

UNIVERSITATEA „ALEXANDRU IOAN CUZA” DIN IAȘI
FACULTATEA DE INFORMATICĂ



LUCRARE DE LICENȚĂ

Detectarea cancerului de piele în imagini digitale

propusă de

Dumitrescu Mălina-Elena

Sesiunea: *Iulie, 2020*

Coordonator științific

Lect. Dr. Anca Ignat

UNIVERSITATEA „ALEXANDRU IOAN CUZA” DIN IAȘI
FACULTATEA DE INFORMATICĂ

Detectarea cancerului de piele în imagini digitale

Dumitrescu Mălina-Elena

Sesiunea: *Iulie, 2020*

Coordonator științific

Lect. Dr. Anca Ignat

Avizat,

Îndrumător Lucrare de Licență

Titlul, Numele și prenumele _____

Data _____ Semnătura _____

DECLARAȚIE privind originalitatea conținutului lucrării de licență

Subsemnatul(a) NUMITRESCU MĂLIŢA-ELENA
 domiciliul în ROMÂNIA, JUDE. VASLUI, ORAŞ VASLUI
 născut(ă) la data de 29.05.1998 identificat prin CNP 2980529374512,
 absolvent(a) al(a) Universităţii „Alexandru Ioan Cuza” din Iaşi, Facultatea de
INFORMATICA specializarea ROMÂNIA, promoţia 2020, declar
 pe propria răspundere, cunoscând consecinţele falsului în declaraţii în sensul art. 326
 din Noul Cod Penal şi dispoziţiile Legii Educaţiei Naţionale nr. 1/2011 art.143 al. 4 si 5
 referitoare la plagiat, că lucrarea de licență cu titlul:
DETECTAREA CANCERULUI DE PIELE ÎN IMAGINI
DIGITALE
 _____elaborată sub îndrumarea dl. / d-na
LECT. DR. ANCA IGNAȚ, pe care urmează să o susțin în fața
 comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată
 prin orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la
 introducerea conținutului său într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări
 științifice în vederea facilitării falsificării de către cumpărător a calității de autor al unei
 lucrări de licență, de diplomă sau de disertație și în acest sens, declar pe proprie
 răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am
 întreprins-o.

Data azi, 23.06.2020

Semnătură student 

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „DETECTAREA CANCERULUI DE PIELE ÎN IMAGINI DIGITALE”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

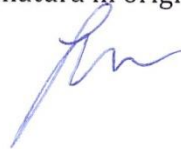
Iași, data

23.06.2020

Absolvent Prenume Nume (în clar)

DUMITRESCU MALINA-ELENA

(semnătura în original)



Cuprins

Introducere	4
Motivație	4
Gradul de noutate	6
Obiective generale ale lucrării	6
Structura lucrării	7
Contribuții	8
Capitol 1 – Descrierea problemei.....	9
Capitol 2 – Abordări anterioare	10
Capitol 3 – Aspecte teoretice și tehnologii folosite	11
Computer Vision.....	11
TensorFlow și Keras	12
Tkinter	12
Rețele neuronale convoluționale.....	13
Operațiunea de convoluție și stratul Convoluțional	13
Funcții de activare.....	15
Operațiunea de pooling și stratul Pooling	17
Stratul complet conectat (Fully Connected Layer)	18
Dropout	19
Arhitectura rețelei neuronale convoluționale VGG16	20
Transferul de învățare(Transfer Learning).....	22
Bottleneck Features.....	23
Capitol 4 - Descrierea soluției.....	25
4.1 Setul de date.....	26
4.2 Preprocesarea setului de date	27
4.3 Augmentarea setului de date de antrenare	28

4.4 Detalii despre implementare.....	30
Capitol 5 – Experimente și rezultate	32
Capitol 6- Concluzii.....	37
Bibliografie	38

Introducere

Pielea reprezintă cel mai mare organ al corpului uman, aceasta acoperind muschii, oasele și toate părțile din corp. Funcțiile pielii în corpul uman au un rol foarte important, deoarece, chiar și o mică schimbare a funcționării acesteia, poate duce la afectarea altor părți din corpul uman. Din această cauză, trebuie să acordăm o atenție sporită acesteia, leziunile pielii fiind primele semne clinice ale unor boli, precum varicela, melanomul etc.

În zilele noastre, domeniul medical se bazează foarte mult pe funcționalitățile oferite de calculatoare, pe baza acestora, reușind chiar să ofere diagnostice precise pacienților. Totodată, unul dintre obiectivele celor care se află în domeniul de cercetare medicală este depistarea din timp a cancerului de piele, fapt care duce de obicei la reducerea ratei mortalității și a tratamentului mai puțin extensiv.

Motivație

Cea mai comună formă de cancer este cea a cancerului de piele, reprezentând aproximativ o treime din toate cazurile diagnosticate cu cancer. Această boală afectează anual între două și trei milioane de oameni la nivel global. Melanomul este un tip de cancer de piele și reprezintă aproximativ 75% din decesele asociate cu cancerul de piele. Pielea umană este compusă din două straturi majore numite epidermă și derm. Stratul superior sau exterior al pielii, care se numește epidermă este compus din trei tipuri de celule, celule care asigură pielii culoarea acesteia și o protejează împotriva afectării ei. Epiderma are 5 straturi, iar în stratul profund (bazal) se formează celule care se cheeratinizează continuu și avansează spre stratul exterior (cornos) unde se exfoliază. În epiderm se găsesc melanocitele, celule care produc melanina. Sunt între 1000-2000 melanocite pe mm^2 . Melanomul este un cancer de melanocite.

Melanomul este etapizat în patru segmente, în funcție de răspândirea bolii. Etapa 0 se mai numește și melanom în situ: celulele melanomului sunt doar în stratul exterior al pielii

(epidermul); Etapele I și II înseamnă că există celule de melanom în strat direct sub epidermă sau care ating stratul următor în jos, dar nu există niciun semn că s-a răspândit la ganglioni sau alte părți ale corpului. Ultima etapă descrie melanomul care s-a răspândit prin fluxul sanguin în alte părți ale corpului, cum ar fi locații îndepărtate pe piele sau țesut moale, ganglioni limfatici îndepărtați sau alte organe precum plămânul, ficatul, creierul, oasele sau tractul gastrointestinal.

