|  |  |  |
| --- | --- | --- |
|  | **Departamentul Automatică şi Informatică Industrială**  **Facultatea Automatică şi Calculatoare**  **Universitatea POLITEHNICA din Bucureşti**  Splaiul Independenţei 313, 060042, Bucureşti, România  Sala ED 412, Tel. 021/402.92.69  [www.aii.pub.ro](http://www.aii.pub.ro), email: secretariat@aii.upb.ro |  |

|  |  |
| --- | --- |
|  |  |

**Sesiunea de Comunicări Ştiinţifice Studenţeşti**

**Ediţia 2021**

**Sistem suport bazat pe rețele neuronale pentru diagnosticarea unor afecțiuni medicale**

**Autor: Dinu Silviu-Alexandru, ACS, 1, PCSAM**

**Adresa e-mail: silviu\_dinu@yahoo.com**

**Îndrumător ştiinţific: Conferențiar dr. ing., Loretta Ichim**

Cuprins

[1. Introducere 3](#_Toc71048378)

[2. Stadiul actual 4](#_Toc71048379)

[3. Metodologia abordată 7](#_Toc71048380)

[3.1. Baza de date utilizată 7](#_Toc71048381)

[3.2. Metodologia sistemului propus 7](#_Toc71048382)

[4. Implementarea soluției 11](#_Toc71048383)

[4.1. Antrenarea și validarea rețelei 11](#_Toc71048384)

[4.2. Serverul aplicației WEB 13](#_Toc71048385)

[4.3. Interfața grafică a aplicației WEB (Frontend) 14](#_Toc71048386)

[5. Rezultate experimentale și discuții 16](#_Toc71048387)

[6. Concluzii și dezvoltări ulterioare 19](#_Toc71048389)

[7. Bibliografie 20](#_Toc71048390)

# Introducere

Din cauza stilului de viață din zilele noastre și a contextului pandemic care a determinat populația să petreacă din ce în ce mai mult timp în izolare, omenirea a devenit mai predispusă la dezvoltarea de afecțiuni cauzate de alimentație incorectă și lipsa activității fizice. Mai mult decât atât, consultațiile efectuate de medici au fost realizate într-o măsură considerabilă în regim online. Din această cauză, sistemul medical a întâmpinat dificultăți în diagnosticarea precisă a pacienților. De aceea, acum mai mult decât niciodată s-a conștientizat nevoia utilizării unor soluții bazate pe deep learning pentru a contribui la diagnosticarea corectă a pacienților bazându-se pe simptomele pacientului și antecedentele medicale ale acestuia.

Deep learning este un concept apărut în urma dorinței cercetătorilor de a copia mecanismul de funcționare al creierului uman. Fiind un subdomeniu al inteligenței artificiale, deep learning se axează pe algoritmi capabili să ajute un calculator să „gândească” precum creierul uman. Așa cum este menționat și în [1], avantajul creierului uman comparativ cu calculatorul este abilitatea acestuia de a crea conexiuni rapide. De exemplu, creierul nostru poate recunoaște o altă față umană chiar și într-un mediu slab luminat, pentru ca a mai văzut-o în trecut.

Plecând de la premisa de mai sus, abilitatea unui calculator de a realiza conexiuni și asocieri precum creierul uman poate avea avantaje numeroase în domenii precum medicina sau economia.

În ceea ce privește domeniul medical, aplicațiile inteligenței artificiale sunt vaste. Odată cu dezvoltarea tehnologiei, majoritatea metodelor medicale de tratament sau diagnosticare au la bază programe software rezultate în urma cercetării intensive de-a lungul mai multor ani. După cum este precizat în [2], volumul de date pe care medicii sunt nevoiți să le prelucreze pentru a diagnostica corect un pacient crește pe zi ce trece, lucru pe care factorul uman nu îl poate efectua cu o acuratețe de 100%. În acest caz, metodele de deep learning și inteligență artificială pot facilita procesul de prelucrare a datelor, și pe baza istoricului datelor medicale disponibile, pot prezice un diagnostic ce poate fi validat de către medic.

Astfel, se poate observa necesitatea unei soluții bazată pe inteligență artificială în domeniile unde procesarea cu o acuratețe mare a unor cantități mari de date poate face diferența între succes sau eșec. Din punct de vedere medical, pentru a se putea implementa soluția de prezicere a unui diagnostic menționată anterior, se pot utiliza rețelele neuronale. Rețelele neuronale, așa cum sugerează numele, sunt construite pentru a copia creierul uman. În cadrul acestui lucrări, s-au utilizat rețelele neuronale pentru a obține un diagnostic plecând de la o serie de simptome oferit ca intrare.

Această lucrare își propune să realizeze un sistem bazat pe utilizarea conceptelor de deep learning și rețele neuronale în domeniul medical pentru a analiza și procesa simptomele unor pacienți în vederea generării un diagnostic. De asemenea se vor evidenția rezultatele experimentale obținute în urma punerii în practică a conceptelor amintite în capitolele ce urmează. Lucrarea se va concluziona cu o aplicație pe care utilizatorii o pot folosi pentru a afla un diagnostic preliminar pentru simptomele prezente.

# Stadiul actual

În ciuda evoluției rapide a tehnologiei în domeniul medical din zilele noastre, încă nu se poate afirma că tehnicile bazate pe inteligență artificială vor înlocui prea curând factorul uman. Desigur că, deși încă nu se poate ridica problema înlocuirii, medicii utilizează astfel de soluții pentru a ajunge la un diagnostic sau tratament optim pe baza observațiilor în urma unui consult, ținând cont totodată și de istoricul pacientului în cauză.

Având în vedere stadiul actual al algoritmilor de predicție, se poate afirmă că existența și securitatea datelor medicale individuale sunt la fel de importante precum modelul în sine. Integritatea datelor este vitală pentru felul în care un model antrenat se va comporta pe viitor, de aceea este important ca sursele acestor date să fie credibile și verificate.

După cum este menționat în [3], una dintre principalele modalități de a utiliza inteligența artificială în domeniul medical o reprezintă abilitatea unui calculator de a scana și lucra cu imagini. În prezent, sunt efectuate cercetări pentru a antrena diverse modele bazate pe deep learning să deosebească tumorile maligne de cele benigne, sau chiar să detecteze tumori pe creier din timp. Astfel, un diagnostic corect în fazele incipiente ale unei afecțiuni tumorale poate face diferența între viață și moarte pentru pacienți.

Lucrul cu imaginile în contextul rețelelor neuronale sau algoritmi bazați pe inteligență artificială presupune o etapa importantă de pre-procesare a datelor. Cauza principala a acestui aspect o reprezintă faptul ca acești algoritmi necesită intrări și ieșiri digitale (numerice), drept urmare imaginile trebuie descompuse în reprezentații matriceale numerice și redimensionate astfel încât să poată fi înțelese de algoritm.

