Numele proiectului : **Disease prediction**

**Echipa**:

1. Stroiescu Cosmin-Ilie-> baza de cunoștințe
2. Tîrîși Claudiu-Laurențiu-> mașină de inferență
3. Stoinel Bogdan-Cristian->interfața web

**Scopul proiectului:**

Scopul proiectului este realizarea unui sistem expert în care utilizatorul poate introduce simptome, respectiv caracteristici fizice ale sale, și care pe baza faptelor introduse va găsi cea mai plauzibilă boală pe care o poate avea persoana respectivă.

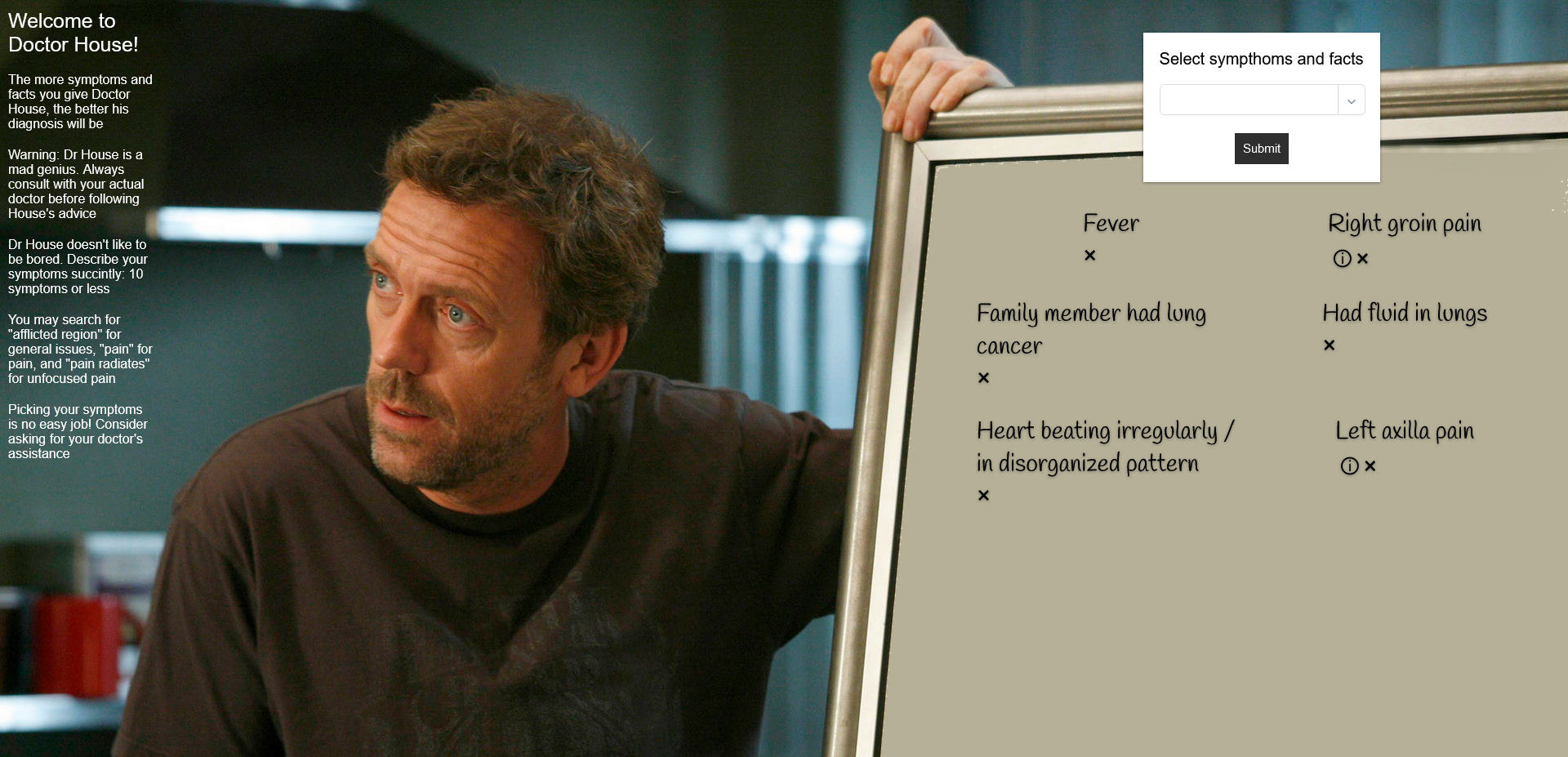
**Detalii tehnice:**

**Mașina de inferență** este un model de **Deep Learning**, format din 4 blocuri de tip **ResBlock**, fiecare astfel de bloc fiind format din 4 grupuri de straturi, anume un **strat liniar, un strat de Batch Normalization, ReLU, și în final Dropout**. Detalii despre funcționarea straturilor se pot găsi in documentația bibliotecii Pytorch. Straturile de BN și Dropout au rolul de a preveni memorarea, modelul având un număr mare de parametri (1.6 milioane), ceea ce se apropie de punctul teoretic în care modelul ar fi in