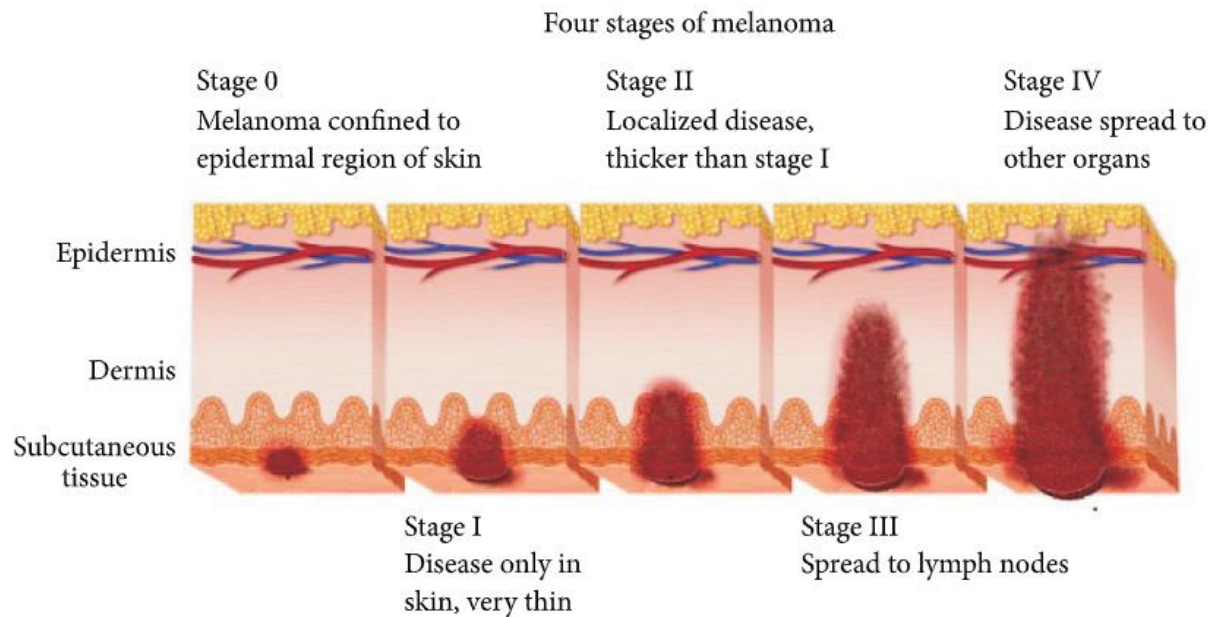
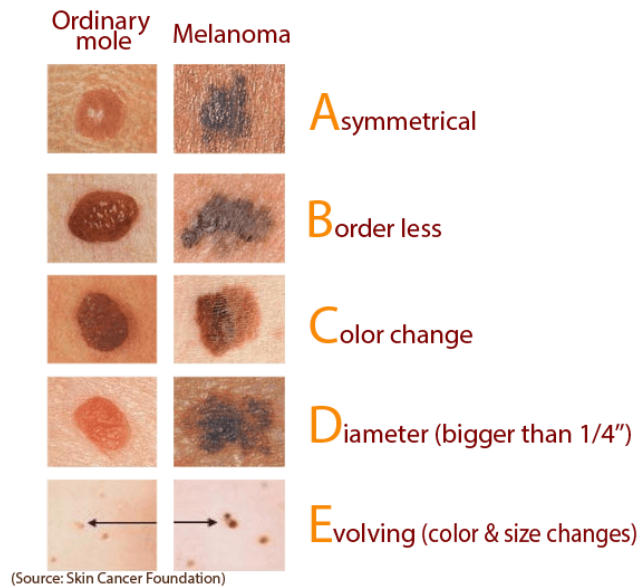


Fig 1.

Câteva dintre caracteristicile pistruiilor sau alunițelor care cresc riscul de melanom sunt: o nouă aluniță care apare după vârsta de 30 de ani, apariția unei schimbări la o aluniță deja existent, 20 sau mai multe alunițe cu diametrul mai mare de 2 milimetri, 5 sau mai multe alunițe mai mari de 5 milimetri (mai mari decât un bob de mazare). Există o regulă după care putem diferenția o aluniță sau o altfel de leziune a pielii de melanom. Această regulă se numește regula ABCDE. A înseamnă asimetrie: forma unei jumătăți a aluniței nu seamănă cu cealaltă, B vine de la margine(“border”): marginile sunt neregulate și slab definite, C vine de la culoare: culoarea nu este uniformă, leziunea având mai multe nuanțe, D înseamnă diametru: leziuni cu dimensiuni mari, (mai mult de 6 mm) sau care au o creștere semnificativă în mărime, E se referă la evoluția în timp a leziunii.

The ABCDE of Melanoma



Gradul de noutate

Subiectul lucrării, adică detectarea cancerului de piele în imagini digitale este unul de actualitate, deoarece după cum am menționat anterior, milioane de oameni se confruntă cu această problemă la nivel global, deși acest lucru nu poate fi unul îmbucurător.

Obiective generale ale lucrării

Așa cum am menționat și în Introducere și Motivație, un lucru foarte important și probabil cel mai important atunci când vine vorba de tratarea cancerului de piele este depistarea lui din timp. Este foarte important să mergem periodic să ne facem un consult general și analizele de sânge, chiar dacă nu avem simptome pentru ceva anume. Deși acest lucru este foarte important, majoritatea oamenilor nu îl fac, găsind diferite motive(lipsa timpului, cost mare, efort prea mare de depus) și merg foarte rar la un astfel de consult, iar acest lucru poate dăuna chiar propriei persoane, detectând anumite probleme când este deja prea târziu. Consider că, în cazul problemei studiate, cu ajutorul unei astfel de instrument, oamenii pot detecta din timp un

eventual cancer al pielii, efortul de a-și testa o aluniță de exemplu, fiind considerabil mai mic decât cel de a merge la doctor, deși acest lucru nu este indicat.

Structura lucrării

În Capitolul 1 este prezentată descrierea problemei. În cel de-al doilea capitol prezentăm câteva abordări anterioare care au fost găsite în timpul documentării asupra subiectului lucrării de licență. În cel de-al treilea capitol prezentăm aspectele teoretice și tehnologiile folosite în crearea aplicației. În Capitolul 4 detaliem descrierea soluției. Următorul capitol, cel cu numărul 5, prezintă niște experimente și rezultate pe care le-am efectuat pe parcursul dezvoltării aplicației, fiind nevoie de mai multe teste și experimentări, pentru a găsi soluția optimă pentru problema noastră. În ultimul capitol prezentăm concluziile lucrării, în care includem și eventuale îmbunătățiri ulterioare care pot fi aduse aplicației.

Contribuții

Pentru realizarea acestui proiect de licență, a fost necesar, în primul rând, de o documentare minuțioasă, pentru a aborda tema în cel mai corect mod. Domeniul pe care se bazează această lucrare, și anume, procesarea imaginilor digitale, nu este unul studiat la facultate, și de aceea a fost acordat mai mult documentației decât dacă lucrarea ar fi fost realizată pe un domeniu studiat complet în cadrul facultății. Desigur, au fost întâlnite și noțiuni studiate la facultate, și în același timp, decizia de a alege pentru lucrarea de licență un subiect care nu se studiază în întregime la facultate trebuie asumată de către student.

Urmatorul pas a fost alegerea modului în care se va face clasificarea unei imagini, și anume dacă se va recurge la o procesare de imagine cu ajutorul regulei ABCD, adică analizarea asimetriei, contului, culorii și diametrului imaginii date spre clasificare, sau la o clasificare utilizând învățarea profundă. După studierea amănunțită a ambelor tehnici, s-a ales varianta învățării profunde, decizie luată datorită gradului de noutate a domeniului, dorința de a învăța mai multe lucruri despre acest domeniu și în același timp, siguranța (dobândită prin documentare) că se vor obține rezultate mai bune pentru țelul propus, acela de a identifica dacă o leziune a pielii, este sau nu, malignă sau benignă.

Ulterior, a fost nevoie de o documentare asupra ceea ce presupun rețelele neuroale convoluționale, arhitecturi specifice acestora, ce arhitectură trebuie aleasă pentru ceea ce ne propunem să realizăm și nu în ultimul rând, detalii privind implementarea acestora.