În zilele noastre, evoluțiile rapide ale sistemelor avansate de calcul și imagistică din domeniile ingineriei biomedicale au dat naștere unei noi dimensiuni de cercetare, iar dimensiunea crescândă a datelor biomedicale necesită algoritmi preciși bazați pe inteligență artificială. Studiul realizat în [3] amintește de câteva tehnici de deep learning utilizate pentru a face deosebirea între pacienți diagnosticați cu boala Parkinson’s și cei sănătoși. Printre aceste tehnici se numără k-NN (k-nearest-neighbour utilizat pentru a găsi cel mai apropiat rezultat), NB (Naive Bayes), RT (arbori de regresie) și SVM (mașină cu suport vectorial). Dintre acestea, deși nu sunt specificate cu exactitate valorile, SVM s-a dovedit a avea o performanță medie mai mare decât celelalte.

În afară de utilizarea imaginilor pentru a prezice apariția diverselor mase maligne sau afecțiuni greu detectabile, inteligență artificială are și alte cazuri de utilizare în medicină, precum analiza simptomelor. Analiza simptomelor este utilizată și în studiile efectuate în [4], [5], [6] și [7] care se axează pe utilizarea tehnicilor de machine learning pentru a antrena modele capabile să poată diagnostica un pacient pe baza simptomelor.

În [8] și [9], sunt prezentate două studii concentrate pe utilizarea tehnicilor de inteligență artificială pentru a crea o serie de clasificatoare capabile să analizeze o serie de simptome date ca intrare, urmând ca ieșirea sa fie afecțiunea potrivită intrării. În astfel de cazuri, pentru a testa acuratețea unui model predictiv, acesta se validează după antrenare cu un nou set de date numit set de testare sau validare. Acest nou set trebuie să conțină date ce nu au mai fost văzute de model în trecut, pentru a asigura robustețea acestuia.

Plecând de la acest aspect, un studiu efectuat în [10] prezintă cum modelele de clasificare bazate pe tehnici de deep learning poate ajuta în deciziile luate privind tratarea pacienților ce suferă de boli cardiace. Fiind una dintre principalele cauze ale numărului crescând de decese anual, tratarea și descoperirea bolilor cardiace a devenit o problema importantă în medicină, iar utilizarea inteligenței artificiale poate avea un impact pozitiv în rezolvarea acestei probleme. Cu toate acestea, aceste tehnici au și dezavantaje. Spre exemplu, pentru a fi mai eficienți, algoritmii bazați pe deep learning necesită un set mare de date de antrenare, iar procurarea acestor date nu se poate face decât în timp. Pe de altă parte, acest lucru poate fi și un avantaj, și anume că aceste tehnici devin mai precise odată cu trecerea timpului.

Un studiu asemănător cu cel prezentat în cadrul acestei lucrări îl reprezintă cel menționat în [11], unde s-a dorit utilizarea unor modele predictive pentru a prezice și diagnostica afecțiuni. Avantajul unor astfel de studii pot contribui la lansarea unor inițiative în cadrul sistemului medical pentru a ajuta la diagnosticarea unei afecțiuni în stadiile sale incipiente, permițând astfel personalului medical să recomande tratamente eficace care pot preveni dezvoltarea afecțiunii înainte ca această să devină o complicație. În cadrul acestui studiu s-au utilizat clasificatorul Bayes și arborii de decizie.

Comparativ cu studiul din [11] în care s-a analizat performanța pe un set de date specific axat pe 3 afecțiuni (diabet, boli cardiace și cancer), rezultatele obținute în prezenta lucrare sunt concentrate pe contribuția cu o soluție mai generalizată, luând în calcul 41 de afecțiuni diferite.

În [12] este prezentată o altă abordare privind predicția diagnosticului plecând de la simptomele unui pacient. În cadrul acestui studiu se vor utiliza mai multe tehnici de clasificare și predicție printre care se pot menționa algoritmul k-NN și rețele neuronale convoluționale. Scopul studiului cuprinde afecțiuni precum: diabet, afecțiuni cerebrale și boli de inimă.

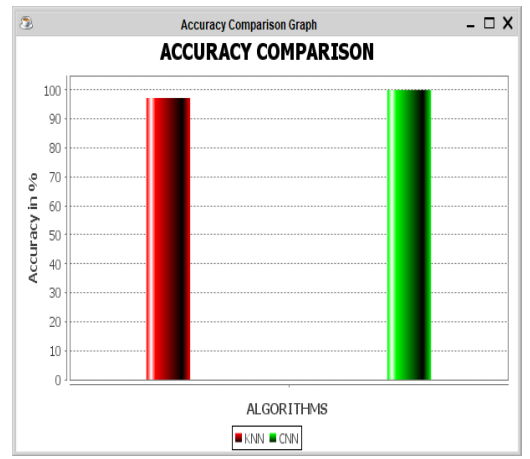


Fig. 1- Rețele neuronale convoluționale vs. k-NN [12]

Așa cum se observă în graficul de mai sus, concluziile rezultatelor din [12] reflectă faptul că performanța în acuratețe a rețelelor neuronale convoluționale este superioară față de algoritmul k-NN, iar în din punct de vedere al eficienței, k-NN este mai lent din punct de vedere al utilizării resurselor decât rețelele neuronale. O alta comparație similară care include clasificatorul Naive-Bayes, k-NN și rețele neuronale este prezentată și în [13].

Ținând cont de rezultatele obținute în prezent, se poate afirma că cercetările continue efectuate în scopul dezvoltării unor soluții de predicție ale bolilor poate aduce un beneficiu pe termen lung atât sistemului medical, dar și pacienților. Pentru aceștia din urmă există deja soluții online bazate pe metode de algoritmi bazați pe inteligență artificială.

# Metodologia abordată

## 3.1. Baza de date utilizată

Setul de date necesar antrenării și validării rețelei neuronale din cadrul aplicației implementate în această lucrare provine de pe website-ul Kaggle [14], o sursă bine cunoscută de date ce pot servi ca seturi de antrenare și validare pentru cercetătorii ale căror proiecte implică analiza și procesarea datelor.

Datele alese pentru această aplicație provin dintr-o bază de date creată în anul 2020 și sunt constituite din simptomele a aproximativ 5000 de pacienți, al căror diagnostic a fost deja evaluat. Datele fiecărui pacient sunt dispuse pe rânduri într-un document de tip .CSV, simptomele fiecărui pacient și diagnosticul corespunzător fiind așezate pe 133 de coloane. Fiecare rând al setului de date conține aproximativ 132 de simptome posibile diferite, având valoarea inițială 0 (inactiv), iar dacă pacientul curent suferă de un anumit simptom, acela este marcat ca 1 (activ). Setul de antrenare constă în aproximativ 4920 de rânduri, iar setul de validare este reprezentat de restul (aproximativ 43).

