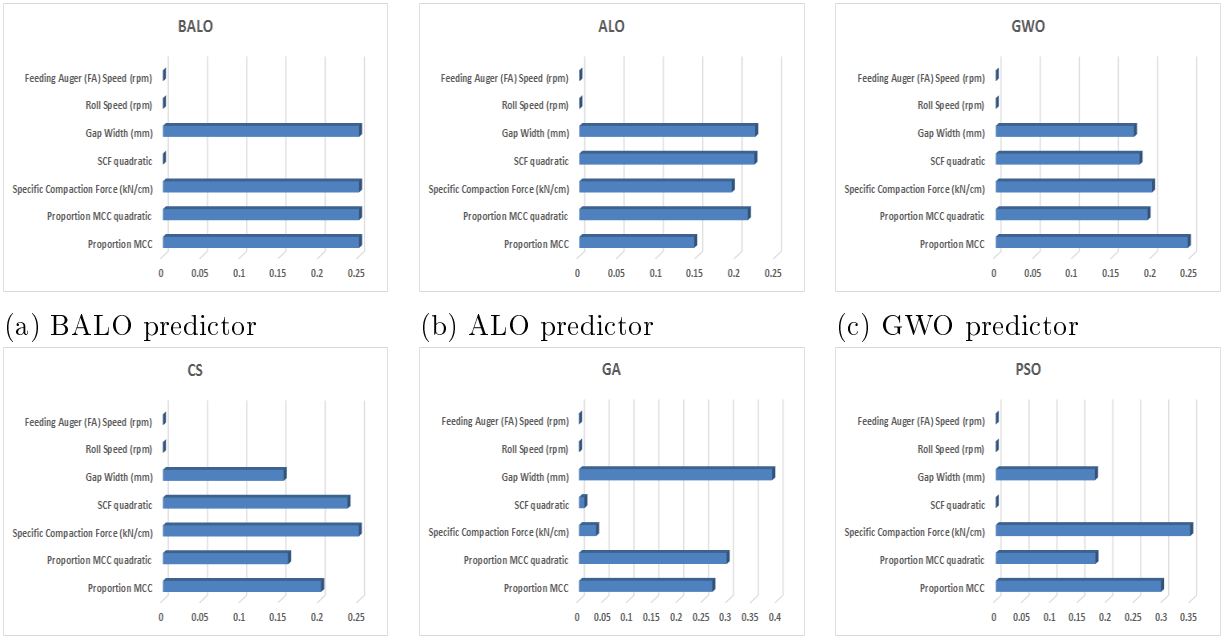
În acest capitol discutăm aplicarea BIOA în diverse procese din industria farmaceutică. ELM şi ANN se folosesc ca funcţii de potrivire în modelele de regresie, ca în formula (4.3). S-au prelucrat trei seturi de date din industria farmaceutică: compactarea cu role, compactarea cu ştanţe şi acidul poli-lactic-coglicolic (poly-lactic-co-glycolic acid PLGA). Rezultatele prezentate în acest capitol sunt preluate din lucrările publicate de autor (53), (54) şi (55).

Seturile de date.

Am folosit trei seturi de date, preluate din cele trei procese farmaceutice menţionate mai sus în cadrul proiectului IPROCOM. De asemenea, pentru PLGA am folosit şi alte seturi de date din literatură, deoarece PLGA are o importanţă deosebită în multe aplicaţii farmaceutice.

5.1 Analiza farmaceutică şi discuţii

În datele farmaceutice, pentru realizarea FS folosind modelarea CI, ANN se foloseşte ca model de regresie pentru evaluarea performanţei prediţiei fiecărui algoritm folosit. Scopul acestei secţiuni este folosirea BIOA la FS pentru domeniul farmaceutic, pentru minimizarea numărului caracteristicilor selectate şi reducerea erorii predicţiei. De asemenea, vom evidenţia caracteristicile selectate şi importanţa fiecăreia dintre ele în modelul de predicţie.