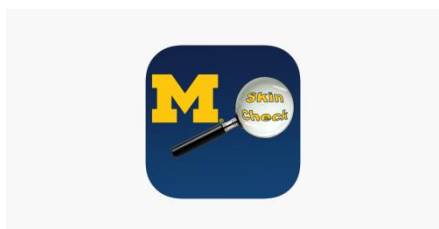
Capitol 1 – Descrierea problemei

Tema acestei lucrări de licență a pornit în primul rând de la pasiunea pe care o am pentru îngrijirea pielii. Sunt de parere că este unul dintre cele mai importantante organe ale pielii și trebuie sa îi acordăm o atenție sporită, mai ales că este organul care acoperă celelalte organe și are un rol important pentru aspectul nostru fizic. Cred că de când eram în anul I, mă gândeam că licența mea sigur va fi pe acest domeniu. Inițial, ideea era mai generală, adică detectarea mai multor probleme ale pielii, dar m-am gândit ca ar fi mai bine să merg pe ceva mai concret, care să nu fie atât de general. Începusem deja documentarea pentru subiectul pe care deja l-am menționat când, făcând parte dintr-un grup de pe Facebook în care se discută despre îngrijirea pielii, era în vogă subiectul despre expunerea la soare si efectele sale. Astfel, am aflat că dacă ne expunem prea mult la soare și nu folosim o protecție corespunzătoare, riscăm să dezvoltăm un melanom. Având foarte multe alunițe pe corp, am început să ma documentez mai mult și așa m-am gândit că ar fi o temă buna pentru lucrarea mea de licență.

Capitol 2 – Abordări anterioare



SkinVision este o aplicație care promite o acuratețe de peste 95% pentru detectarea cancerului de piele și care spune că este mult mai precisă atunci când vine vorba de această problemă chiar decât un dermatolog. Aplicație este una mobilă, pentru IOS și Android și are peste un milion de utilizatori din peste 50 de țări. Partea care poate constitui un inconvenient a acestei aplicații este că nu este gratuită. Ceea ce este interesant este faptul că a fost fondată la București, în 2011, dar ulterior a fost mutată în Amsterdam. Aplicația este capabilă să detecteze melanomul, carcinomul cu celule scuamoase, carcinomul cu celule bazale, precum și keratoza actinică precanceroasă. SkinVision folosește algoritmul „camera”. Această tehnologie calculează dimensiunea fractală a leziunilor pielii și țesutului înconjurător și construiește o hartă structurală care dezvăluie tipare anormale de creștere.



UMSkinCheck este felul ca cea precedentă, sau aplicarea pentru detectarea cancerului de piele. Este important pentru dezvoltarea universității din Michigan. Este o aplicație mobilă, disponibilă pentru Android și iOS. Aplicația oferă utilizatorilor posibilitatea de a-și monitoriza alunițele de corp, pentru a detecta eventuale modificări ale formei, culorii, asimetriei acesteia. Pe lângă asta, aplicația oferă posibilitatea calculării riscului de a face cancer de piele și de asemenea oferă informații utile și modalități de evitare a acestei boli.

Capitol 3 – Aspecte teoretice și tehnologii folosite

Computer Vision

Una dintre cele mai puternice arii ale Inteligenței Artificiale este reprezentată de Computer Vision. Computer Vision este domeniul informaticii care se concentrează pe replicarea părților din complexitatea sistemului de viziune umană și care permite calculatoarelor să identifice și să proceseze obiecte în imagini și videoclipuri în același mod în care oamenii o fac. Datorită progreselor în domeniul inteligenței artificiale și inovațiilor în învățarea profundă și a rețelelor neuronale, domeniul a reușit să facă mari salturi în ultimii ani și a reușit să depășească oamenii în unele sarcini legate de detectarea și etichetarea obiectelor.

Una dintre cele mai populare aplicabilități ale Computer Vision implică recunoașterea lucrurilor din imagini:

- Clasificarea obiectelor: Din ce categorie de obiecte face parte aceasta imagine?
- Identificarea obiectului: Ce tip de obiect este în această fotografie?
- Verificarea obiectului: Este obiectul în fotografie?
- Detectarea obiectelor: Unde sunt obiectele din fotografie?
- Detectarea reperului obiectului: Care sunt punctele cheie pentru obiectul din fotografie?
- Segmentare obiect: Ce pixeli aparțin obiectului din imagine?
- Recunoașterea obiectelor: Ce obiecte sunt în această fotografie și unde sunt?

TensorFlow si Keras



TensorFlow este al doilea framework pentru învățarea automată creat de Google. Este utilizat pentru a proiecta, construi și antrena modele de învățare profundă.

Keras este o librărie open-source de rețele neuronale scrisă în Python. Aceasta se folosește de serviciile oferite de TensorFlow. Este concepută pentru a simplifica experiența developerului cu rețelele neuronale.

Tkinter

Tkinter este o bibliotecă GUI Python open-source, cunoscută pentru simplitate și flexibilitate. Tkinter este orientată mai mult pe desktop decât pe cartea mobilă, iar din această cauză mi s-a părut potrivită pentru acest proiect.

Rețele neuronale convoluționale

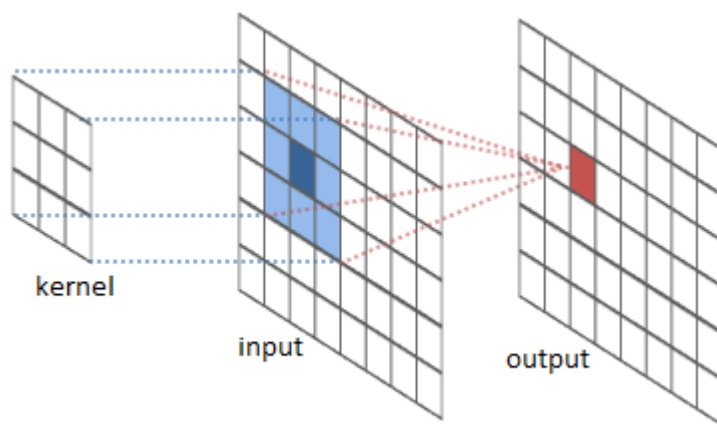
Rețelele neuronale convoluționale reprezintă un tip specializat de model de rețele neuronale proiectate pentru a lucra cu imagini în format bidimensional, deși sunt utilizate și pentru lucrul cu date unidimensionale, sau chiar tridimensionale.

O rețea convoluțională se bazează în special pe stratul convoluțional, de unde vine și numele acestui tip de rețea neuronală. Acest strat realizează operația de „convoluție”.

Operațiunea de convoluție și stratul Convoluțional

Convoluția este o operație matematică simplă, care este fundamentală pentru procesarea de imagini. Convoluția oferă o modalitate de a „multiplica împreună” două tablouri de numere, în general de dimensiuni diferite, dar de aceeași dimensionalitate, pentru a produce un al treilea tablou de numere de aceeași dimensionalitate. Acest lucru poate fi utilizat în procesarea imaginii pentru a implementa operatori ale căror valori ale pixelilor de ieșire sunt combinații liniare simple ale anumitor valori ale pixelilor de intrare.