Pentru a asigura faptul că setul de date de test nu include cazuri similare cu datele de antrenare, se vor elimina duplicatele în momentul antrenării și validării rețelei și recalcula proporțiile în care sunt împărțite aceste date. Acest pas este necesar pentru calculul unor metrici reali de performanță ai rețelei neuronale.

## 3.2. Metodologia sistemului propus

Conform [15], rețelele neuronale sunt un mijloc de a învățarea automată, în care un sistem de calcul învață să îndeplinească unele sarcini analizând exemple de antrenament. Sunt constituite din straturi de intrare, ieșire și straturi ascunse în care neuronii comunică și se activează în funcție de rezultatul la care trebuie să ajungă. În general, nu există control asupra straturilor ascunse utilizate în cadrul unei rețele neuronale, deoarece parametrii sunt ajustați automat pentru a ajunge la rezultatul final. Pentru a minimiza diferențele dintre ieșirea așteptată provenită de la rețea neuronală și ieșirea reală, se pot calcula erori de predicție și, cu ajutorul unor tehnici de optimizare, parametrii rețelei neuronale sunt ajustați astfel încât eroarea să fie minimă.

În general, când se pune problema obținerii unei predicții dintr-un strat curent al unei rețele neuronale, putem considera următoarea formulă matematică (1) de mai jos:

(1)

unde, ***y***este predicția obținută în urma însumării tuturor parametrilor ***w*** cu neuronii stratului precedent ***x*** la care se poate aduna și un parametru adițional numit *bias.*În principiu, fiecare strat curent din parcurgerea rețelei poate fi considerat ieșire la pasul curent, dar și intrare pentru stratul de la pasul viitor.

Pentru obținerea unor rezultate experimentale în cadrul acestei lucrări s-a optat pentru implementarea unei rețele neuronale de tip feed-forward cu scopul de a prezice diagnosticul simptomelor date ca intrare. Pentru această sarcină s-a utilizat limbajul de programare Python, una dintre cele mai populare alegeri atunci când se lucrează cu machine learning. Versatilitatea limbajului Python permite construirea de algoritmi de pre-procesare a datelor și facilitează lucrul cu algoritmi bazați pe deep learning. Pe lângă faptul că sintaxa este intuitiva, decizia de a utiliza Python s-a concretizat și datorită numeroaselor biblioteci existente dezvoltate cu scopul de a oferi unelte pentru lucrul cu rețele neuronale. Dintre acestea, s-a ales biblioteca Pytorch, creată și dezvoltată de divizia de cercetare pentru Machine Learning de la Facebook. Documentația [16] pusă la dispoziție de Pytorch descrie un API bine structurat cu ajutorul căruia procesarea și conversia în tensori a seturilor de date se poate face cu ușurință.

Odată cu alegerea tehnologiilor utilizate se poate afirma că aplicația va fi împărțită în 4 etape de bază:

* Pre-procesarea setului de date și definirea rețelei neuronale
* Antrenarea și validarea rețelei, urmată de salvarea modelului
* Crearea unei arhitecturi de server bazate pe REST API capabilă să primească și să elaboreze răspunsuri pentru o aplicație WEB
* Elaborarea unei interfețe grafice WEB cu ajutorul căreia utilizatorul poate să ceară o predicție a simptomelor de care suferă

În ceea ce privește pre-procesarea setului de date ales, s-au creat două scripturi Python, unul pentru datele de antrenare și unul pentru datele de validare, care iterează prin toate rândurile fișierelor .CSV. La fiecare iterație, deoarece diagnosticul asociat simptomelor este reprezentat ca un șir de litere și ieșirea din rețeaua neuronală va fi alcătuita din date numerice, s-a înlocuit denumirea bolii curente cu indicele acesteia asociat cu ordinea din document (indicii aparținând intervalului [0-40]).

De asemenea s-a construit o hartă a acestor înlocuiri, pentru a avea o evidență asociativă între indici și numele bolii corespunzătoare, pentru a evita o eventuală problemă în care ordinea rândurilor din seturilor de date este schimbată. Această hartă va fi utilizată pentru a identifica numele bolii prezise de rețeaua neuronală, deoarece ieșirea acesteia va fi chiar indicele bolii. Harta va avea forma:

|  |  |
| --- | --- |
| Indice | Nume |
| 0 | Urinary tract infection |
| 1 | Hyperthyroidism |
| 2 | Malaria |
| ... | ... |
| 39 | Chicken pox |
| 40 | GERD |

Tabel 1. Harta asocierii indicilor cu numele bolilor

După obținerea seturilor de date în formatul dorit și alcătuirea hărții precizate mai sus, s-a creat o rețea neuronală ca o clasă denumită Net, și cu ajutorul bibliotecii Pytorch, s-a definit funcția de inițializare a rețelei. Rețeaua a fost împărțită pe 4 straturi astfel:

1. Primul strat prezintă 132 noduri sau caracteristici la intrare ceea ce reprezintă numărul de simptome posibile, iar la ieșire vor exista 64 de noduri.
2. Al doilea strat primește la intrare cele 64 de noduri din primul strat, și emite la ieșire un număr egal.
3. Al treilea strat este identic ca specificații cu stratul numărul 2.
4. Al patrulea și ultimul strat primește la intrare cele 64 de noduri de la stratul precedent și la ieșire va avea 41 de noduri ceea ce reprezintă rezultatul rețelei.

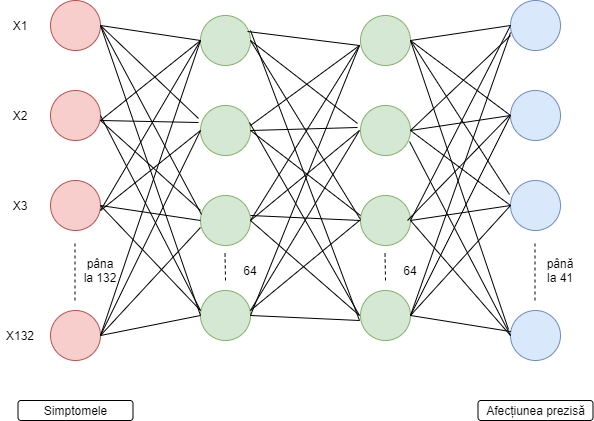


Fig. 2 - Structura rețelei neuronale utilizată

Funcția de activare utilizată pentru a transforma ieșirea unui strat în intrarea următorului poartă denumirea de *ReLU,* iar la final se va utiliza *softmax*. Funcția de activare este utilizată conform schemei următoare:

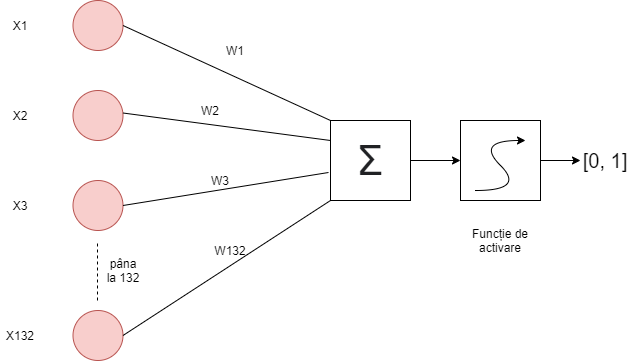


Fig. 3 - Rolul funcției de activare în obținerea datelor de intrare / ieșire

Precum este amintit și în [17], o atribuție importantă a funcțiilor de activare este de a menține datele într-un interval controlat.

Primele 3 straturi folosesc la ieșire funcția de activare *ReLU*, care este o funcție liniară (pentru valori mai mari decât 0) folosită să transmită intrarea din stratul curent nealterată următorului strat dacă intrarea este pozitivă, iar în caz contrar va transmite 0. Ieșirii din ultimul strat i se va aplica *softmax* pentru a scala ieșirea în intervalul [0, 1], obținând astfel procentaje care însumate vor rezulta maxim 1.

Deci, predicția finală a rețelei neuronale va fi astfel: *argmax(softmax(ieșire))* care se traduce prin valoarea maximă a ieșirilor posibile din rețeaua neuronală.

# Implementarea soluției

## 4.1. Antrenarea și validarea rețelei

Pentru a realiza procesul de antrenare și testare, s-a utilizat un calculator bazat pe sistemul de operare macOS, ce dispune de un procesor Intel i5 generația a 10-a de 2GHz. În urma execuției comenzii *time* din UNIX (folosită pentru a cronometra durata unui proces), timpul necesar antrenării și salvării modelului pe acest procesor, utilizând toate cele 4920 de rânduri pentru antrenare și 43 pentru validare, este de aproximativ 11.594 secunde, iar timpul necesar parcurgerii setului de testare este aproximativ 1.914 secunde. Pornind de la rețeaua definită anterior și datele pre-procesate, s-a realizat un script care declară o instanță a clasei Net și încarcă datele de antrenare și validare pentru a putea fi utilizate.

Utilizând biblioteca Pytorch, s-au convertit caracteristicile intrărilor și ieșirilor într-un tip de date numit tensor. Tensorii sunt un tip de date utilizați în machine learning și pot fi comparați cu vectori sau matrice, fiind distribuiți pe linii și coloane.

Datele de antrenare au fost încărcate în loturi de câte 10 rânduri de simptome, și 10 ieșiri corespunzătoare fiecărui rând. S-a specificat parametrul *shuffle=true,* a cărui funcție este de a selecta arbitrar datele, asigurând astfel un proces uniform de învățare. Seturile de date dispun de forma:

tensor([[1., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,0.,0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,0., 0., 0.,0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,0., 0., 0., 0.,0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.,0., 0., 0., 1., 0.,0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,0., 0., 0., 0., 0., 1.,0., 0., 0., 0., 1., 0.]]), tensor([[22.]])

unde tensorul cu valori cuprinse în intervalul [0, 1] reprezintă caracteristicele intrării, iar tensorul cu valoarea 22, reprezintă ieșirea așteptată a rețelei.

Procesul de antrenare al rețelei neuronale constă într-un bloc iterativ *for,* care se repetă timp de 3 epoci. O epocă este echivalentul a unei parcurgeri complete prin setul de date, care la rândul său conține loturi de 10. S-au ales 3 epoci pentru parcurgere deoarece un număr prea mic poate rezulta în abilități scăzute de învățare ale rețelei, iar un număr prea mare poate induce riscul ca rețeaua să învețe întocmai setul de date, iar predicțiile să sufere în acuratețe pe viitor.

În cadrul acestui bloc, se iterează prin datele de antrenare și se separă în 2 componente, *X* pentru caracteristici și *y* pentru țintele acestora. La fiecare pas, componenta *X* este oferită ca intrare rețelei neuronale, iar ieșirea este comparată cu ținta *y* pentru a calcula eroarea de predicție. Eroarea de predicție este necesară pentru a optimiza parametrii *W* ai rețelei neuronale astfel încât acuratețea prezicerii să fie îmbunătățită. Acest lucru se realizează aplicând eroarea de predicție parametrilor parcurgând invers rețeaua neuronală și optimizându-le cu optimizatorul *Adam*. Gradientul este mereu reactualizat cu valoarea 0 pentru a nu re-optimiza pașii anteriori. *Adam* este un algoritm de optimizare folosit pentru deep learning care actualizează parametrii rețelei în funcție de gradient și eroarea de predicție la fiecare pas. Procesul de antrenare este redat de următoarea secțiune de pseudocod:

|  |
| --- |
| SET optimizer TO optim.Adam(net.parameters(), lr=0.001)  FOR epoch IN range(3):  FOR data IN trainset:  SET X, y TO data  net.zero\_grad()  SET output TO net(X)  SET loss TO F.nll\_loss(output, y.squeeze().long())  loss.backward()  optimizer.step() |

unde *net* este instanța clasei rețelei neuronale și *loss* este eroarea de predicție. De precizat că nu este obligatorie reinițializarea gradientului la 0.

Validarea rețelei s-a realizat într-o manieră similară, iterând prin tensorii distribuiți în loturi de câte 10 în datele de testare. De această dată, nu s-au mai efectuat optimizări sau calcule ale erorilor de predicție, în schimb s-a realizat un calcul procentual al preciziei rețelei neuronale în validare. Acuratețea s-a calculat ca un raport între numărul de predicții corecte ale rețelei (raportat la țintele așteptate în datele de validate) și numărul total de predicții. Validarea s-a efectuat cu ajutorul secțiunii de cod în pseudocod de mai jos:

|  |
| --- |
| with torch.no\_grad():  FOR data IN testset:  SET X, y TO data  SET output TO net(X)  FOR idx, i IN enumerate(output):  IF torch.argmax(i) EQUALS y[idx]:  correct += 1  total += 1  SET accuracy TO round(correct/total, 3)  OUTPUT("Accuracy: ", accuracy) |

Modelul antrenat a fost salvat într-un fișier denumit *model.pth* pentru a putea fi reutilizat în construirea aplicației.

