Detectia stresului din imagini

1st Chelea Diana-Maria
1306B

2nd Spiridon Bianca
1306B

***Abstract*—Stresul constituie un factor ce afecteaza aspectele cotidiene si influenteaza atat sanatatea mintala, cat si gandirea. Acest proiect abordeaza tehnici de machine learning, utilizand biblioteca Keras si Haar Cascade, pentru a detecta expresiile faciale din imagini, fiind capabil sa precizeze nivelul de stres al persoanelor. O alta ramura a ML folosita este Deep Learning, care consta in utilizarea de retele neuronale artificiale pentru a prelua caracteristici, adica structuri alcatuite din mai multe straturi, vizibile si ascunse, care transforma imaginea initiala. Partea de cod foloseste libraria Python facerecognition,tensorflow, OpenCV si mediul de programare Colab. Pentru inceput, vom prelua un set de imagini, iar algoritmul este implementat prin tehnici precum Haar Cascades, metoda care analizeaza variatiile pixelilor din subregiuni ale imaginii pentru a determina anumite caracteristici specifice.**

***Index Terms*—recunoastere faciala,ML, retele neuronale**

I. Introducere

Acest fenomen omniprezent in existenta umana este o realitate inevitabila si complexa care ne insoteste in fiecare aspect al vietii noastre. Este important sa intelegem ca cresterea sau intensificarea stresului poate avea un efect negativ, chiar si atunci cand este considerat adaptiv sau normal. Intelegerea si gestionarea acestuia sunt esentiale pe masura ce societatea noastra se schimba si avem ritmuri din ce in ce mai alerte si solicitante. Interesul in cercetare a atins cote considerabile, desfasurandu-se studii in domenii precum neurostiinte, psihologie si medicina. In acest scop, s-au conturat diverse solutii. Aceste solutii variaza de la strategii de adaptare personala, pana la implementarea unor tehnologii ce pot detecta stresul. Scopul acestei lucrari este de a dezvolta si a implementa un program pentru detectarea stresului in imagini, punand accent pe analiza expresiilor faciale. Prin utilizarea tehnologiilor de recunoastere faciala si al algoritmilor deep learning, intentionam sa identificam caracteristici faciale specifice, asociate starii de stres. Ne propunem sa dezvoltam un instrument precis, capabil sa ofere o evaluare obiectiva a stresului. Prin imbinarea tehnologiei cu intelegerea profunda a comportamentului uman, aceasta lucrare va ajuta in cercetarea stiintifica, dar si in implementarea unor solutii practice pentru prevenirea stresului. Proiectarea si implementarea acestui program ar include: procesul de colectare a datelor, preprocesarea fotografiilor: redimensionarea si normalizarea imaginilor pentru a garanta coerenta si eficienta analizei, implementarea algoritmului de identificare a fetei, selectarea unui model de deep learning (CNN) pentru extragerea caracteristicilor fetei, utilizarea unui model machine learning pentru a urmari coordonatele sprancenei, testarea algoritmului si analiza expresiilor faciale in contextul stresului.

II. State-of-the-art

Diversele abordari si implementari care au fost dezvoltate și aplicate in ultimii ani au imbunatatit semnificativ procesul de identificare si evaluare a stresului prezent in imagini. Ca exemplu, acestea includ:

A. Retele neuronale CNN

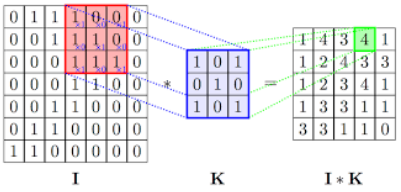
O metoda simpla de a construi o arhitectura liniara a retelei neuronale in Keras este modelul secvential (Sequential), in care straturile sunt adaugate in ordine secventiala. Modelul este adecvat pentru informatiile care sunt procesate de la stratul de intrare la stratul de iesire. Biblioteca Keras, o resursa puternica pentru analiza modelelor de deep-learning, se integreaza cu arhitectura Rețelelor Neurale Secvențiale. Pentru antrenarea retelelor, acest model folosește Theano și TensorFlow. Aceasta arhitectura foloseste straturi de convolutie, maxpooling si dropout .Retelele neuronale extrag caracteristicile fetelor din imagini pentru recunoașterea facială. Acest lucru le permite să distinga diferite persoane din imagini noi. CNN contine: input layer, hidden layers si output layer, toate conectate intre ele. Primul nivel accepta intrari in diferite forme, straturile ascunse le prelucreaza,iar layerul de iesire contine rezultatul final.

1. Convolution layer: rezultatele filtrarii imaginii initiale (peste imaginea de intrare se aplica filtre, pentru a evidentia obiecte).

Formula dupa care se calculeaza convolutia:

$$s(i,j) = (X \times W)(i,j) + b = \sum_{k-1}^{n\_in}(X_k \times W_k)(i,j) + b$$

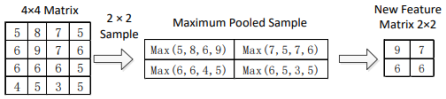
Convolutie:



2.Pooling layer: reprezinta layerul obtinut prin micsorarea imaginii de intrare, ajuta la pastrarea caracteristicilor relevante.

De exemplu, max pooling reduce dimensiunea selectand valoarea maxima dintr-o regiune specifica.

Pooling:



Scopul convolutiei si al pooling-ului este de a reduce progresiv volumul datelor si de a extrage elemente care devin tot mai abstracte in straturile ulterioare ale retelei.

3.ReLU: reprezinta layerul Rectified Linear Unit. Astfel, daca valorile din matrice sunt mai mici decat 0, clar pixelii nu sunt relevanti pentru cautare, deci ii vom ignora, atribuindu-le valoarea 0.