(a) BALO predictor (b) ALO predictor (c) GWO predictor

(d) CS predictor (e) GA predictor (f) PSO predictor

Figure 5.1: Exemplu de importanţă a percentilelor pentru ieşirea granule X10

5.1.1 Rezultate de la compactarea cu role

În acest experiment, s-au folosit toate datele (caracteristicile) de intrare (7) pentru a face predicţia a 4 date (caracteristici) de ieşire (Granule X10, Granule X50, Granule X90 şi granule reziduale). S-au aplicat 6 BIOA pentru FS, combinaţi cu ELM pentru regresie (predicţia celor 4 ieşiri). S-a observat că cele mai importante caracteristici sunt proporţia de MCC, proporţia de MCC pătratică, forţa specifică de compactare (kN/cm) şi distanţa dintre role (mm), după cum se poate observa în figura (5.1). Mai mult, se poate conchide că GWO este cel mai bun dintre algoritmii de optimizare folosiţi în acest experiment, pentru două ieşiri (predicţia de percentile pentru Granule X50 şi granule reziduale), realizând un bun compromis între cele două obiective contradictorii, RMSE şi reducere.

5.1.2 Analiza compactării cu ştanţe

Algoritmii BIOA s-au folosit pentru FS, pentru a obţine date de calitate pentru ANN. Pe urmă, ANN este folosită pentru predicţia celor două ieşiri continue (porozitatea şi

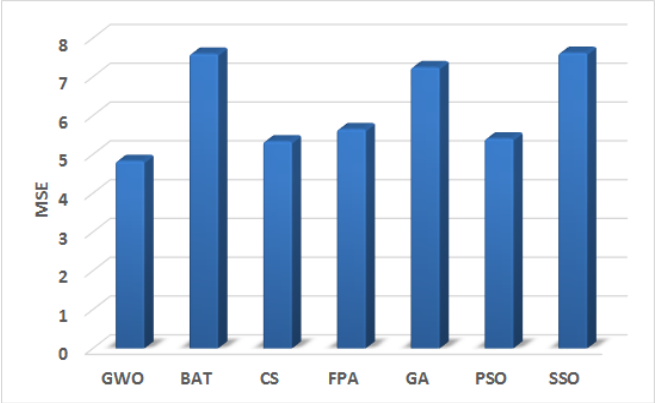


Figure 5.2: Pătratul mediu al erorii (Mean Square Error MSE) pentru porozitate

rezistenţa la traţiune). Rezultatele prezentate în figura (5.2) conţin valorile MSE pentru fiecare algoritm (20 de rulări diferite), iar în figura (5.3) este reprezentată reducerea medie de caracteristici pentru cele două ieşiri amintite anterior. S-a constatat că algoritmul GWO este cel mai precis pentru predicţia porozităţii, iar SSO pentru predicţia rezistenţei la tracţiune. Cea mai mare reducere a numărului de caracteristici a fost de 60%, cu un pătrat mediu al erorii (MSE) de 7.2 pentru predicţia porozităţii, respectiv 5.1 pentru predicţia rezistenţei la tracţiune. Cumulativ, algoritmul GWO a obţinut cel mai bun compromis între MSE şi reducere pentru ambele ieşiri. Cele mai importante intrări au fost "presiunea de compactare", urmată de "material" şi de "limita superioară a dimensiunii granulei" (53).

5.1.3 Acidul poli-lactic-co-glicolic PLGA

Pentru consistenţa comparării rezultatelor ambelor abordări, datele de intrare au fost preluate din literatură, iar structura lor a fost menţinută la fel ca în Szlęk et al. (54). Pe scurt datele s-au colectat din aproximativ 200 de lucrări ştiinţifice. Datele extrase constă din ratele de livrare a 68 formule de PLGA din 24 publicaţii. Vectorul de intrare original avea 320 variabile (descriptori moleculari ai proteinelor, excipienţi, caracteristici ale formulelor şi condiţii experimentale) şi 745 observaţii (înregistrări). Selectarea variabilelor esenţiale a fost realizată în ipoteza că predictibilitatea şi simplitatea modelului sunt la fel de importante pentru rezultatul final. La început, s-au folosit 4 BIOA – ALO, BALO, GWO and SSO – ca instrument pentru FS. Pentru fiecare mulţime