stare să memoreze întreg setul de date, fără a generaliza neapărat aceste capabilități și pe intrări ce nu aparțin setului de date. Pregătirea mașinii de inferență utilizată biblioteca **PyTorch**.

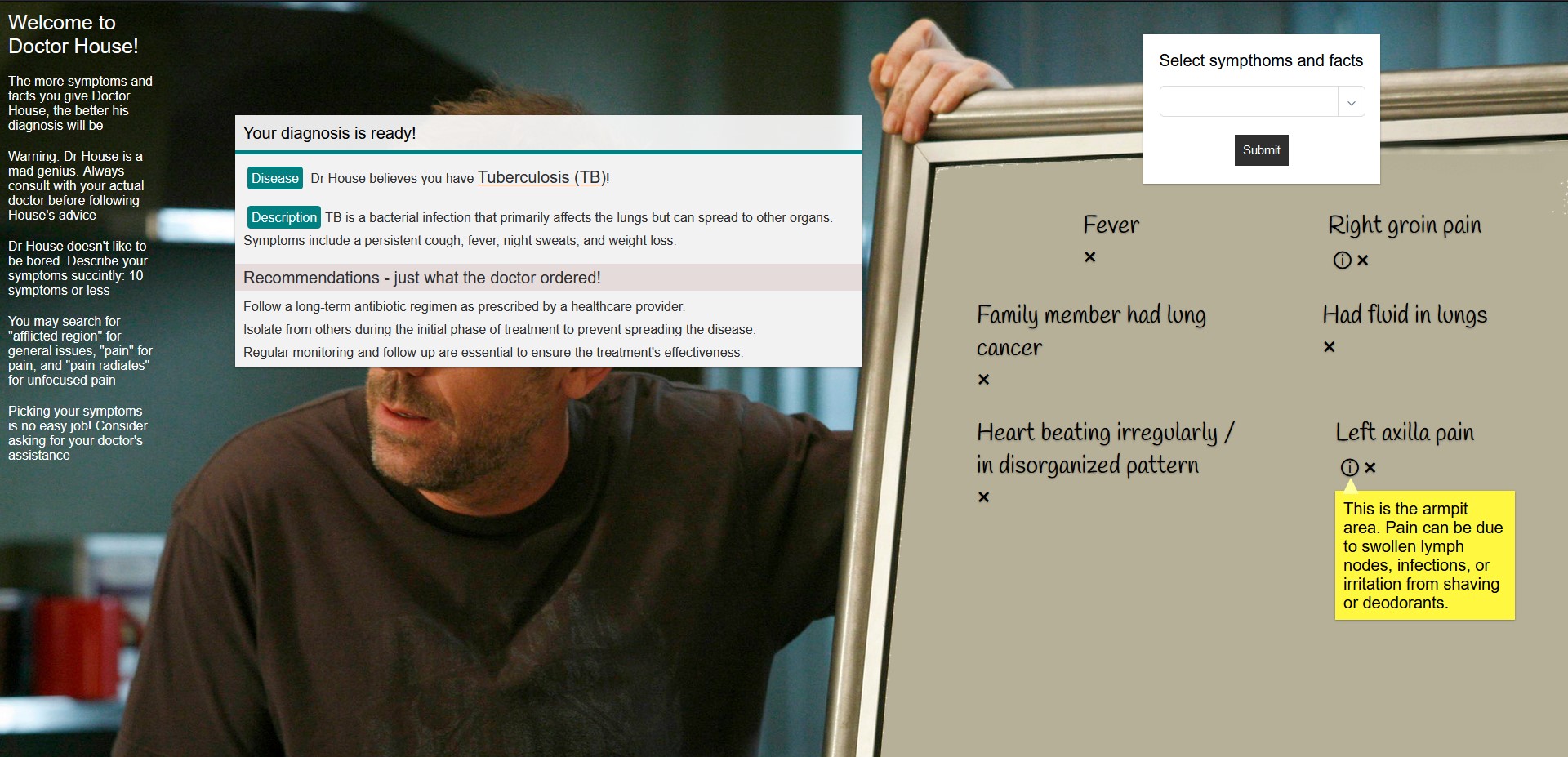
**Antrenarea mașinii de inferență** a fost realizată timp de aproximativ **700 de epoci** (adică modelul a trecut de 700 de ori peste fiecare exemplu), cu **optimizatorul SGD** (metoda clasică ce împinge parametrii în sensul invers al gradientului). Funcția de cost utilizată este **Cross-Entropy**, deoarece mașina de inferență modelează distribuția de probabilitate a bolilor condiționată de simptome. Particularitatea interesantă despre antrenare este că există o probabilitate ca unul dintre simptomele curente să fie eliminate, cu scopul de a avea un model ce face preziceri rezonabile chiar și atunci când utilizatorul nu a introdus un număr prea mare de simptome. Codul specific antrenării mașinii de inferență poate fi găsit în fișierul „**model.ipynb**”, care este un Jupyter Notebook (<https://jupyter.org/>).