4.Dropout: reprezinta layerul de regularizare care este folosit pentru a preveni suprapunerea.Pentru a forta modelul sa invete reprezentari puternice, un procent de neuroni este dezactivat in timpul antrenarii .

B. The Histogram of Oriented Gradients

The Histogram of Oriented Gradients este o tehnica de extragere a caracteristicilor din imagini, utilizata pentru detectia de obiecte sau recunoastrea lor in imagini . Aceasta tehnica numara aparitiile orientarii gradientului intr-un segment specific al unei imagini. Descriptorul HOG se concentreaza pe forma sau structura unui obiect. Deoarece calculeaza caracteristicile utilizand atat magnitudinea, cat si unghiul gradientului, aceasta tehnica este mai eficienta decat orice descriptor de margini , creand histograme pentru zonele imaginii . Pasii de calculare ai histogramei sunt :

- redimensionarea imaginii la o forma standard de 128x64 pixeli
- calcularea gradientului, a magnitudinii si unghiul gradientului

  Sobel

$$G_x \rightarrow \begin{bmatrix} -1 & 0 & 1 \\ -2 & \boxed{0} & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$G_y \rightarrow \begin{bmatrix} -1 & -2 & -1 \\ 0 & \boxed{0} & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

  Prewitt

$$G_x \rightarrow \begin{bmatrix} -1 & 0 & 1 \\ -1 & \boxed{0} & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

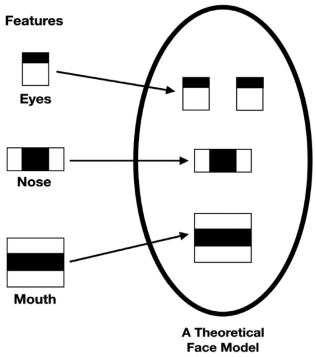
$$G_y \rightarrow \begin{bmatrix} -1 & -1 & -1 \\ 0 & \boxed{0} & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- dupa ce obtinem gradientul pentru fiecare pixel, matricile sunt impartite in blocuri si se creeaza histograme ale gradientilor
- urmeaza normalizarea in recunoasterea faciala, care este utilizata pentru a pregati cadrele si a crea un cadru uniform pentru machine learning
- formarea histogramei finale a gradientilor orientate: un vector de caracteristici ce contine informatii despre pozitia ochilor, nasului , textura pielii , etc. Acest vector este rezultatul extragerii si sumarizarii informatiilor relevante din imaginea fetei .
- ultimul pas este clasificarea si detectia, unde se foloseste un algoritm de invatare pentru probleme de clasificare si regresie .

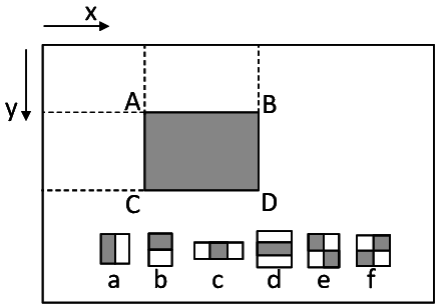
C. Haar Cascades

Detectarea obiectelor utilizand Haar feature-based cascade classifiers reprezinta o metoda machine learning unde o functie cascade este antrenata pe mai multe imagini pozitive si negative. Desi sunt antrenati sa identifice un singur tip de obiect, acesti clasificatori pot fi folositi in paralel, pentru a detecta de exemplu ochii si fata.

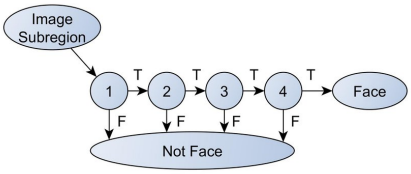
- **Algoritmul are nevoie de un set de imagini pozitive** (in care se regasesc expresii faciale) si unul de **imagini negative** (in care nu exista oameni) pentru a colecta Haar features (trasaturile). Acestea sunt extrase folosind blocuri de tipul:



  Haar features sunt calculate prin scaderea sumei intensitatilor pixelilor din zona alba din cea a pixelilor din zona neagra. Deoarece acestea sunt greu de determinat pentru o imagine de mari dimensiuni, algoritmul foloseste integral images.
- **Creare integral images**: In loc de calcularea pentru fiecare pixel in parte, se vor crea subregiuni ale imaginii pentru care vom calcula trasaturile.



- **Adaboost alege cele mai bune trasaturi**: Se va folosi o combinatie de weak classifiers (metode de clasificare nu foarte precise, dar care pot sa dea rezultate importante folosite impreuna) pentru a crea un strong classifier pe care algoritmul il poate folosi pentru a detecta obiectele. Pentru a crea acesti clasificatori slabi, algoritmul va aplica ferestre peste imagine, calculand pentru fiecare in parte caracteristicile Haar. Acestea vor fi comparate cu un prag anterior, care separa fetele de restul obiectelor.
- **Combinarea acestor clasificatori slabi intr-un clasificator puternic** se realizeaza prin intermediul cascade classifiers.
- **Implementare Cascade Classifier**: este alcatuita dintr-o serie de stari, unde fiecare nivel este o colectie de weak classifiers. Fiecare nivel decide daca obiectul este sau nu o expresie faciala – positive: s-a identificat o fata, negative:cauta mai departe in urmatorii classifiers. Este esential sa avem o rata cat mai mica de raspunsuri negative, deoarece afecteaza grav algoritmul de detectie.