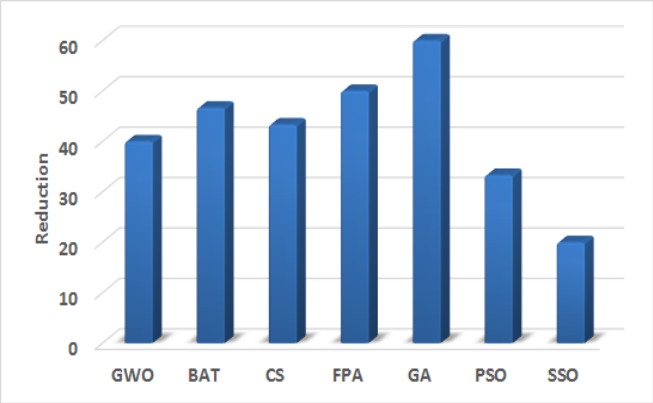


Figure 5.3: Reducerea medie pentru porozitate şi rezistenţa la tracţiune

de caracteristici propusă de instrumentul de FS, s-a folosit o procedură de screening pentru determinarea erorii de generalizare minime peste diversele modele predictive şi setările/arhitecturile acestora. Pentru modelarea predictivă s-au folosit Cubist, RF, ANN (MLP monoton, MLP deep learning) şi FugeR.

5.1.3.1 Rezultate şi discuţii pentru PLGA

Măsurile descrise în secţiunile următoare se folosesc pentru cuantificarea calităţii rezultatelor obţinute de modelele computaţionale. Pentru măsurarea preciziei clasificării datelor, s-a folosit RMSE (root mean squared error, radical din eroarea medie pătratică) atât la FS cât şi la modelele predictive. Dimensiunea reducerii, indicator de dimensiune, este folosit la metodele de FS. În total, considerând toate metodele şi mulţimile 10-cv, aproape 18,000 de modele au fost antrenate şi testate. RMSE normalizată (NRMSE, normalized RMSE) a variat de la 31.1 la 15.9%. Algoritmul RF (Random Forest RF) a produs cea mai mică eroare; în consecinţă, a fost folosit pentru selectarea vectorilor de intrare optimali, vezi tabelul (5.1). Modelul RF dezvoltat pe 9 seturi de date de intrare , 9in(2), selectate cu algoritmul BALO, a produs una dintre cele mai mici NRMSE, vezi tabela (5.2). S-au obţinut rezultate comparabile cu cele ale lui Szlęk et al. (54), adică 15.97% faţă de 15.4%, dar vectorul intrărilor a fost mai mic, adică 9 (în loc de 11) (55).

Table 5.1: NRMSE pentru vectori de intrare selecta cti de BIOA

Metoda FS	Nr. intrări	Cubist	Mon-mlp	RF	FugeR
ALO	8	22.45	24.55	21.95	-
	12	25.95	25.15	22.19	-
	20	18.73	20.20	16.33	20.15
BALO	9(1)	21.20	20.63	18.81	-
	9(2)	18.26	17.31	15.97	18.09
	9(3)	22.60	21.88	19.79	-
	11	19.40	19.35	18.70	-
	12	17.26	18.17	16.56	18.73
GWO	15	19.30	18.88	16.73	19.10
	18	20.65	18.58	17.63	-
	24	20.30	22.30	17.90	-
	25	20.04	19.29	15.86	19.10
	26	17.32	22.22	16.22	-
SSO	8	30.49	31.12	28.89	-
	13	27.09	25.82	24.86	-

Table 5.2: Resultate pentru 9in (2), antrenat şi testat pe 10cv seturi de date.

Algoritmul	NRMSE	R2
Cubist	18.26	0.611
Monmlp	17.31	0.652
ANN cu deep learning	16.87	0.655
FugeR	18.09	0.612
RF	15.97	0.692

Chapter 6

Concluzii şi cercetări viitoare

6.1 Concluzii

În lucrarea de faţă, au fost propuşi şi aplicaţi BIOA pentru FS în modul wrapper (învelitoare). Au fost studiaţi cei mai recenţi BIOA, iar rezultatele obţinute au fost comparate cu metode consacrate de FS, precum PSO şi GA. Evaluarea s-a efectuat folosind o mulţime de criterii de evaluare, în scopul evidenţierii unor aspecte multiple. Selectarea caracteristicilor (feature selection FS) este formulată ca problemă de optimizare multicriterială, cu funcţia de potrivire reflectând performanţa predicţiei şi gradul de reducere a numărului de caracteristici. S-a folosit o mulţime de indicatori de evaluare pentru caracterizarea performanţelor algoritmilor de optimizare, aplicaţi pentru peste 21 de seturi de date în problemele de clasificare şi 10 seturi de date în problemele de regresie, date preluate din repozitoriul UCI.