## 4.2. Serverul aplicației WEB

Pentru a realiza un sistem interactiv pe care un utilizator îl poate folosi în prezicerea unui diagnostic pe baza simptomelor sale, este necesar un server în cadrul căruia se va elabora un REST API. În cadrul acestui API se va încărca modelul rețelei neuronale antrenat și validat la pașii anteriori, și va fi utilizat pentru a prezice diagnosticul unor simptome noi date la intrare din interfața grafică.

|  |  |  |
| --- | --- | --- |
| Endpoint / Metoda HTTP | POST | GET |
| api/v2/predict |  |  |
| api/v2/symptoms |  |  |

Comunicarea între server și interfața grafică va urma conceptele REST care impun transferul de informații utilizând protocolul HTTP și metodele sale: GET, POST. Serverul va fi creat utilizând biblioteca Flask din Python, care pune la dispoziție uneltele necesare pornirii unui server WEB capabil să asculte cereri și să elaboreze răspunsuri cu mai mulți clienți. Astfel, au fost definite două rute pentru accesarea resurselor necesare, rute ce vor fi apelate de interfața grafică:

Tabel 2. Rutele puse la dispoziție de server

unde ruta *symptoms,* va întoarce o listă cu toate cele 132 de simptome posibile, iar ruta *predict* va prelua datele transmise din interfața grafică, le va procesa într-un format acceptabil pentru rețeaua neuronală, și va trimite înapoi drept răspuns predicția rețelei în urmă procesării simptomelor trimise.

Odată ce rețeaua neuronală va prezice o afecțiune, ieșirea acesteia va fi indicele bolii cu ajutorul căruia vom identifica numele din harta creată la pașii anteriori.

## 4.3. Interfața grafică a aplicației WEB (Frontend)

Partea de frontend reprezintă aplicația din browser ce dispune de o interfață grafică prin care utilizatorii pot selecta simptome din lista de 132 puse la dispoziție pentru a obține o predicție privind afecțiunea de care ar putea suferi.

Pentru implementarea interfeței grafice s-a optat pentru React, o bibliotecă de Javascript dezvoltată și menținută de Facebook. React, cum se menționează și în documentația oficială din [18], oferă capacități versatile de a construi o interfață de utilizator bazându-se pe concepte numite “stări” care reprezintă starea unei componente din interfață la un moment de timp. Programatorul are libertatea de a decide prin ce eveniment componenta poate trece la o altă stare, moment în care componenta este actualizată cu noile informații.

Interfața grafică este una simplă ce constă într-o listă de simptome ce pot fi selectate și un buton utilizat pentru a trimite cererea către server pentru a prezice afecțiunea.

Fiind multe simptome, a fost implementată și o funcționalitate de filtrare pe baza numelui, în funcție de cuvintele cheie introduse de utilizator. Simptomele selectate vor fi afișate pe un rând aflat deasupra listei de simptome, așa cum se poate observa mai jos:

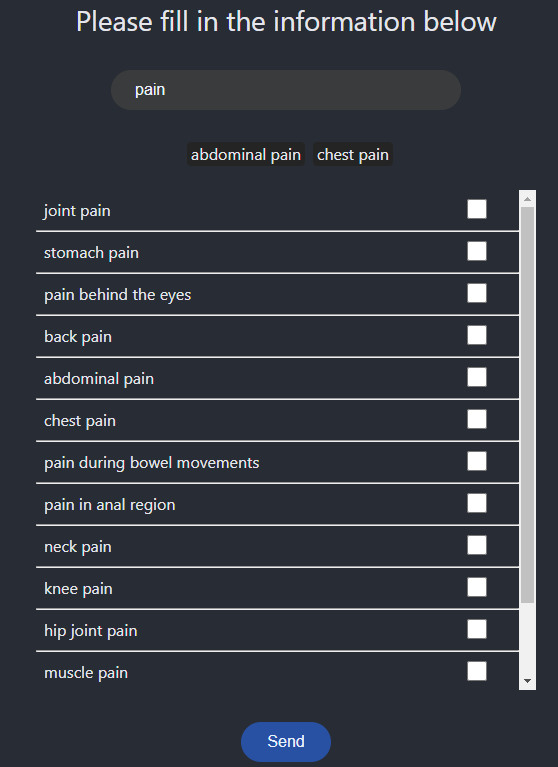


Fig. 4 - Interfața grafică a aplicatiei și funcția de filtrare după termenul „pain” (durere)

Odată selectate simptomele potrivite, utilizatorul poate apăsa pe butonul de la baza listei pentru a trimite informațiile alese către server. Datele trimise către server au forma:

{"symptoms":[{"id":0,"symptom":"itching"},{"id":1,"symptom":"skin rash"}],"ids":[0,1]}

Pe server sunt preluate datele din câmpul *ids,* care corespund cu indicii simptomelor de care pacientul suferă. Acești indici sunt necesari pentru a parcurge întreaga listă de 132 de simptome și atribuirea valorii 1 (activ) pentru simptomele care corespund acestor indici, și 0 (inactiv) în rest. Astfel se va obține un tensor de lungime 132, care va fi folosit ca intrare pentru rețeaua neuronală.

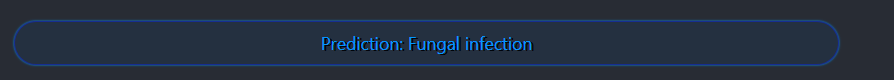
Rețeaua neuronală va prezice un indice care va fi folosit să identifice numele bolii corespunzătoare din harta de la pașii anteriori. Ca ultim pas, serverul va trimite predicția rețelei înapoi către interfața grafică, iar utilizatorul va fi notificat astfel:  


Fig. 5 - Predicția rețelei pe baza simptomelor alese de utilizator

# Rezultate experimentale și discuții

În urma parcurgerii rețelei prin setul de testare pre-procesat anterior s-a obținut o acuratețe a predicției de 97.6%, număr ce reprezintă un acuratețea totală de predicție a rețelei. Acuratețea s-a obținut prin următoarea formulă (2):

acuratețea = (2)

Desigur că, valoarea acurateței obținute mai sus este influențată puternic de faptul că setul de date poate conține anumiți pacienți cu simptome identice pentru același diagnostic. Drept urmare, pentru evita un overfitting, a fost necesară adăugarea unui pas adițional în pre-procesarea datelor și recalculați metricii de performanță pentru a evidenția performanțele rețelei neuronale. Pașii adiționali de pre-procesare efectuați pentru calculul precis al metricilor de performanță presupun:

1. suplimentarea datelor de testare cu ajutorul datelor de antrenare pentru a obține o proporție mai favorabilă în ceea ce privește numărul de date de testare.
2. crearea unei funcții de verificare a numărului de apariții ale unei boli în setul de testare pentru a asigura o distribuție egală pentru fiecare afecțiune la validare.
3. eliminarea datelor adăugate în setul de testare din setul de antrenare pentru a evita antrenarea și validarea pe date identice.