D. AdaBoost

AdaBoost este un algoritm de invatare automata utilizat pentru clasificare si regresie . Acest algoritm construieste un model puternic prin combinarea unor modele de baza(weak learners). Metoda asociază o pondere fiecărui exemplu de antrenare.Iniţial toate instanţele au aceeaşi pondere de 1/n. Algoritmul se bazeaza pe ideea de a gasi clasificatorul slab cel mai bun pe baza erorii ponderate(cea mai mica valoare), apoi modificam importanta exemplelor. Exemplele care au fost clasificate gresit vor avea o pondere mai mare, ceea ce le va face mai importante pentru urmatorii clasificatori slabi, in timp ce exemplele care au fost clasificate corect vor avea o pondere mai mica. In urmatorul pas vom descoperi un alt clasificator ineficient, deoarece trebuie sa clasifice corect instantele cu pondere mai mare, avand o parte mai mare din eroarea ponderata. Scorul final este influentat de fiecare clasificator slab. Aceasta influenta este evaluata in functie de cat de bine performeaza pe setul de antrenare.

E. Adam

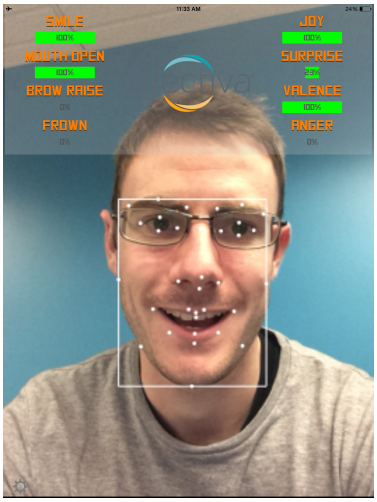
Adam(Adaptive Moment) este un algoritm de optimizare a ratei de invatare adaptat pentru retele neuronale. Acesta calculeaza ratele de invatare individuale in functie de o varietate de parametri si este o abordare adaptiva de ajustare a ratei de invatare. Adam adapteaza rata pentru fiecare ponderare a retelei neuronale folosind estimarile primului si celui de al doilea moment, ceea ce ii da numele. Primul moment este media, iar al doilea moment este varianta , aceasta informatie spunand cat de variabile sunt ponderile (conexiunile dintre neuroni). Pe baza acestor informatii, Adam modifica ratele de invatare pentru fiecare dintre aceste variabile in felul urmator : rata de invatare pentru parametrii cu varianta gradientul mediu mai mare va fi mai mica. Acest lucru incetineste schimbarile si impiedica oscilatiile nedorite. Pentru a accelera schimbarile in directia dorita, Adam va ajusta rata de invatare , incercand sa accelereze procesul de convergenta.

F. SVM

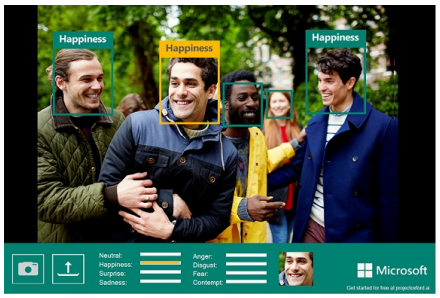
Un instrument de învățare automată eficient cunoscut sub numele de SVM (Support Vector Machine) are capacitatea de a gestiona seturi de date extrem de complexe, de recunoaștere facială, insa nu este practic in sistemele real-time. Practic, SVM cauta un hiperplan pe care il va plasa astfel incat distanta dintre 2 elemente sa fie maxima. Atunci cand introducem o noua data (o noua persoana), algoritmul va sti sa o plaseze in multimea corecta. CNN este mai precis decat SVM, fiind utilizat in prezent.

III. Solutii comerciale

- Tehnologia Affectiva poate fi integrata in aplicatii care doresc sa includa functionalitati de recunoastere a emotiilor. Prin utilizarea unei camere web, Affectiva poate identifica atat fata, cat si punctele cheie ale acesteia, pentru a clasifica expresiile faciale in sapte emotii principale(furie, dezgust, frica, bcurie, tristete si surpriza). Aplicatia este destinata agentiilor de publicitate care doresc sa testeze reactia consumatorilor la videoclipuri, reclame si emisiuni tv. Rezultatele tehnologiei Affectiva sunt extrem de promitatoare avand in vedere parteneriatele cu Coca Cola si Mars. Este folosita de peste 70 la suta dintre companiile de publicitate.



- O alta aplicatie cunoscuta care utilizeaza recunoasterea faciala pentru detectarea si recunoastere emotiilor este Microsoft Emotion API. Aceasta poate identifica emotiile umane, cum ar fi bucurie, tristete, furie, dispret si altele, folosind expresiile faciale. Dezvoltatorii pot obtine rezultate privind emotiile din fetele media prin incarcarea imaginilor sau transmiterea fluxurilor video catre API-ul Microsoft Emotion.



IV. Descrierea metodelor

Algoritmii Haar Cascades si HOG(Histogram of Oriented Gradients) reprezinta tehnici de detectare a obiectelor. Aceste metode nu se ocupa de segmentarea fetei(identificarea contururilor), ci se concentreaza mai mult pe identificarea zonelor care ar putea contine fete. Scopul principal al algoritmilor de detectare a fetei, utilizati in face recognition nu este de a identifica si de a evidentia anumite contururi, ci mai degraba de a diferentia zonele cu fete, de cele fara. Pentru implementarea codului nostru, am folosit limbajul Python, impreuna cu libraria facerecognition (din dlib, foloseste CNN), functii din OpenCV si mediul de programare Colab. Am creat un nou folder pentru a pune imaginile pe care le-am selectat de pe site-uri web pentru a incepe procesarea. Am ales multiple fotografii cu aceleasi persoane, din ipostaze diferite sau care afiseaza diverse emotii, pentru a pune in evidenta posibilitatea recunoasterii si gruparii imaginilor. La rulare, codul va incadra intr-un chenar rosu expresia faciala identificata in poza, urmand ca la final sa afiseze, in ordine, imaginile care contin aceeasi persoana.