**Setul de date** utilizat este setul de date **DDXPlus** ( [[1]](#DDXPlus) ), cel mai mare set de date medical pe care l-am găsit, conținând peste 1 milion de pacienți. Pentru fiecare pacient, setul de date pune la dispoziție o listă de simptome, și boala ce s-a dovedit că o are acel pacient. Setul de date a fost convertit din formatul **CSV** în formatul specific librăriei PyTorch (adică 2 tensori, unul ce conține toate simptomele, și unul ce conține boala, evident simptomele și bolile fiind codificate numeric). Codul ce realizează această transformare se află în fișierul „**dataset-preparation.ipynb**”.

**Interfața** este o pagină web, realizată în HTML-CSS și Javascript. Partea dreaptă a paginii conține un selector de simptome. Utilizatorul poate căuta prin simptome (căutarea suportă cuvinte incomplete, și nu ține cont de ordinea cuvintelor). Simptomele apar pe tabla Doctorului. Acestea pot fi apoi șterse, iar unele dintre ele au și o mică descriere. Utilizatorul trebuie să selecteze un minim de 3, și un maxim de 10 simptome. Majoritatea simptomelor sunt de tip binar (pacientul fie le are, fie nu), dar există și câteva unde utilizatorul trebuie să selecteze o valoare din mai multe opțiuni (spre exemplu, simptomul **Pain intensity**). Codul ce realizează adăugarea simptomelor se află în principal în metoda locală **handler** din metoda **add\_listener** din fișierul **script.js**.