Convoluția bidimensională este o operație destul de simplă la început: se începe cu un nucleu(engl; kernel) , care este pur și simplu o mică matrice de greutate. Acest kernel „glisează” peste datele de intrare bidimensionale, efectuând înmulțirile aferente cu partea de intrare pe care este în prezent, apoi însumând rezultatele într-un singur pixel de ieșire. Kernel-ul repetă acest proces pentru fiecare locație pe care glisează, transformând o matrice bidimensională de caracteristici într-o altă matrice bidimensională de caracteristici.



Țelul pe care îl are operația de colvoluționare este acela de a extrage din datele de intrare anumite caracteristici. În mod convențional, primul strat convoluțional este responsabil de extragerea caracteristicilor de nivel inferior, precum culoare, marginile, orientarea gradientului. Pe măsură ce adăugăm tot mai multe astfel de straturi, arhitectura se adaptează și la extragerea unor caracteristici de nivel superior, oferindu-ne o rețea capabilă să înțeleagă toate caracteristicile pe care developer-ul le dorește a fi percepute de aceasta.

Înainte de a merge mai departe cu detalierea celorlalte straturi dintr-o rețea convoluțională, este necesar să vorbim puțin despre două operații des întâlnite și utilizate în straturile convoluționale: Padding si Strides.

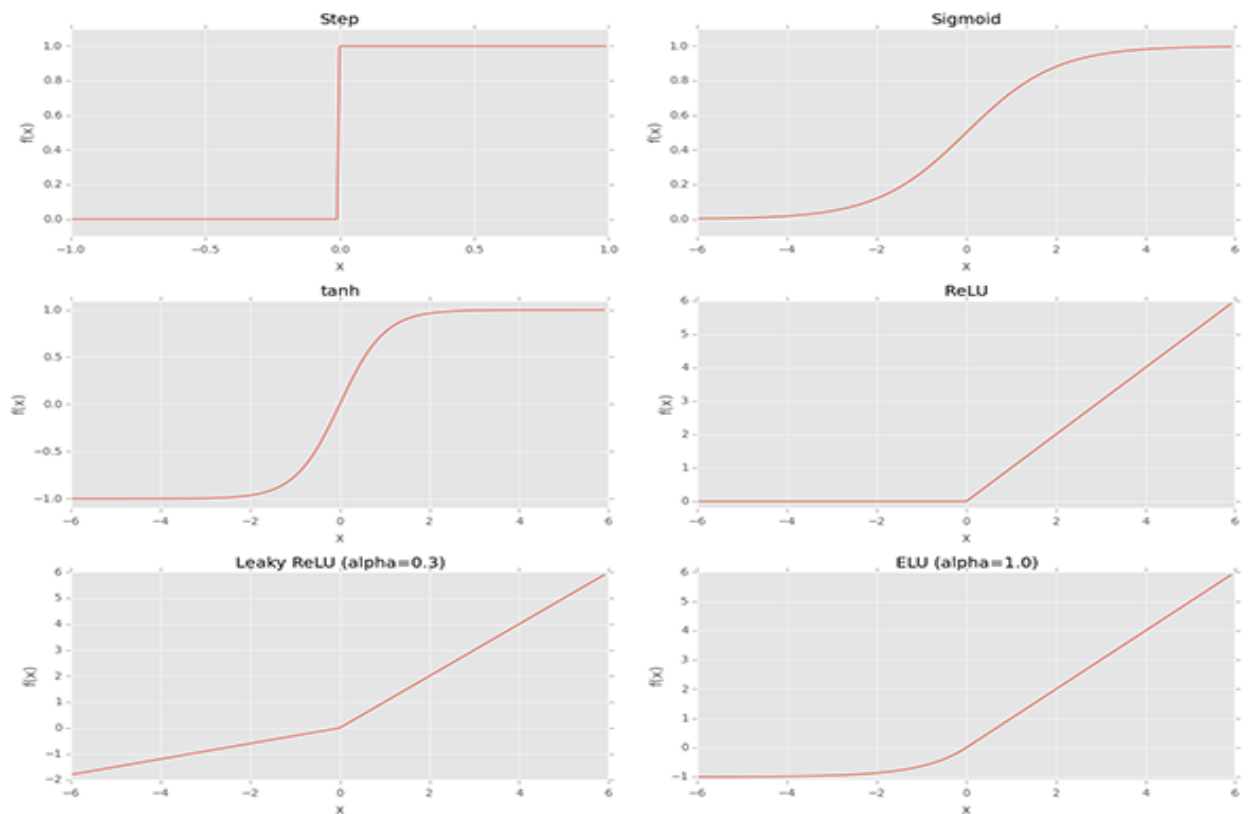
Uneori, se întâmplă ca prin glisarea kernel-ului peste imaginea de intrare, aceasta să nu reușească să parcurgă întreaga imagine, fapt pe care nu ni-l dorim, deoarece vrem ca dimensiunea inputului să fie aceeași cu a outputului. De exemplu, dacă avem o imagine cu o dimensiune de 64x64 și un kernel cu dimensiunea 5x5, atunci când se va ajunge la pixelii (61,62,63,64), kernelul nu va putea glisa, pentru că nu există spațiu suficient. În acest caz, vom adăuga niște pixeli „falși” care de cele mai multe ori vor avea valoarea 0. Prin această operațiune, acum imaginea noastră va avea dimensiunea (66x66) iar peste cei patru pixeli „descoperiți” anterior, se va putea realiza glisarea de către kernel.

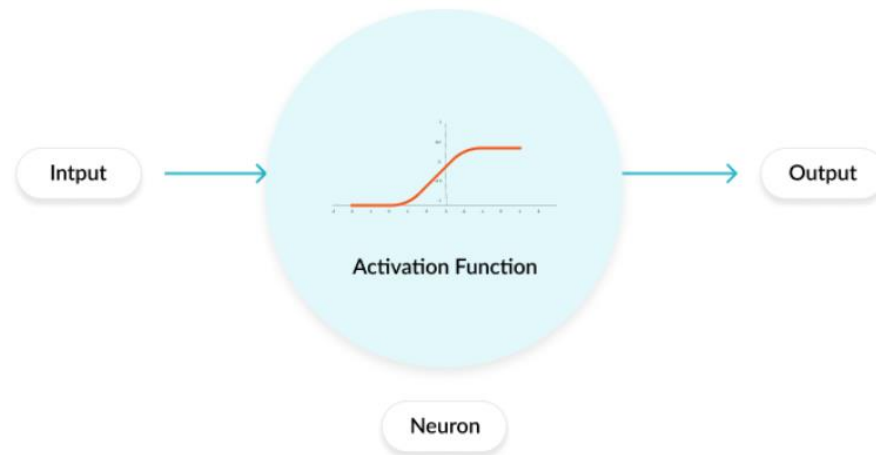
Adesea, atunci când executăm un strat de convoluție, dorim o ieșire cu o dimensiune mai mică decât intrarea. Acest lucru este obișnuit în rețelele neuronale convoluționale, unde dimensiunile spațiale sunt reduse la creșterea numărului de canale. O modalitate de a realiza acest lucru este folosirea unui strat de tip pooling. Un alt mod de a face este să folosim

stride. Ideea acestui stride este să sărim peste unele locații din glisările kernel-ului. Atunci când avem un stride cu valoarea 1, înseamnă că kernel-ul glisează peste toate combinațiile de pixeli consecutivi care se încadrează în dimensiunea kernel-ului. Dacă avem un stride cu valoare 2, atunci înseamnă că se vor sări 2 pixeli de fiecare dată când kernel-ul glisează.