Dintre functiile din libraria cv2 amintim:

- **cv2.imread**: citeste imaginea, primeste ca argument path-ul
- **cv2.rectangle**: pentru afisarea chenarului
- **cv2.imshow**: pentru afisarea imaginii

O alta biblioteca open-source utilizata in proiect este Keras, implementata in Python pentru antrenarea CNN (retele neurale convolutionale). Keras reprezinta o interfata pentru TensorFlow, un framework care faciliteaza implemetarea deep learning. Cu ajutorul bibliotecii Keras, vom putea aplica pe setul de imagini layerele de convolutie, maxPooling si dropout. Un layer reprezinta o transformare I/O, iar un model este alcatuit dintr-un set de layere. Din libraria Keras am folosit urmatoarele clase:

- **ImageDataGenerator**: ajuta la generarea de seturi de imagini preprocesate pentru antrenarea retelelor neurale convolutionale; in codul nostru, am redimensionat imaginile,am normalizat valorile pixelilor (intre 0 si 1) pentru a face mai rapida procesarea, am rotit si inversat; aceasta clasa modifica direct pe setul de imagini, pastrand doar varianta finala;
- **Sequential**:metoda de a construi o arhitectura liniara a retelei neuronale in Keras.Aceasta arhitectura foloseste straturi de convolutie, maxpooling si dropout
- **Conv2D**: strat ce realizeaza operatia de convolutie; pentru a extrage in mod progresiv caracteristicile care ne intereseaza, reteaua va aplica un set de filtre peste imagini;
- **MaxPooling2D**: strat ce realizeaza redimensionarea hartii de caracteristici create in urma convolutiei; in functie de valoarea dimensiunii ferestrei de pooling, acest strat pastreaza caracteristicile relevante, selectand valoarea maxima din regiunea specificata;
- **Dropout**: strat ce ajuta la reducerea overfittingului in CNN; acest lucru imbunatateste generalizarea modelului pe date noi, fortand reteaua neurala sa nu se bazeze pe interdependentele dintre anumiti neuroni, eliminandu-i periodic;
- **Flatten**: strat plasat intre straturile convolutionale si cele complet conectate, pentru a transforma datele din format 2D intr-un format liniar; astfel, le vom putea introduce in straturile complet conectate; (de exemplu, o imagine cu matrice de pixeli poate fi transformata intr-un vector unidimensional)
- **Dense**: toti neuronii din stratul curent sunt conectati cu cei din stratul anterior, formandu-se astfel stratul final; fiecare neuron din acest strat primeste intrari de la toti neuronii din cel anterior si produce o iesire; acest strat foloseste urmatoarele functii de activare:
  - **ReLu**: folosit pentru a introduce non-liniaritate, ignora pixelii care au valori negative, considerandu-I irelevanti pentru cautarea caracteristicilor;
  - **Softmax**: folosit pentru clasificarea cu mai multe clase; converteste output-ul numeric intr-o distributie de probabilitati (in cazul nostru, distributia pentru fiecare emotie intr-o imagine);

Algoritm detectare fete in imagini:

1) facerecognition.facelocations: returneaza un array care contine lista de coordonate pentru fiecare fata din lista de imagini
2) salvam coordonatele pentru fiecare fata din imagine intr-un tuplu (top,right,bottom,left); acestea vor fi folosite la trasarea conturului cu rosu pentru a evidentia zona capului

Set date de intrare: Pentru antrenarea modelului nostru am utilizat un setul de date FER-2013, disponibil pe Internet. Arhiva contine o serie de imagini (28.709 de imagini pentru antrenare si 3589 pentru testare) grayscale 48x48 in care se regasesc fete in diverse unghiuri si pozitii. Imaginile sunt impartite in 7 foldere, fiecare reprezentativ pentru emotia redata, iar distributia acestora este relativ echilibrata, cu acelasi numar de imagini pentru fiecare categorie. In cadrul proiectului am asociat starea de stres cu emotii precum nervozitate, dezgust, frica, pastrand pentru comparare sentimentele de bucurie, tristete si neutru. La fiecare rulare a codului, pe imagine se va afisa cu text in jurul fetei persoanei emotia pe care aceasta o afiseaza.

Algoritm recunoastere fete si afisarea imaginilor in ordine:

1) vom intera prin setul de date, adaugand intr-o lista knownnames toate numele imaginilor;
2) pentru a afla daca in imagini se repeta aceeasi oameni, comparam imaginile una cate una (prima cu a doua, cu a treia, samd). Vom folosi functia comparefaces, care primeste ca argumente cele 2 imagini. Aceasta functie returneaza un array de tip bool, comparand cele 2 arrayuri ale functiei facerecognition; atunci cand gaseste valori comune sau foarte asemanatoare pe aceeasi pozitie, returneaza True;
3) Daca s-au gasit 2 imagini cu aceeasi persoana, vom adauga numele acesteia intr-o lista doar o singura data; Algoritmii de recunoastere a fetei din biblioteca face recognition folosesc detectarea. In recunoasterea fetei, detectarea se refera la procesul de a determina daca o fata este prezenta intr-o imagine. Acest lucru implica gasirea zonelor care ar putea contine fete, delimitandu-le in dreptunghiuri. In ultima parte a codului, am evidentiat zonele de interes pentru algoritm, prin utilizarea functiei facelandmarks, care returneaza coordonatele pentru diferite caracteristici ale fetei (chin, left-eyebrow, nose-bridge, etc), mai apoi colorandu-le cu functia fill din matplotlib.pyplot.