De asemenea, au fost evaluaţi algoritmii de FS inspiraţi din biologie Îa modelarea proceselor de fabricaţie a tabletelor din industria farmaceutică, în particular pentru predicţia unor proprietăţi precum porozitatea şi rezistenţa la traţiune a pulberilor. Eficienţa modelării a fost evaluată în raport cu reducerea medie a numărului de caracteristici şi RMSE. Cel mai precis în predicţia porozităţii s-a dovedit a fi algoritmul GWO, iar pentru prediţia rezistenţei la tracţiune campionul a fost SSO. În final, putem conchide că BIOA sunt algoritmi eficienţi de căutare, fiind adecvaţi pentru rezolvarea problemelor de FS.

6.2 Cercetări viitoare

Pe baza cercetărilor de până acum, rezultă cel puţin următoarele şase noi direcţii de cercetare:

1. Evaluarea algoritmilor BIOA propuşi pe seturi de date de complexitate mare.
2. Prelucrarea avansată (din punct de vedere statistic) a rezultatelor experimentale.
3. Folosirea BIOA la rezolvarea unor probleme interesante şi din alte domenii.
4. Folosirea şi a altor tehnici de ML, precum SVM, SVR şi RF, pentru evaluarea potrivirii bazată pe învelitoare (wrapper-based).
5. Propunerea unei funcţii de potrivire multi-obiectiv care foloseşte BIOA pentru determinarea unui subset optimal de caracteristici.
6. Propunerea unor combinaţii ale tehnicilor de BIOA recente la rezolvarea problemelor de FS.

References

[1] B. Chizi, L. Rokach, and O. Maimon. **A Survey of Feature Selection Techniques**. *IGI Global*, pages 1888–1895, 2009. 8

[2] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*, **2nd Edition**. Wiley-Interscience, 2000. 8, 9

[3] Y. Chen, D. Miao, and R. Wang. **A rough set approach to feature selection based on ant colony optimization**. *Pattern Recognition Letters*, **31**(3):226–233, 2010. 8

[4] S. Shoghian and M. Kouzehgar. **A Comparison among Wolf Pack Search and Four other Optimization Algorithms**. *Computer, Electrical, Automation, Control and Information Engineering*, **6**(12):1619–1624, 2012. 9

[5] I.F. Jr., X.S. Yang, I. Fister, J. Brest, and D. Fister. **A Brief Review of Nature-Inspired Algorithms for Optimization**. *Elektrotehniski Vestnik*, **80**(3):116–122, 2013. 9

[6] I. Guyon and A. Elisseeff. **An introduction to variable and attribute selection**. *Machine Learning Research*, **3**:1157–1182, 2003. 9

[7] B. Xue, M. Zhang, and W.N. Browne. **Particle swarm optimization for feature selection in classification: a multi-objective approach**. *IEEE transactions on cybernetics*, **43**(6):1656–1671, 2013. 9

[8] R. Akbari, A. Mohammadi, and K. Ziarati. **A novel bee swarm optimization algorithm for numerical function optimization**. *Communications in Nonlinear Science and Numerical Simulation*, **15**:3142–3155, 2010. 9, 16

[9] V. Vapnik. **Statistical Learning Theory**. *IEEE Transactions on Neural Networks*, **10**(5):988–999, 1999. 11

[10] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse. **OP-ELM: Optimally Pruned Extreme Learning Machine**. *IEEE Transactions Neural Networks*, **21**(1):158–162, 2010. 11

[11] C. Jiuwen and L. Zhiping. **Extreme Learning Machines on High Dimensional and Large Data Applications: A Survey**. *Mathematical Problems in Engineering*, **2015**(1):1–13, 2015. 11, 14

[12] J.H. Holland. *Adaptation in natural and artificial systems.* MIT Press, Cambridge, MA, USA, 1992. 12