În urma acestor etape adiționale proporția seturilor de date a devenit 4552 de pacienți pentru antrenare și 410 pacienți pentru testare. În acest punct, a mai fost nevoie de o ultimă etapă de filtrare a datelor de antrenare pentru a elimina numărul de rânduri duplicate ca simptome, și anume: iterarea prin toate rândurile matricei datelor de antrenare și filtrarea unora dintre ele astfel încât să se minimizeze numărul de rânduri identice cu datele de testare. Filtrarea a fost realizată în 2 variante cu proporțiile datelor diferite, iar împărțirea acestora a afectat acuratețea rețelei neuronale astfel:

1. 1254 pacienți pentru setul de antrenare și 164 de pacienți pentru testare. În acest caz, numărul de pacienți unici selectați pentru fiecare afecțiune a fost 4. Drept urmare numărul 164 a fost obținut înmulțind numărul de boli (41) cu numărul de pacienți pentru fiecare, adică 4.
2. 837 pacienți pentru setul de antrenare și 410 de pacienți pentru testare. În acest caz, numărul de pacienți unici selectați pentru fiecare afecțiune a fost 10.

Pentru calculul metricilor de performanță specifice problemelor de clasificare și predicție utilizând machine learning, este necesar calculul matricei de confuzie. Matricea de confuzie reprezintă un tabel utilizat pentru a măsura performanța modelelor de clasificare și predicție, oferind informații privind frecvența corectitudinii predicțiilor. Matricea de confuzie are forma generală:

# 

Fig. 6 - Matricea de confuzie generală [19]

Forma generalizată a matricei de confuzie din figura de mai sus presupune lucrul cu 2 clase de clasificare, spre exemplu: pozitiv sau negativ, de aceea matricea are forma 2x2. Adaptând terminologia matricei de confuzie la contextul problemei abordate în cadrul acestei lucrări, trebuie specificat faptul că cele 41 de boli posibile pe care rețeaua neuronală le poate prezice vor determina o matrice de confuzie cu 41 de clase. Acest lucru implică o formă a matricei de 41x41.

Lucrând cu o matrice de confuzie bazată pe 41 de clase, calculul performanței soluției rețelei neuronale se va diviza în 41 de subprobleme, urmând să calculăm metrici de performanță pentru fiecare subproblema în parte. Acest lucru se poate traduce prin măsurarea performanței de predicție a fiecărei boli. Astfel, după calculul matricei de confuzie, se vor calcula individual senzitivitatea, specificitatea, precizia și scorul F1.

După obținerea predicțiilor rețelei în urma parcurgerii datelor de test, s-au creat 2 vectori ce conțin predicțiile și diagnosticul real pe baza simptomelor. Cu ajutorul acestor vectori și biblioteca Python *sklearn.metrics,* s-a putut genera o matrice de 41x41, a căror valori diferite de 0 se află predominant pe diagonala principală, cu mici excepții. Acest lucru se datorează faptului că acuratețea rețelei este mare, deci predicțiile false sunt minime.

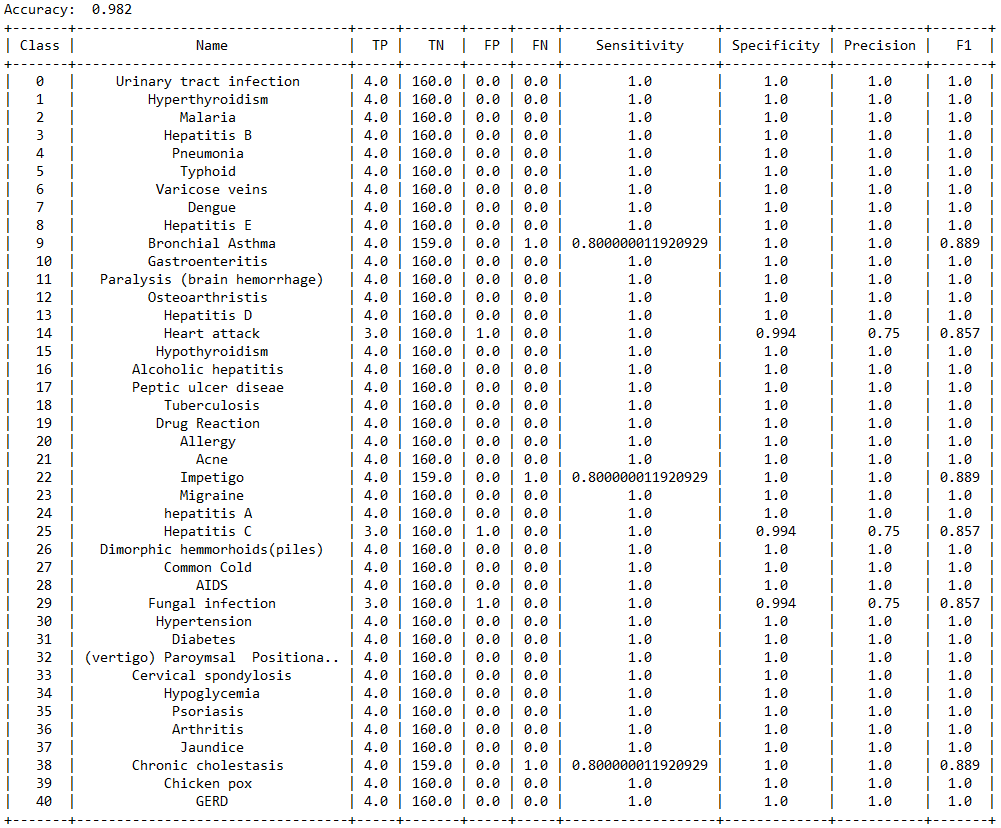
Iterând prin cele 41 de clase ale matricei (bolile posibile), a fost creată o buclă *for* care la fiecare pas calculează valorile:

* TP, unde predicția bolii curente a coincis cu simptomele în mod corect
* TN, unde rețeaua a stabilit corect faptul că boala curentă nu se potrivește cu simptomele
* FP, unde rețeaua a prezis în mod eronat că o boală se potrivește cu simptomele
* FN, unde rețeaua a prezis în mod eronat că o boală nu se potrivește cu simptomele

Odată ce s-au obținut obținut valorile de mai sus (specifice fiecărei iterații prin clasele matricei), s-a utilizat următoarele formule pentru calculul:

* senzitivitatea =
* specificitatea =
* precizia =
* scorul F1 =

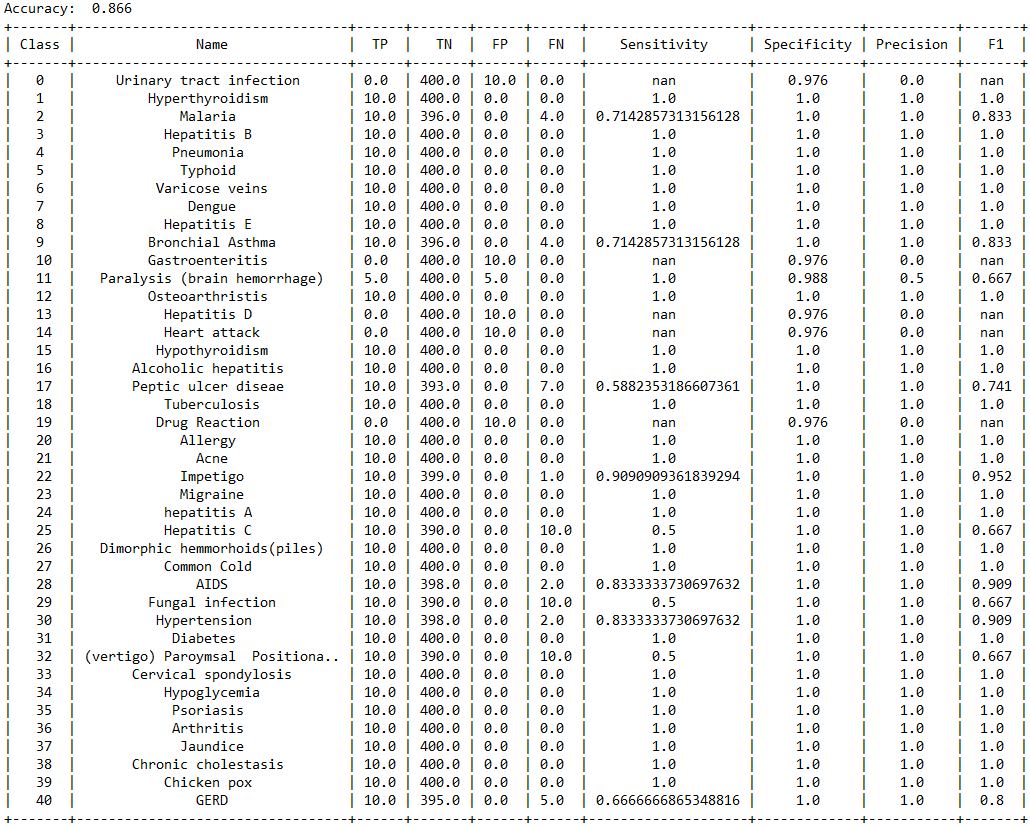
Rezultatele calculelor de mai sus obținute iterativ pentru toate cele 41 de clase, atât pentru scenariul 1 de împărțire al datelor menționat mai sus, cât și pentru scenariul 2, au fost salvate și afișate în 2 tabele pentru a evidenția corelația între clasa sau numele bolii și metricii de performanță calculați. Tabelele au următoarea formă:



Tabel 3. Calculul metricilor de performanță (1254 date de antrenare și 164 date de test)

După cum se poate observa, acuratețea rețelei prezintă o valoare bună de aproximativ 98.2% chiar și în urma eliminării numărului de rânduri cu simptome duplicate. Astfel, se poate deduce că și pentru proporția inițială de date, riscul de overfitting este mic.

Se poate observa în tabelul de mai sus cum pentru atacul de cord, doar 3 dintre cele 4 apariții au fost corect identificate, iar într-un caz, clasa 9 (astmul bronșic) a fost confundat cu atacul de cord. Prin urmare, în acest scenariu, atacul de cord are un fals pozitiv și astmul bronșic are un fals negativ.



Tabel 4. Calculul metricilor de performanță (837 date de antrenare și 410 date de test)

Diferența în performanță între cazurile 1 și 2 reprezentate prin tabelele de mai sus este considerabilă. Se poate observa cum proporționarea mai corectă a datelor de antrenare și testare poate avea un impact mare asupra abilității rețelei neuronale de a prezice boli. În tabelul 4 de mai sus, există unele clase în cadrul cărora predicția nu a putut obține nici măcar un singur caz pozitiv, având în schimb multe cazuri fals pozitive. Acest lucru afectează negativ metricile precum senzitivitatea și scorul F1, deoarece raportul final de calcul ajunge în forma , efectul fiind apariția valorii *nan* în tabel.

În ceea ce privește tabelul 3, rezultate sunt mai bune obținând o acuratețe de 98.2% comparativ cu cazul al doilea unde acuratețea este de 86.6%.

Comparativ cu alte abordări utilizând același set de date de pe Kaggle precum [20], performanțele obținute sunt similare în ceea ce privește valoarea acurateței totale a rețelei (aprox. 97.6 % în ambele cazuri, dacă este să consideram utilizarea setului de date nefiltrat). De asemenea, se poate afirma că rețeaua neuronală din această lucrare prezintă un avantaj din punct de vedere al utilizării comparativ cu rețeaua utilizată în [12], în special pentru că în [12] studiul se axează pe un număr mai restrâns de boli, în timp ce modelul antrenat în cadrul acestei lucrări se concentrează pe un uz mai general, fiind capabil să recunoască până la 41 de boli.

# Concluzii și dezvoltări ulterioare

Această aplicație aduce medicina mai aproape de pacient permițând oricui să determine un diagnostic pe baza simptomelor fără să fie nevoie să consulte un medic specialist. Interfața grafică oferă o „punte” de comunicare între utilizator și modelul rețelei neuronale. Desigur că diagnosticul obținut este preliminar și este recomandată validarea diagnosticului un medic specialist.

În ceea ce privește performanța programelor create în cadrul aplicației, sunt de părere că există metode mai optim e de a procesa și parcurge seturile de date descărcate. Complexitatea maximă calculată a programului de pre-procesare este O(n3), deoarece sunt efectuate 3 blocuri iterative de tip *for.*

Se poate afirma că în urma definirii și antrenării rețelei neuronale, rezultatele obținute în ceea ce privește performanța și acuratețea rețelei sunt bune comparativ cu rezultatele obținute în [21], unde rețeaua neuronală artificială utilizată a obținut o acuratețe de 77%. În ceea ce privește scorul F1 pentru fiecare clasă în parte, se poate observa că predomină valoarea 1, ceea ce reprezintă performanță maximă pentru clasificarea claselor respective.