Algoritmul modelului de antrenare:

1) Alegerea bibliotecilor : principala biblioteca folosita in algoritmul nostrum este Keras. Cu ajutorul bibliotecii Keras, vom putea aplica pe setul de imagini layerele de convolutie, maxPooling si dropout.
2) Un alt aspect important este definirea directoatele ce contin date de antrenare si validare a modelului . Directorul care contine datele de antrenare va fi folosit pentru a antrena modelul,iar imaginile din directorul de validare vor fi utilizate pentru a evalua performanta modelului dupa ce a fost antrenat, acestea fiind niste imagini noi pentru model.
3) Se definesc 2 obiecte folosind ImageDataGenerator . ImageDataGenerator este utilizat pentru a genera imagini noi prin aplicarea unor transformari si augmentari asupra imaginilor existente. Rotirea imaginilor cu un unghi de pana la 40 de grade permite modelului sa devina mai fiabil la variatii ale pozitiei obiectelor din imagini , rescale normalizeaza valorile pixelilor intre [0,1], shear range pentru a face modelul mai rezistent la variatii in aspectul obiectelor, iar zoom range este folosit pentru a permite modelului sa invete caracteristici in aspect de micsorare sau marire . Obiectele ImageDataGenerator se folosesc pentru a prelua imaginile din directoarele specifice. Se va specifica batch size, adica cate antrenamente vor fi procesate simultan , class mode il vom alege categorical , deoarece avem mai multe directoare impartite pe categorii , target size de 48x48, avand imgini de 48x48 in directoare , horizontal flip=True pentru a antrena modelul in simetria faciala si fill mode=nearest care umple pixelii in zonele ramase libere dupa inversarea imaginilor.
4) La emotii vom alege Angry , Disgust si Fear ca fiind parte a emotiei de stress.
5) In continuare se creeaza modelul secvential , unde datele vor trece dintr-un strat in altul.Vom avea 5 posibilitati de numere de filtre . Am ales aceste filtre deoarece indica o creste progresiva a complexitatii extragerii caracteristicilor in straturile convolutionale. Straturile profunde vor fi capabile sa detecteze caracteristici mai complexe.. Se adauga filtrele : Conv2D, MaxPooling2D si Dropout. Cu aceste filtre vom extrage caracteristicile de interes si le vom pastra, selectand valoarea maxima din regiune. De asemenea vom elimina periodic legaturile dintre neuroni , pentru ca modelul sa se poata adapta si pe alte seturi de valori si sa eliminam dependenta de setul de antrenare. Ultimele filtre care se vor adauga vor fi cele de netezire(Flatten , Dense) .Flatten va transorma datele intr-un strat unidimensional, pregatindu-le pentru straturile fully connected. Filtrul dense este un filtru fully connected care preia reprezentarile In forma liniara si le conecteaza la neuroni . Stratul softmax genereaza distributii de probabilitati pentru care niste caracteristici apartin unei clase de emotie.
6) Se configureaza modelul de antrenament, monitorizandu-se acuratetea modelului.
7) In codul nostru am folosit 100 de epochs, deci am trecut de 100 de ori prin setul de date de antrenare, incercand de fiecare data sa imbunatatim performanta algoritmului.Un epoch reprezintă o trecere completă a setului de date de antrenare printr-o rețea neurală convoluțională. Algoritmul de antrenare al rețelei neuronale utilizează întregul set de date de antrenare în timpul unui epoch pentru a determina gradientul funcției de pierdere (funcția de pierdere), pentru a ajusta ponderile rețelei și pentru a îmbunătăți performanța modelului. Acest lucru indică faptul că, într-o singură iterație de antrenare, toate imaginile din setul de antrenare sunt utilizate pentru a face parametrii rețelei mai buni, insa imaginile sunt prezentate în forme ușor diferite la fiecare iterație datorita augumentarii.
8) Parametrii modelului sunt modificați folosind funcția model.fit(). In continuare, vom prezenta fiecare parametru al functiei: - train generator: aici se afla setul de date pe care vom antrena modelul - steps per epoch: numarul total de pasi pe care modelul ii parcurge pentru a acoperi tot setul de date; - epochs: de cate ori va trece modelul prin setul de date; - validation data: utilizat pentru a evalua performantele modelului dupa fiecare epoca; - validation steps: numarul total de pasi pe care il parcurge modelul pentru a acoperi setul de validare;
9) Antrenarea modelului am realizat-o pe Colab din cauza timpului de rulare ridicat de pe Visual Studio Code. Dupa salvarea modelului pe drive, il putem descarca si utiliza in aplicatii ulterioare.

10) Urmatorul pas consta in adaugarea fisierului haarcascade.frontalface.default.xml, utilizat pentru detectarea fetelor folosind tehnica Haar Cascades. Fisierul contine date despre caracteristici ale fetei plasate frontal, cum ar fi marginile gurii, ale nasului si ale ochilor, reusind sa identifice daca aceste caracteristici alcatuiesc o fata. Astfel, se determina daca o persoana este prezenta in imagine.