[13] B. Chakraborty. **Genetic algorithm with fuzzy fitness function for feature selection**. In *International Symposium on Industrial Electronics*, pages 315–319. IEEE, 2002. 12

[14] R. Eberhart and J. Kennedy. **A New Optimizer Using Particle Swarm Theory**. In *International Symposium on Micro Machine and Human Science*, pages 39–43. IEEE, 1995. 12, 15

[15] H.H. Gao, H.H. Yang, and X.Y. Wang. **Ant colony optimization based network intrusion feature selection and detection**. In *International Conference on Machine Learning and Cybernetics*, pages 3871–3875. IEEE, 2005. 12

[16] H. Ming. **A rough set based hybrid method to feature selection**. In *International Symposium on Knowledge Acquisition and Modeling*, pages 585–588. IEEE, 2008. 12

[17] D. Karaboga and B. Basturk. **A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm**. *Journal of Global Optimization*, **39**:459–471, 2007. 12, 16

[18] X.S. Yang. **Engineering optimizations via nature-inspired virtual bee algorithms**. *Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach*, **3562**:317–323, 2005. 12

[19] S. Mirjalili. **The Ant Lion Optimizer**. *Advances in Engineering Software*, **83**:80–98, 2015. 12, 18, 25

[20] A. Kibriya and E. Frank. **An empirical comparison of exact nearest neighbour algorithms**. *European Conference on Principles and Practice of Knowledge Discovery in Databases*, **4702**:140–151, 2007. 13

[21] V.Y. Kulkarni and P.K. Sinha. **Efficient Learning of Random Forest Classifier using Disjoint Partitioning Approach**. *World Congress on Engineering*, **2**, 2013. 14

[22] L. Breiman. **Random forests**. *Machine learning*, **45**(1):5–32, 2001. 14

[23] G. Biau, L. Devroye, and G. Lugosi. **Consistency of Random Forests and Other Averaging Classifiers**. *Machine Learning Research*, **9**:2015–2033, 2008. 14

[24] D.S. Huang. **A constructive approach for finding arbitrary roots of polynomials by neural networks**. *IEEE Transaction on Neural Networks*, **15**(2):477–491, 2004. 14

[25] G.B. Huang, Q.Y. Zhu, and C.K. Siew. **Extreme Learning Machine: theory and applications**. *Neurocomputing*, **70**(1):489–501, 2006. 14

[26] J.R. Quinlan. **Learning with continuous classes**. *Proceedings of the 5th Australian joint Conference on Artificial Intelligence*, **92**:343–348, 1992. 15

[27] A. Liaw and M. Wiener. **Classification and Regression by randomForest**. *R News*, **2**(3):18–22, 2002. 15

[28] A.J. Cannon. **monmlp: Monotone Multi-Layer Perceptron Neural Network** [online]. 2015 [cited 2016 Jan 15]. 15

[29] H. Zhang and Z. Zhang. **Feedforward networks with monotone constraints**. *International Joint Conference on Neural Networks*, **3**:1820–1823, 1999. 15

[30] Y. Bengio. **Learning deep architectures for AI**. *Foundations and Trends in Machine Learning*, **2**(1):1–127, 2009. 15

[31] B. Xue, M. Zhang, and W.N. Browne. **Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms**. *Applied Soft Computing*, **18**:261–276, 2014. 15, 21

[32] B. Chakraborty. **Feature subset selection by particle swarm optimization with fuzzy fitness function**. In *Third International Conference on Intelligent System and Knowledge Engineering*, pages 1038–1042. IEEE, 2008. 15

[33] X.S. Yang. **Nature-Inspired Metaheuristic Algorithms**. *Luniver Press, UK*, **2nd Edition**, 2010. 16, 19, 21, 24

[34] X.S. Yang and S. Deb. **Cuckoo Search via Levy Flights**. *World Congress on Nature and Biologically Inspired Computing*, 2009. 16

[35] X.S. Yang. **A New Metaheuristic Bat-Inspired Algorithm**. *Nature Inspired Cooperative Strategies for Optimization*, **284**:65–74, 2010. 16

[36] X.S. Yang. **Flower pollination algorithm for global optimization**. *Unconventional Computation and Natural Computation, Lecture Notes in Computer Science*, **7445**:240–249, 2012. 16