Se poate observa că utilizarea unui set mai mare de date, atât pentru antrenare, dar mai ales pentru testare poate rezulta în performanțe îmbunătățite a modelului, iar concentrarea pe o arie mai restrânsă de boli poate contribui, de asemenea, la performanțe. Deoarece setul de date utilizat este mic comparativ cu cel de antrenare, este greu de stabilit în ce măsură performanțele bune obținute se vor reflecta și în cazuri reale. Utilizarea unui set mai mare de date pentru testare poate afecta metricii de performanță calculați în capitolul anterior

Sistemul construit în cadrul acestei lucrări reprezintă o contribuție pentru studiile aplicabilității tehnicilor de machine learning în medicină, fiind capabil să prezică 41 de afecțiuni umane comune pe baza unor simptome. Aplicația poate fi utilizată ca o soluție completă dispunând atât de o rețea neuronală, un server pentru procesarea datelor și o interfață grafică ușor de folosit.

În ceea ce privește o eventuală dezvoltare ulterioară, aceasta poate fi schimbarea abordării privind tipul de rețea neuronală utilizată. Momentan, această aplicație utilizează un tip de rețea neuronală feed-forward (liniară), acest lucru constituind o altă diferență comparativ cu studiul efectuat în [12], iar trecerea către o rețea neuronală convoluțională cu mai multe straturi poate contribui la îmbunătățirea rezultatelor obținute.

O altă direcție de dezvoltare ulterioară poate fi concentrarea pe un număr mai restrâns de afecțiuni umane precum bolile cardio-vasculare ( [21] ), bolile cronice precum diabetul sau chiar afecțiunile neurologice sau pe sistem nervos. Minimizarea caracterului generalizat de a lua în calcul mai multe afecțiuni comune de care dispune modelul antrenat în această lucrare poate avea un impact pozitiv în creșterea performanțelor. Utilizarea unui set de date axat pe un număr mai restrâns de boli pentru antrenarea unui model specializat poate constitui o contribuție semnificativă în cercetările din zilele noastre pentru prevenirea afecțiunilor cronice.

# Bibliografie

|  |  |
| --- | --- |
| [1] | A. Krogh, „What are artificial neural networks?,” *Nat Biotechnol 26,* p. 195–197, 2008. |
| [2] | D. Drygin și N. Pronkin, „Application of artificial intelligence in medicine,” *International Journal of Professional Science №1-2020,* nr. UDC 67, 2020. |
| [3] | C. Park, C. C. Took și J.-K. Seong, „Machine learning in biomedical engineering,” *Biomedical Engineering Letters,* vol. 8, nr. 1, pp. 1-3, 2018. |
| [4] | S. Chae, S. Kwon și D. Lee, „Predicting Infectious Disease Using Deep Learning and Big Data,” *International Journal of Environmental Research and Public Health,* vol. 15, 2018. |
| [5] | M. CHEN, Y. HAO, K. HWANG, L. WANG și L. WANG, „Disease Prediction by Machine Learning Over Big Data From Healthcare Communities,” *IEEE Access,* 2017. |
| [6] | G. Battineni, G. G. Sagaro, N. Chinatalapudi și F. Amenta, „Applications of Machine Learning Predictive Models in the Chronic Disease Diagnosis,” *Journal of Personalized Medicine,* vol. 10, nr. 21, pp. 1-11, 2020. |
| [7] | C. Kim, Y. Son și S. Youm, „Chronic Disease Prediction Using Character-Recurrent Neural Network in The Presence of Missing Information,” *applied sciences,* vol. 9, pp. 1-17, 2019. |
| [8] | K. R. Foster, R. Koprowski și J. D. Skufca, „Machine learning, medical diagnosis, and biomedical engineering research,” *BioMedical Engineering OnLine,* nr. 94, 2014. |
| [9] | S. S. Shirsath și P. S. Patil, „Disease Prediction Using Machine Learning Over Big Data,” *International Journal of Innovative Research in Science, Engineering and Technology,* vol. 7, nr. 6, pp. 1-6, 2018. |
| [10] | A. K. Gárate-Escamila, A. H. E. Hassani și E. Andrès, „Classification models for heart disease prediction using feature selection and PCA,” *Informatics in Medicine Unlocked,* vol. 19, 2020. |
| [11] | V. Jackins, S. Vimal și M. Kaliappan, „AI-based smart prediction of clinical disease using random forest classifier and Naive Bayes,” *J Supercomput,* vol. 77, p. 5198–5219, 2021. |
| [12] | D. Dahiwade, P. G. Patle și P. E. Meshram, „Designing Disease Prediction Model Using Machine Learning Approach,” în *International Conference on Computing Methodologies and Communication*, 2019. |
| [13] | S. Uddin, A. Khan, M. E. Hossain și M. A. Moni, „Comparing different supervised machine learning algorithms for disease prediction,” *BMC Medical Informatics and Decision Making,* vol. 19, nr. 281, pp. 1-16, 2019. |
| [14] | K. KAUSHIL268, „Disease Prediction Using Machine Learning,” 15 May 2020. [Interactiv]. Available: https://www.kaggle.com/kaushil268/disease-prediction-using-machine-learning. [Accesat 20 April 2021]. |
| [15] | L. Hardesty, „Explained: Neural networks,” 14 April 2017. [Interactiv]. Available: https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414. [Accesat 24 April 2021]. |
| [16] | F. Torch Contributors, „PYTORCH DOCUMENTATION,” Facebook, 2021. [Interactiv]. Available: https://pytorch.org/docs/stable/index.html. [Accesat 16 April 2021]. |
| [17] | S. Sharma, „ACTIVATION FUNCTIONS IN NEURAL NETWORKS,” *International Journal of Engineering Applied Sciences and Technology,* vol. 4, nr. 12, pp. 310-316, 2020. |
| [18] | Facebook, „Getting Started - React,” 2021. [Interactiv]. Available: https://reactjs.org/docs/getting-started.html. [Accesat 15 April 2021]. |
| [19] | S. Narkhede, „Understanding Confusion Matrix,” 9 May 2018. [Interactiv]. Available: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62. [Accesat 4 May 2021]. |
| [20] | R. Agrawal, „Disease Prediction using ML,” May 2021. [Interactiv]. Available: https://www.kaggle.com/rxsraghavagrawal/disease-prediction-using-ml. [Accesat 4 May 2021]. |
| [21] | A. U. Haq, J. P. Li, M. H. Memon, S. Nazir și R. Sun, „A Hybrid Intelligent System Framework for the Prediction of Heart Disease Using Machine Learning Algorithms,” *Hindawi Mobile Information Systems,* vol. 2018, pp. 1-22, 2018. |
| [22] | C. C. Aggarwal, Neural Networks and Deep - A textbook, New York: Springer International Publishing , 2018. |