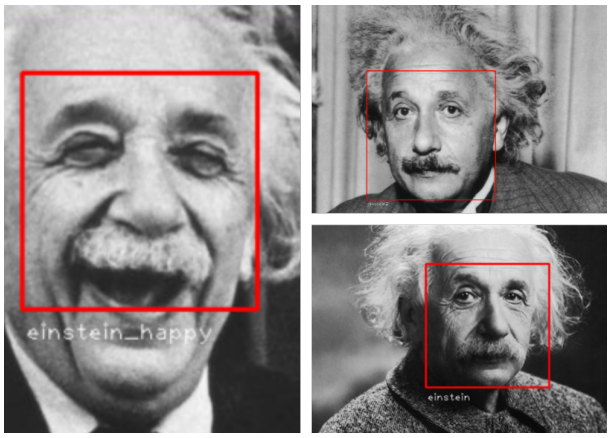
Algoritmul detectiei stresului :

1) Alegerea bibliotecilor : Cv2 : Biblioteca OpenCV, utilizată pentru manipularea imaginilor și operații de detecție a fețelor, Numpy : Bibliotecă pentru manipularea eficientă a matricilor și vectorilor, loadmodel: Funcția din Keras pentru încărcarea unui model salvat, facerecognition: Biblioteca pentru recunoașterea facială, os: Biblioteca pentru interacțiuni cu sistemul de operare.
2) Incarcarea modelul antrenat : Incarcam modelul pe care l- am antrenat anterior.
3) Configurarea detectorului de fete: utilizam clasificatorul Haar Cascade pentru a detecta fetele in imagini .Acest clasificator este utilizat pentru identificarea zonelor de interes, ce pot contine fete.
4) Declaram dictionarul de emotii, ce mapeaza indicele clasei cu eticheta corespunzatoare.
5) Vom parcurge doar fisierele jpg si vom citi imaginile pe rand.
6) Pentru colorarea cadranului in exteriorul fetei , vom volosi functia face location pentru a detecta colturile stanga sus si dreapta jos.
7) Vom preprocesa imaginile, convertindu-le in imagini grayscale , intrucat modelul a fost preantrenat pe aceste tipuri de imagini. Le vom aduce la forma de 48x48 si le vom normaliza.
8) Functia model.predict() prezice emotiile pe baza imaginii fetei prelucrate. Valorile returnate sunt rezultatele probabilitatilor pentru fiecare emotie. Apoi, codul foloseste np.argmax() pentru a gasi emotia cu cea mai mare probabilitate ca fiind emotia anticipata pentru fata respectiva si o adauga sub forma unui text deasupra chenarului rosu in imaginea afisata

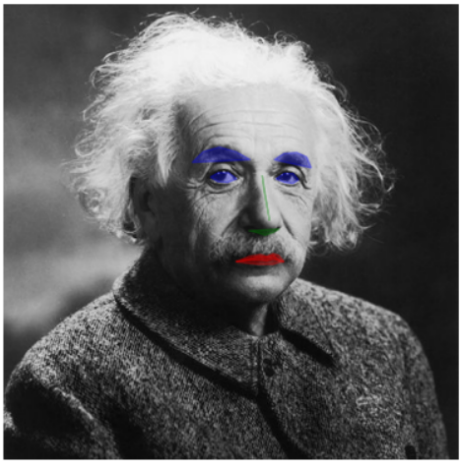
V. Rezultate preliminare

Rezultatele rularii:

Daca s-au gasit mai multe imagini cu aceeasi persoana, se vor afisa din nou doar imaginile in care apare persoana, impreuna cu numele acesteia.