Funcții de activare

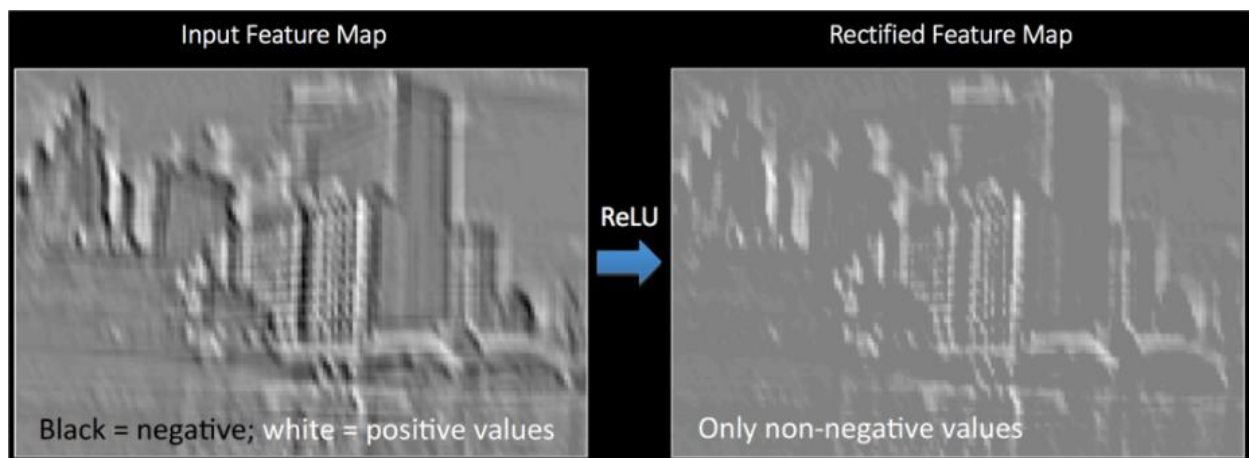
Funcțiile de activare se află în interiorul neuronilor, dar nu toți neuronii au această funcție de activare. Neuronii din straturile ascunse și de ieșire au funcții de activare, dar neuronii din stratul de intrare, nu. Funcțiile de activare efectuează o transformare asupra intrării primite, pentru a menține valorile într-un interval pe care îl putem controla. Deoarece valorile straturilor de intrare sunt în general 0 sau au valori foarte apropiate de 0, și au fost deja scalate în mod corespunzător, acestea nu necesită transformare. Cu toate acestea, aceste valori, odată înmulțite cu greutatea și însumate, depășesc rapid intervalul scalării lor inițiale, iar atunci funcțiile de activare intră în joc, forțând valorile să se încadreze într-un anumit interval și făcându-le utile.





Activarea ReLU (Rectified Linear Unit)

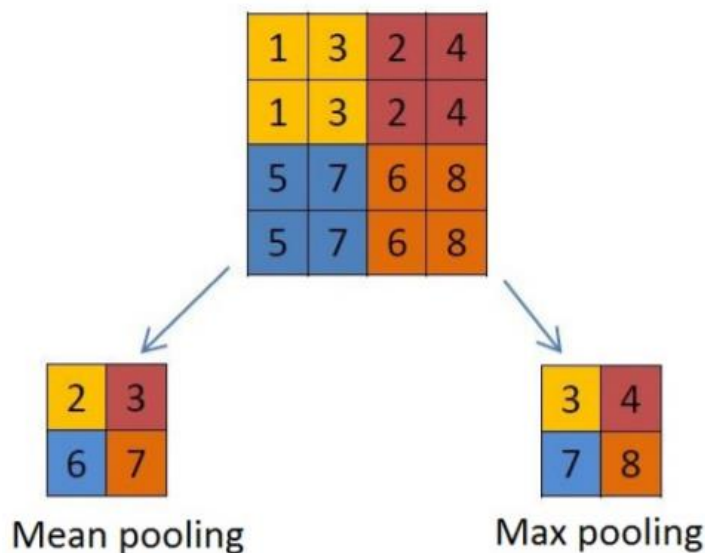
- Are forma $f(x) = \max(0, x)$
- Transformarea duce la faptul că valorile pozitive iau valoarea 1, iar cele negative, valoarea 0
- Calculele necesare nu necesita la fel de multa putere de calcul ca in cazul altor functii.
- Nu activează toți neuronii în același timp, făcând rețeaua eficientă.



Operațiunea de pooling și stratul Pooling

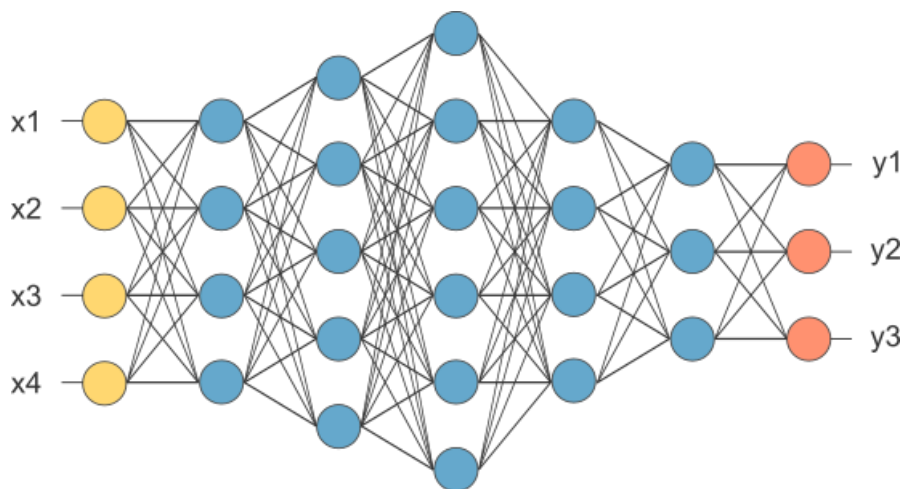
Operațiunea de pooling se folosește pentru a reduce dimensiunea caracteristicilor într-o rețea convoluțională, ceea ce comprimă ieșirile unui strat convoluțional. Deoarece atunci când construim o rețea convoluțională și ulterior, cu ajutorul acesteia ne antrenăm modelul creat, ajungem să obținem niște valori ale caracteristicilor foarte mari în urma stratului convoluțional, care produce un cost mare din punct de vedere computațional. Asadar, prin operațiunea de pooling, se poate calcula media sau maximum unei caracteristici dintr-o regiune a imaginii și se poate folosi aceasta, în locul tuturor valorilor.

Așadar, rezultă niște caracteristici comprimate, acestea reducând dimensiunea datelor și îmbunătățind performanțele clasificatorului. Stratul Pooling poate fi de mai multe tipuri: MaxPooling(pentru cazul în care se ia valoarea maximă), AveragePooling(pentru cazul în care se ia media), MinPooling(pentru cazul în care se ia valoarea minimă).