[37] X.S. Yang, M. karamanoglu, and X. He. **Multi-objective Flower Algorithm for Optimization**. *International Conference on Computational Science, Procedia Computer Science*, **18**:861–868, 2013. 17

[38] E. Cuevas, M. Cienfuegos, D. Zaldivar, and M. Perez-Cisneros. **A swarm optimization algorithm inspired in the behavior of the social-spider**. *Expert Systems with Applications*, **40**(16):6374–6384, 2013. 17

[39] T. Jones and S. Riechert. **Patterns of reproductive success associated with social structure and microclimate in a spider system**. *Animal Behaviour*, **76**(6):2011–2019, 2008. 17

[40] S. Mirjalili, S.M. Mirjalili, and A. Lewis. **Grey Wolf Optimizer**. *Advances in Engineering Software*, **69**:46–61, 2014. 17, 23

[41] A.M.E. Khalil, S.K. Fateen, and A. Bonilla-Petriciolet. **MAKHAŮA New Hybrid Swarm Intelligence Global Optimization Algorithm**. *Algorithms*, **8**(2):336–365, 2015. 17

[42] S. Mirjalili. **Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems**. *Neural Computing and Applications*, **1**(1):1–21, 2015. 17

[43] J.H. Thorp and D.C. Rogers. *Thorp and Covich's freshwater invertebrates*, **4th Edition**. Elsevier, 2014. 17

[44] M. Wikelski, D. Moskowitz, J.S. Adelman, J. Cochran, D.S. Wilcove, and M.L. May. **Simple rules guide dragonfly migration**. *Biology letters*, **2**(1):325–329, 2006. 18

[45] S. Mirjalili. **Moth-Flame Optimization Algorithm: A Novel Nature-inspired Heuristic Paradigm**. *Knowledge-Based Systems*, **89**:228–249, 2015. 18

[46] K.J. Gaston, J. Bennie, T.W. Davies, and J. Hopkins. **The ecological impacts of nighttime light pollution: a mechanistic appraisal**. *Biological reviews*, **88**:912–927, 2013. 18

[47] A.E. Hassanien and E. Emary. **Swarm Intelligence: Principles, Advances, and Applications**. *CRC Press, Taylor & Francis Group*, 2015. 21

[48] I. Pavlyukevich. **Levy flights, non-local search and simulated annealing**. *Computational Physics*, **226**(2):1830–1844, 2007. 21

[49] A.M. Reynolds and M.A. Frye. **Free-flight odor tracking in Drosophila is consistent with an optimal intermittent scale-free search**. *PLoS One*, **2**(4), 2007. 21

[50] R. Vohra and B. Patel. **An Efficient Chaos-Based Optimization Algorithm Approach for Cryptography**. *Communication Network Security*, **1**(4):75–79, 2012. 21, 22

[51] B. Ren and W. Zhong. **Multi-objective optimization using chaos based PSO**. *Information Technology*, **10**(10):1908–1916, 2011. 22

[52] E. Emary, H.M. Zawbaa, and A.E. Hassanien. **Binary Grey Wolf Optimization Approaches for Feature Selection**. *Neurocomputing, Journal indexed*

in 'Journal Citation Reports' (Thomson Reuters), Impact Factor (2014): 2.083, **172**:371–381, 2016. 25

[53] H.M. Zawbaa, S. Schiano, L. Perez-Gandarillas, C. Grosan, C.Y. Wu, and A. Michrafy. **An Evaluation of Bio-inspired Feature Selection Techniques for Computational Intelligence Modeling of Die Compaction**. *International congress on Particle Technology (PARTEC)*, 2016. 28, 30

[54] J. Szlęk, A. Paclawski, R. Lau, R. Jachowicz, and A. Mendyk. **Heuristic modeling of macromolecule release from PLGA microspheres**. *Nanomedicine*, **8**:4601–4611, 2013. 28, 30, 31

[55] H.M. Zawbaa, J. Szlek, C. Grosan, A. Mendyk, and R. Jachowicz. **Computational modelling and optimization of the macromolecule release from PLGA microspheres**. *PLOS ONE*, 2016. 28, 31