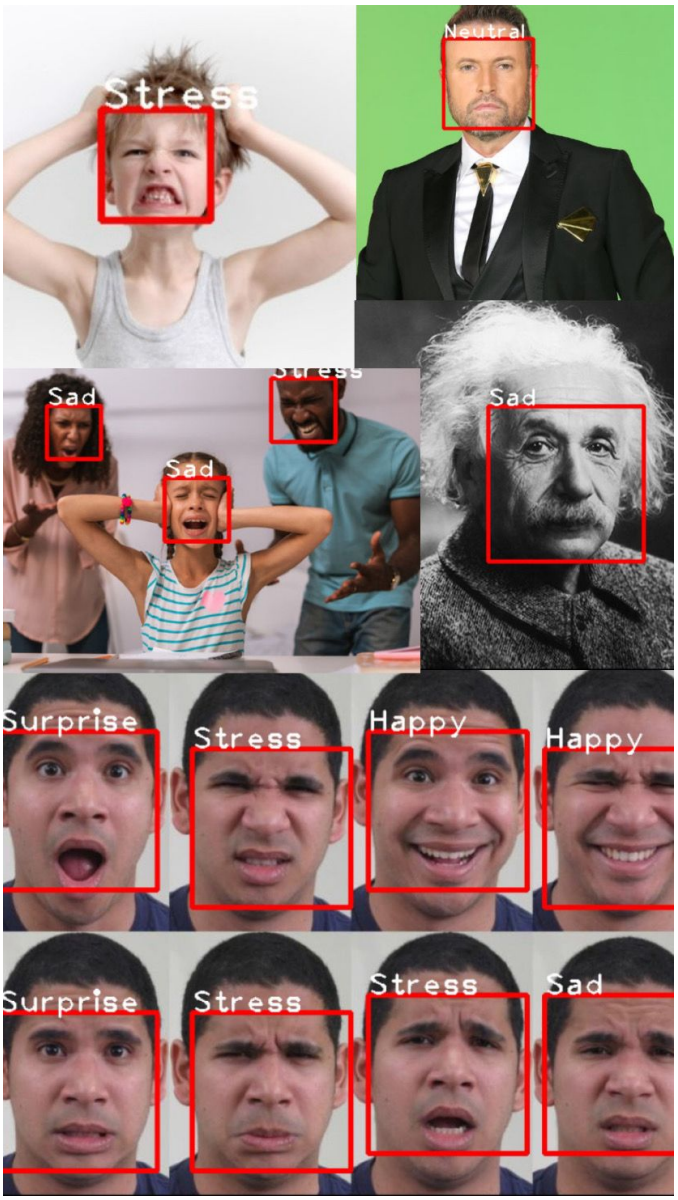


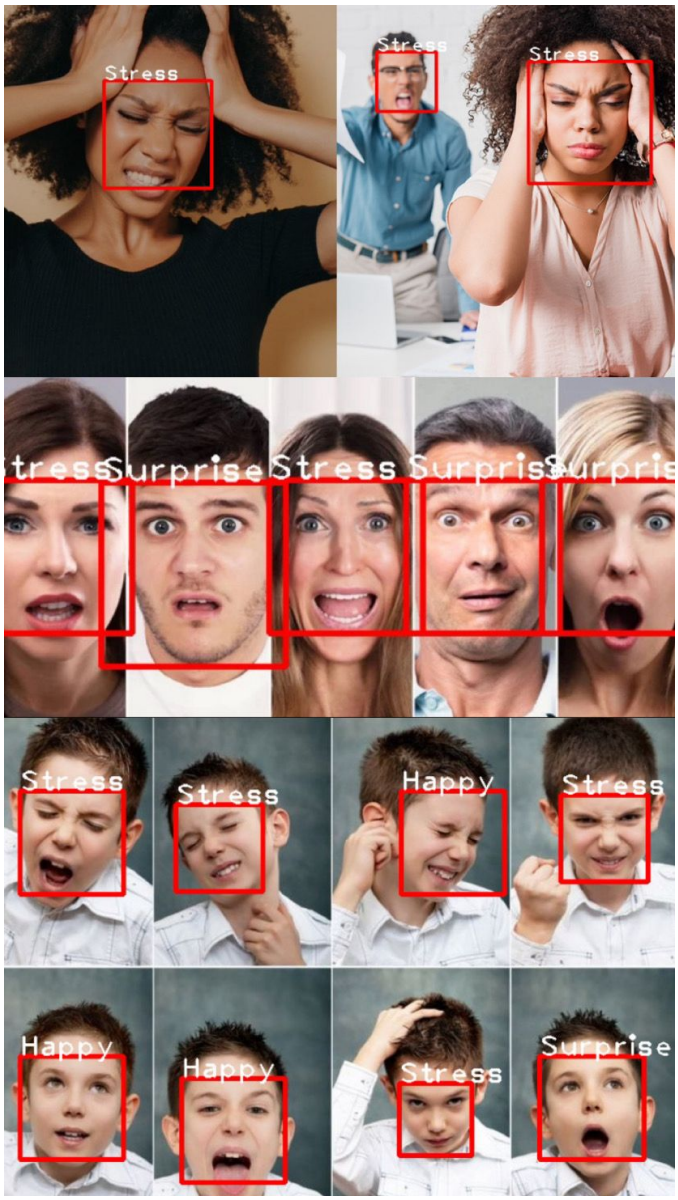
Rezultatul functiei plotfacelandmarks

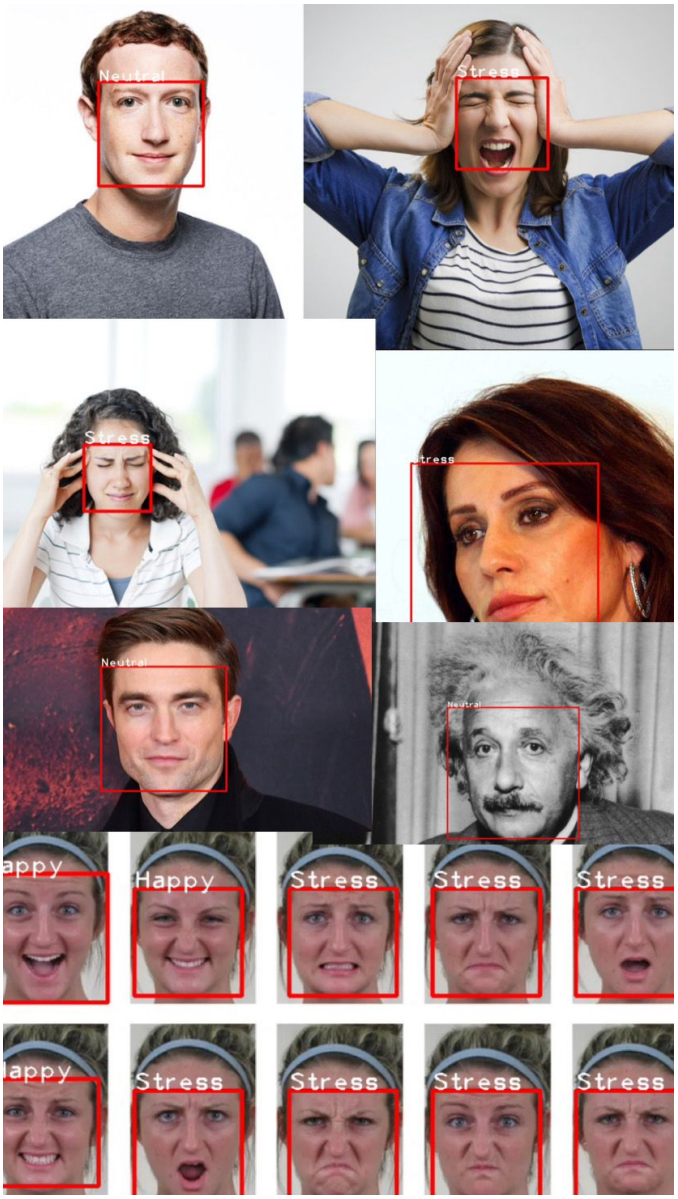


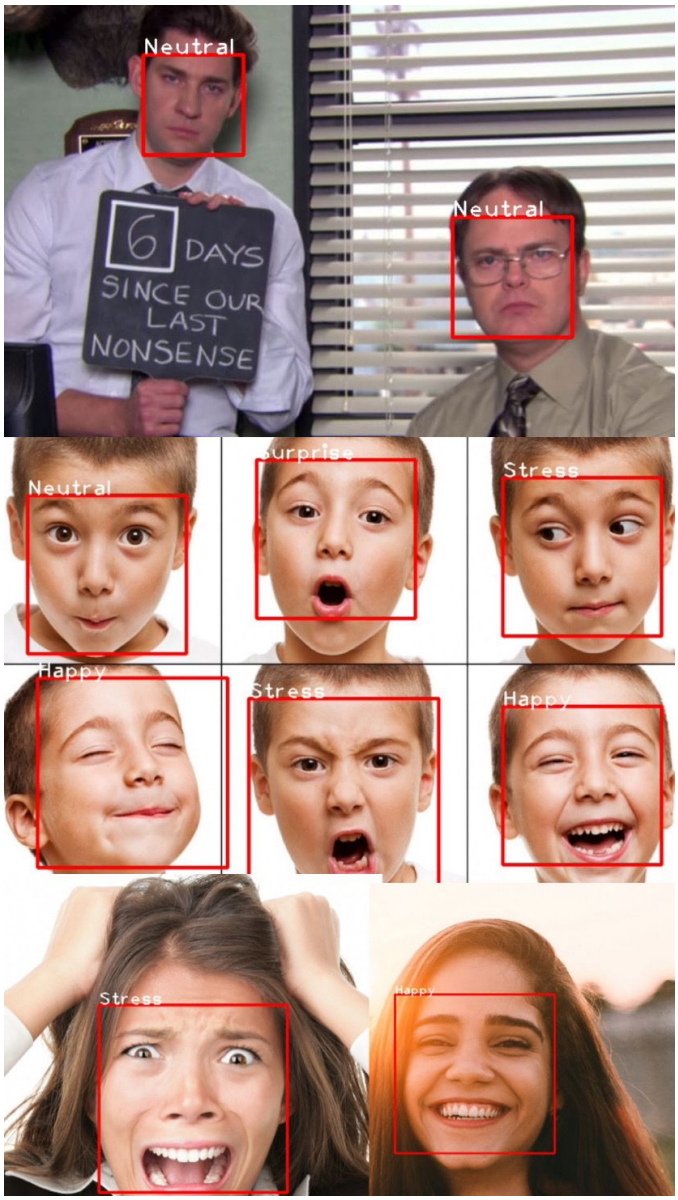
VI. Rezultate finale

Rezultatele rularii :


Stress
Neutral
Sad
Sad
Sad
Surprise
Stress
Happy
Happy
Surprise
Stress
Stress
Sad


Stress
Stress
Stress
Surprise
Stress
Stress
Happy
Stress
Happy
Happy
Stress
Surprise





VII. Concluzii

Asadar, am reusit sa identificam expresiile faciale ale persoanelor din fotografii, utilizand in principal functii din libraria facerecognition. De asemenea, algoritmul poate identifica si daca o persoana se regaseste de mai multe ori in setul de date. In a doua parte, am continuat prin a identifica emotia specifica fiecarei imagini in parte, prin utilizarea retelelor convolutionale neurale. Am identificat trasaturile principale ale fetelor cu tehnica Haar Cascade, un algoritm preantrenat sa localizeze caracteristicile esentiale. Algoritmul primeste o imagine si la iesire returneaza fata incadrata intr-un chenar rosu, cu numele emotiei scris deasupra.

Provocarile tehnice intalnite in realizarea proiectului au fost:

- alegearea unui set de date clar, cu persoane cu fata lizibila
- cautarea bibliotecilor si incercarea de a intelege concepetele de machine learning utilizate
- preprocesarea imaginilor pentru a le incadra intr-o forma standard, compatibila cu algoritmul de antrenare

- antrenarea propriu-zisa a modelului in mediul de programare Colab
- augmentarea datelor