Stratul complet conectat (Fully Connected Layer)

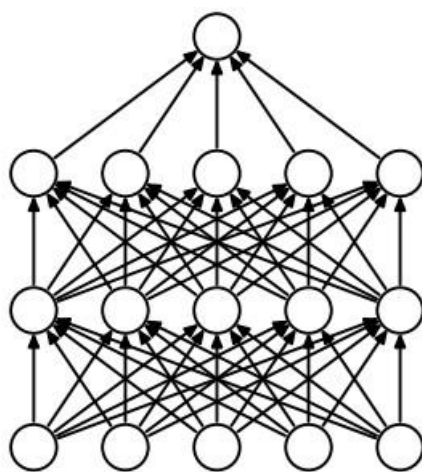
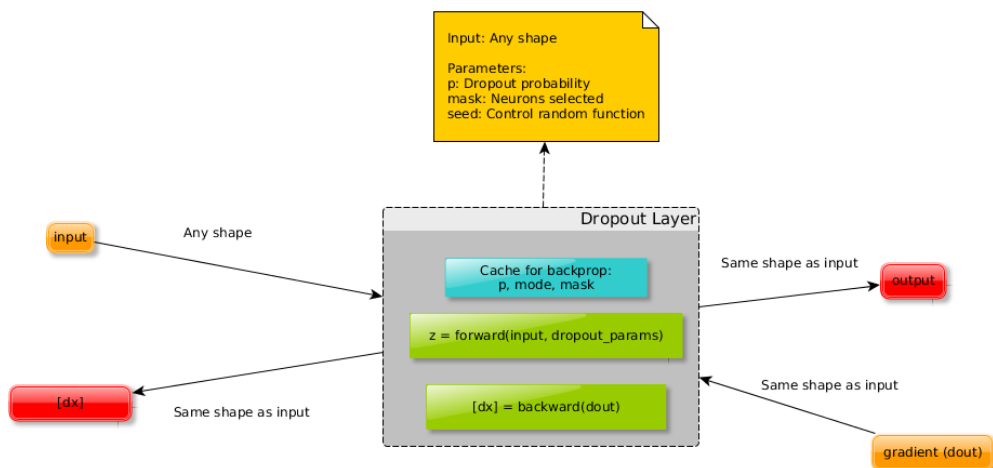
Stratul complet conectat este plasat înaintea output-ului de clasificare dintr-o rețea neuronală convoluțională și este utilizat pentru a aplatiza (flatten) rezultatele înainte de clasificare. Rezultatul operației de convoluție / colectării este aplatizat într-un singur vector de valori, fiecare reprezentând o probabilitate ca o anumită caracteristică să aparțină unei etichete. De exemplu, dacă imaginea este a unei alunițe, caracteristicile reprezentând lucruri precum o colorație bogată sau o formă asimetrică, ar trebui să aibă mari probabilități pentru eticheta „malignă”. În afară de clasificare, adăugarea unui strat complet conectat este și o modalitate (de obicei) ieftină de a învăța combinații neliniare ale acestor caracteristici. Cele mai multe caracteristici de la straturile convoluționale și de pooling pot fi bune pentru clasificare, dar combinațiile acestor caracteristici pot fi și mai bune.



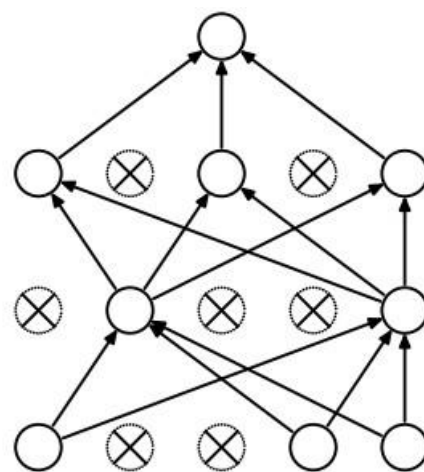
În diagrama de mai sus, matricea hărții caracteristicilor(engl. Features map) va fi convertită în vectorul $(x1, x2, x3, \dots)$. Cu straturile complet conectate, am combinat aceste funcții pentru a crea un model. În sfârșit, avem o funcție de activare, cum ar fi softmax sau sigmoid, pentru a clasifica ieșirile în funcție de etichetele pe care le avem la dispoziție.

Dropout

Dropout-ul este o tehnică de regularizare a modelelor de rețele neuronale. Practic, în timpul antrenamentului, un anumit procent (în funcție de parametru) din neuronii de pe un anumit strat vor fi dezactivați. Acest lucru îmbunătățește generalizarea, deoarece stratul va fi „forțat” să învețe cu diferiți neuroni același „concept”. În faza de predicție, dropout-ul este dezactivat. În mod normal, dropout-ul este utilizat pe straturile complet conectate, dar există posibilitatea ca acesta să fie folosit și pe straturile max-pooling, de această dată rolul fiind de a augmenta zgomotul din imagine. Efectul acestei tehnici este că rețeaua contruită devine mai puțin sensibilă la anumite greutăți ale neuronilor, astfel rezultând o rețea care este în măsură să generalizeze mai bine și de asemenea este mai puțin probabil să apară fenomenul de overfitting.



(a) Standard Neural Net



(b) After applying dropout.

Arhitectura rețelei neuronale convoluționale VGG16

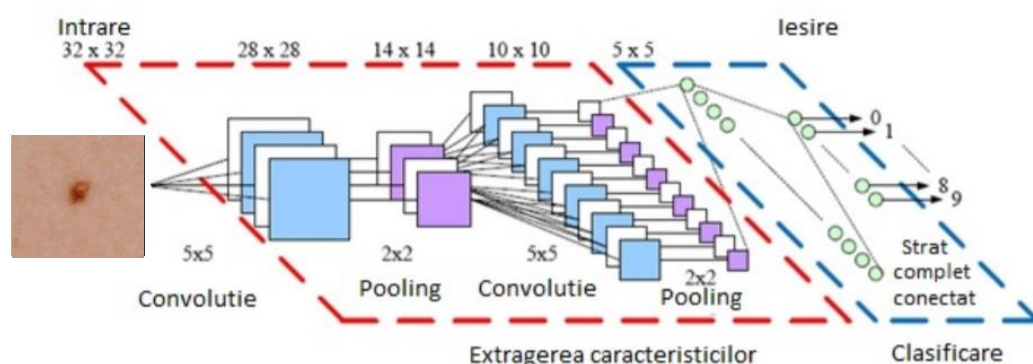
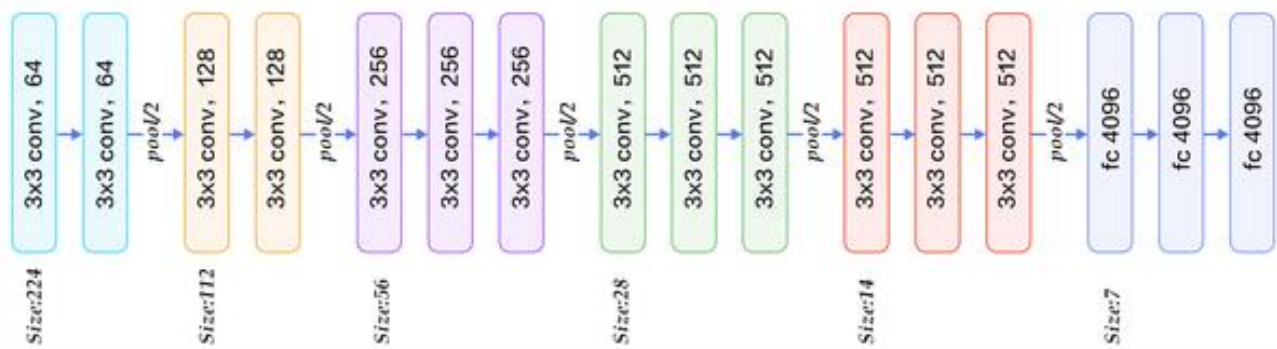


Fig. 8.3 Arhitectura rețelei neuronale convoluționale.