**Conexiunea dintre interfață și mașina de inferență** este realizată prin convertirea modelului din formatul specific **Pytorch** în formatul **Onnx**, care poate fi utilizat în browser, cu ajutorul librăriei **onnx-runtime.** Atunci când utilizatorul apasă butonul **Submit,** un vector de simptome este creat pe baza simptomelor selectate de utilizator (care sunt înainte memorate într-un array și un dicționar, pentru că sunt de 2 tipuri, cum am menționat mai sus). Vectorul este transmis modelului, care returnează distribuția de probabilitate a bolilor. Boala prezisă este selectată utilizând operația **argmax**, apoi detaliile despre boala sunt afișate în partea stângă a interfeței. Funcționalitatea descrisă mai sus poate fi găsită în metoda **attachSubmitHandler** din fișierul **script.js**.



**Pentru utilizarea sistemului** este necesară crearea unui server (evident, dacă sistemul ar fi disponibil publicului, acesta ar fi deja încărcat pe un server). Existența unui server este necesară pentru încărcarea fișierului **onnx** și a fișierelor **json,** browserele moderne nepermițând altfel încărcarea acestora. Cea mai simplă metodă de creare a serverului este utilizarea unui server de python prin rularea comenzii **python –m http.server** în folderul **frontend**, urmată de navigarea la [**http://localhost:8000/**](http://localhost:8000/)(comanda python va menționa care este portul corect).

**Utilizatorii țintă:**

Utilizatorii țintă sunt în principal persoanele care prezintă diverse simptome medicale și doresc să afle de ce boală ar putea suferi. Cazul ideal de utilizare al sistemului este atunci când un specialist medical, împreună cu pacientul, introduce în sistem simptomele acestuia, deoarece în practică este destul de dificil pentru un pacient să numească toate simptomele pe care le prezintă. Pacientul este posibil să aibă simptome precum „commisure pain”, dar este posibil să numească simptomul „durere de cap”, pentru că acesta se referă la un grup de nervi ce conectează cele 2 emisfere ale creierului.

**Concluzie**:

Sistemul este funcțional, în sensul că prezice cu succes boala de care suferă pacientul atunci când simptomele introduse sunt compatibile. Am identificat 3 posibile probleme ale sistemului:

* este extrem de „sigur pe sine” în sensul că distribuția de probabilitate pe care o generează dă mereu o probabilitate mare uneia dintre boli, chiar dacă simptomele au fost introduse aleator. Această problemă este un artefact al metodei de antrenare, care minimizează entropia dintre distribuția generată de model, și distribuția reală a bolilor (care este concentrată în boala de care a suferit pacientul). O posibilă soluție este utilizarea unui diagnostic diferențial, în care doctorii dau probabilități fiecăreia dintre boli. Un model antrenat cu astfel de date ar avea și posibilitatea de a calcula un factor de certitudine (cât de sigur este modelul de diagnosticul pus), prin entropia distribuției de ieșire.
* nu are posibilitatea de a spune că pacientul nu suferă de nicio boală. Setul de date utilizat nu conține astfel de intrări. O posibilă soluție este îmbunătățirea setului de date prin introducerea de intrări în care simptomele sunt aleatoare, și considerând că acestora le corespunde diagnosticul „Nicio boală”. A 2-a soluție este utilizarea factorului de certitudine discutat anterior.
* găsirea simptomelor poate fi dificilă. Sistemul suportă un număr mare de simptome (peste 800), iar oamenii neantrenați în domeniul medical nu au cunoștințele necesare pentru a selecta simptomele de care suferă. O posibilă soluție este modificarea interfeței astfel încât să ghideze utilizatorul în alegerea simptomelor. Considerăm totuși că obținerea de ajutor din partea unui personal medical este mult superioară oricărei încercări de a face mai ușor pentru oamenii obișnuiți să selecteze simptome.

**Bibliografie**

[1] Arsene Fansi Tchango, Rishab Goel, Zhi Wen, Julien Martel, și Joumana Ghos. Ddxplus: A new dataset for automatic medical diagnosis. *Conference on Neural Information*

*Processing Systems Datasets and Benchmarks Track*, 2022. Disponibil la <https://arxiv.org/abs/2205.09148>