- ajustarea parametrilor modelului, cum ar fi rata de invatare, numarul de filtre, batch size etc., deoarece influenteaza semnificativ performantele algoritmului
- evaluarea performantei modelului, referitor la alegerea unor imagini care sa reflecte o gama variata de emotii

REFERENCES

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, Kauai, HI, USA, 2001, pp. I-I, doi: 10.1109/CVPR.2001.990517.

[2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005, pp. 886-893 vol. 1, doi: 10.1109/CVPR.2005.177.

[3] S. Sriramprakash, Vadana D Prasanna, O.V. Ramana Murthy,Stress Detection in Working People,Procedia Computer Science,Volume 115,2017,Pages 359-366,ISSN 1877-0509

[4] Zhiming Xie et al 2019 J. Phys.: Conf. Ser. 1395 012006

[5] P Ramesh Naidu et al 2021 J. Phys.: Conf. Ser. 2089 012039

[6] Nijhawan, Tanya, Girija Attigeri, and T. Ananthakrishna. "Stress detection using natural language processing and machine learning over social interactions." Journal of Big Data 9.1 (2022): 1-24

[7] Lee, Hagyeong and Song, Jongwoo. (2019). Introduction to convolutional neural network using Keras. an understanding from a statistician. Communications for Statistical Applications and Methods. 26. 591-610. 10.29220 CSAM.2019.26.6.591.

[8] Krichen, Moez. (2023). Convolutional Neural Networks: A Survey. Computers. 12. 151. 10.3390/computers12080151.

[9] https://www.kaggle.com/datasets/msambare/fer2013

[10] https://github.com/opencv/opencv/blob/4.x/data/haarcascades/haarcascade frontalface default.xml