VGG16 este un model de rețea neuronală convoluțională propus de K. Simonyan și A. Zisserman de la Universitatea din Oxford în lucrarea “Very Deep Convolutional Networks for Large-Scale Image Recognition”. Îmbunătățirile aduse de acest model față de modelul anterior (AlexNet) sunt realizate prin înlocuirea filtrelor din kernel de dimensiuni mari (11x11 în primul strat convoluțional și 5x5 în al doilea strat convoluțional) cu mai multe filtre de dimensiuni mici (3x3) consecutive. VGG16 conține un număr de 16 straturi convoluționale (de unde și o parte din denumirea acestuia) și este preferat de foarte mulți dezvoltatori datorită arhitecturii sale uniforme.

Inputul primului strat convoluțional este fixat ca fiind o imagine RGB cu dimensiunea 224x224. Ulterior, imaginea este trecută printr-un șir de straturi convoluționale, cu o dimensiune a filtrelor de 3x3. Stride-ul pentru convoluție are valoarea 1, iar padding-ul este astfel încât rezoluția spațială să fie păstrată după operația de convoluție. Pooling-ul spațial este realizat de cinci straturi de max-pooling, care urmează o parte din straturile convoluționale (nu toate straturile convoluționale sunt urmate de max-pooling). Max-pooling-ul se realizează pe o

fereastră de 2×2 pixeli, cu stride cu valoarea 2. Straturile convoluționale sunt urmate de trei straturi complet conectate. Primele două straturi conțin fiecare câte 4096 canale. Toate straturile ascunse sunt echipate cu non-liniaritate de rectificare (ReLU).



Straturile VGG16:

1. Strat convoluțional cu 64 filtre
2. .Strat convoluțional cu 64 filtre + max-pooling
3. Strat convoluțional cu 128 filtre
- 4 Strat convoluțional cu 128 filtre+ max-pooling
5. Strat convoluțional cu 256 filtre
6. Strat convoluțional cu 256 filtre
7. Strat convoluțional cu 256 filtre + max-pooling
- 8 Strat convoluțional cu 512 filtre
9. Strat convoluțional cu 512 filtre
10. Strat convoluțional cu 512 filtre + max-pooling
11. Strat convoluțional cu 512 filtre
12. .Strat convoluțional cu 512 filtre
13. Strat convoluțional cu 512 filtre + max-pooling
14. Strat complet conectat cu 4096 noduri
15. Strat complet conectat cu 4096 noduri
16. Strat de ieșire cu activare Softmax cu 1000 de noduri.

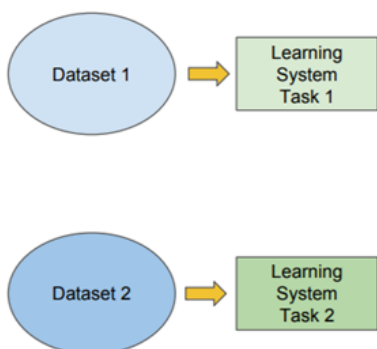
Transferul de învățare(Transfer Learning)

Oamenii au o capacitate esențială de a transfera cunoștințe între lucruri pe care aceștia le au de făcut. Ceea ce dobândim ca și cunoștințe în timp ce învățăm despre un anumit lucru, folosim în același mod pentru a rezolva alte lucruri noi. Cu cât sarcinile sunt legate mai mult, cu atât este mai ușor pentru noi să transferăm sau să ne valorificăm cunoștințele. De exemplu, dacă cunoaștem matematică și statistică, ne va fi mai ușor să învățăm lucruri despre învățarea automată. În exemplul anterior, nu învățăm totul de la zero atunci când încercăm să învățăm aspecte sau subiecte noi. Ne transferăm și ne valorificăm cunoștințele din ceea ce am învățat în trecut. Transferul de învățare depășește paradigma de învățare izolată pe care algoritmi convenționali de învățare automată o au, și utilizează acest principiu, acela de a folosi cunoștințe dobândite deja, pentru a dobândi altele noi într-un mod mult mai simplu.

În contextul nostru, adică cel al viziunii computerizate și a învățării profunde, vom folosi modele care sunt deja antrenate. Există deja multe arhitecturi deja antrenate disponibile în librăria Keras. Setul de date ImageNet este foarte întâlnit atunci când vine vorba de contruirea unui model generalizat, deoarece conține aproximativ 14 milioane de imagini la momentul actual. Cu ajutorul acestui set de imagini, putem clasifica aproximativ 1000 de categorii de obiecte

Traditional ML

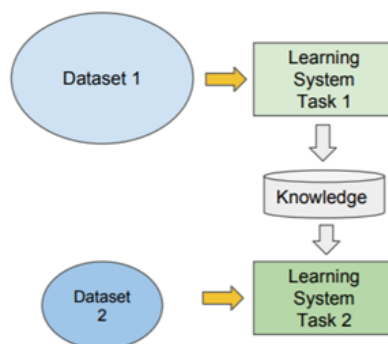
- Isolated, single task learning:
 - Knowledge is not retained or accumulated. Learning is performed w.o. considering past learned knowledge in other tasks



vs

Transfer Learning

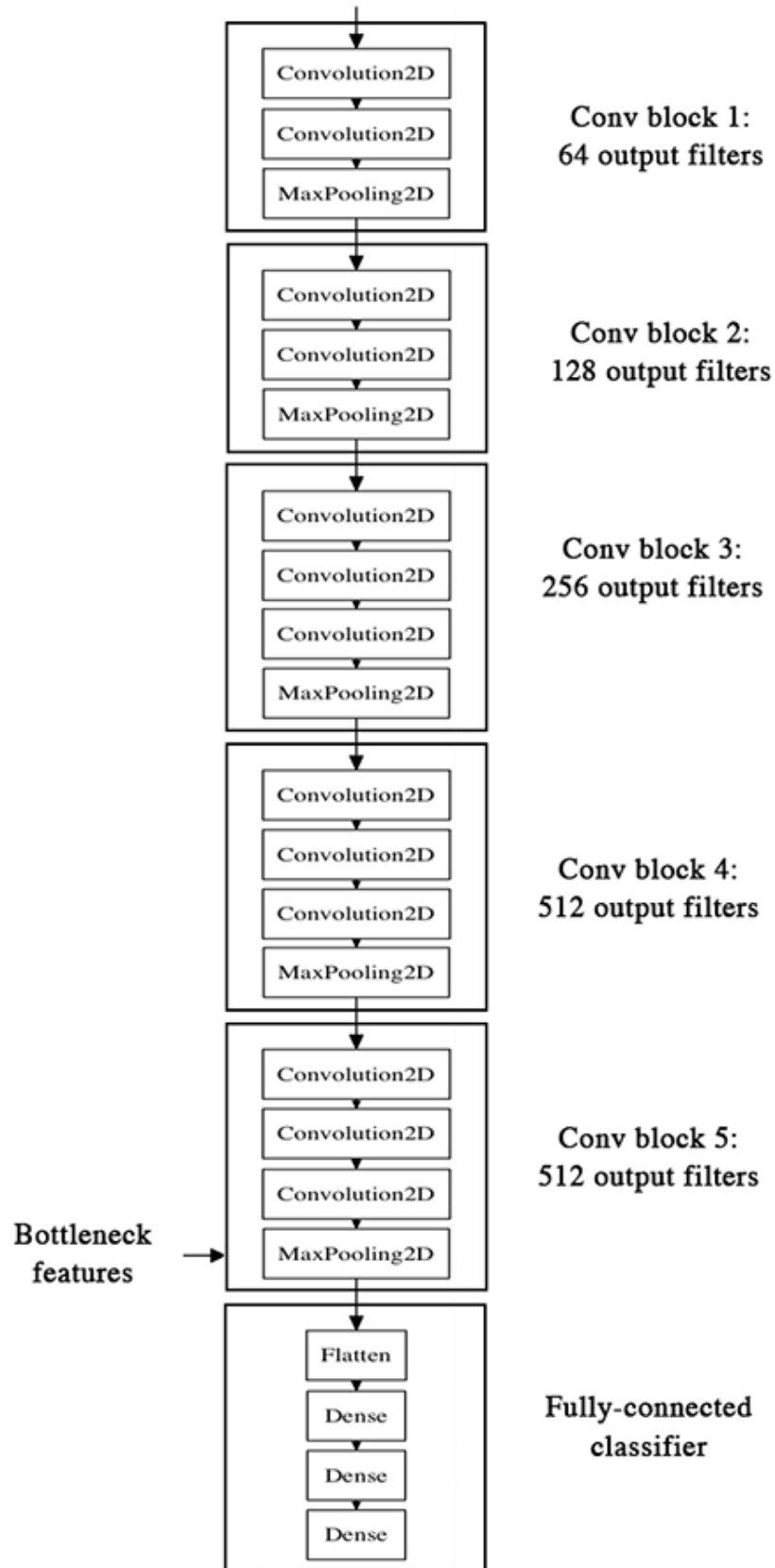
- Learning of a new tasks relies on the previous learned tasks:
 - Learning process can be faster, more accurate and/or need less training data



Bottleneck Features

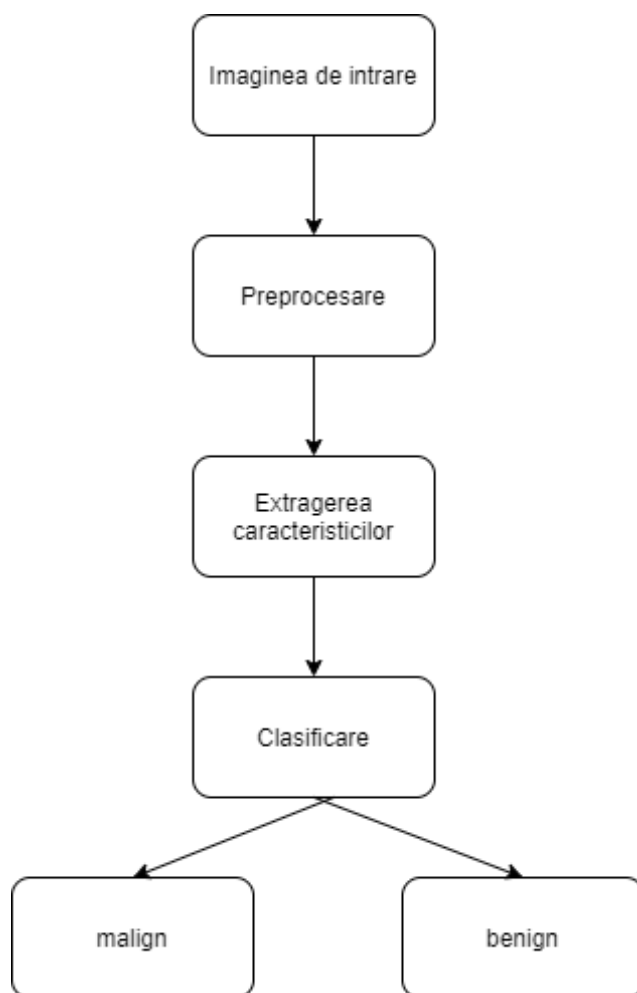
Deoarece construirea unui model de la zero și antrenarea acestuia poate deveni costisitoare în ceea ce privește durata antrenării, dar și pentru că uneori putem să nu obținem rezultatele pe care ni le dorim, există anumite tehnici pentru a preveni acest lucru, care au la bază transferul de învățare, iar extragerea caracteristicilor de tip bottleneck este una dintre acestea.

Strategia folosită pentru a implementa acest lucru presupune instantierea unei părți din model, mai exact, tot ce se află deasupra straturilor complet conectate. Folosim apoi partea de model de care avem nevoie pentru a extrage caracteristici pentru setul nostru de imagini care este mult mai mic în comparație cu cel cu care a fost deja antrenat modelul. Aceste caracteristici au denumirea de caracteristici Bottleneck(engl. Bottleneck features) și cu alte cuvinte, reprezintă ultima hartă de activare dinaintea straturilor complet conectate ale modelului original. Următorul pas presupune construirea unei mici rețele complet conectate pe baza a ceea ce am extras noi anterior, scopul fiind acela de a obține clasele de care noi avem nevoie pentru problema noastră.



Capitol 4 - Descrierea soluției

Pentru rezolvarea problemei propuse pentru lucrarea mea de licență am ales să folosesc un set de date cu ajutorul căruia să-mi antrenez un model pe baza căruia aplicația să fie capabila să recunoască dacă o leziune a pielii, o aluniță este malignă sau benignă. Găsirea arhitecturii potrivite pentru problema propusă și pentru setul de imagini a fost una din părțile care au creat problemele în ceea ce privește timpul acordat acestui aspect.



4.1 Setul de date

Crearea unui set de date adecvat este foarte importantă pentru orice lucrare, deoarece setul de date este un lucru principal necesar pentru proiectarea și testarea sistemului.

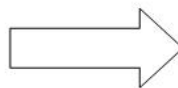
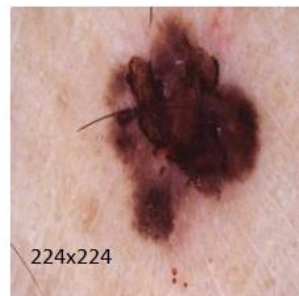
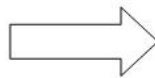
Setul de date folosit pentru această lucrare de licență a fost preluat de pe site-ul ISIC (International Skin Imaging Collaboration), site ce conține imagini cu diferite tipuri de alunițe clasificate de către specialiști în acest domeniu în două categorii: maligne și benigne. Am creat trei dosare pentru a împărți imaginile în funcție de scopul pentru care vor fi folosite: antrenare, validare și testare. Fiecare dintre aceste dosare conține două sub-dosare denumite în funcție de categoria din care fac pozele cu leziunile regăsite în ele: malign și benign.

Setul de date pentru antrenare conține un număr total de 9600 de imagini, împărțit în două categorii cu număr egal de imagini, adică cu câte 4800 de imagini pentru fiecare categorie. Setul de validare conține la rândul lui, 900 de imagini, câte 450 pentru fiecare categorie, iar setul de date pentru testare conține pentru fiecare categorie, 50 de imagini.



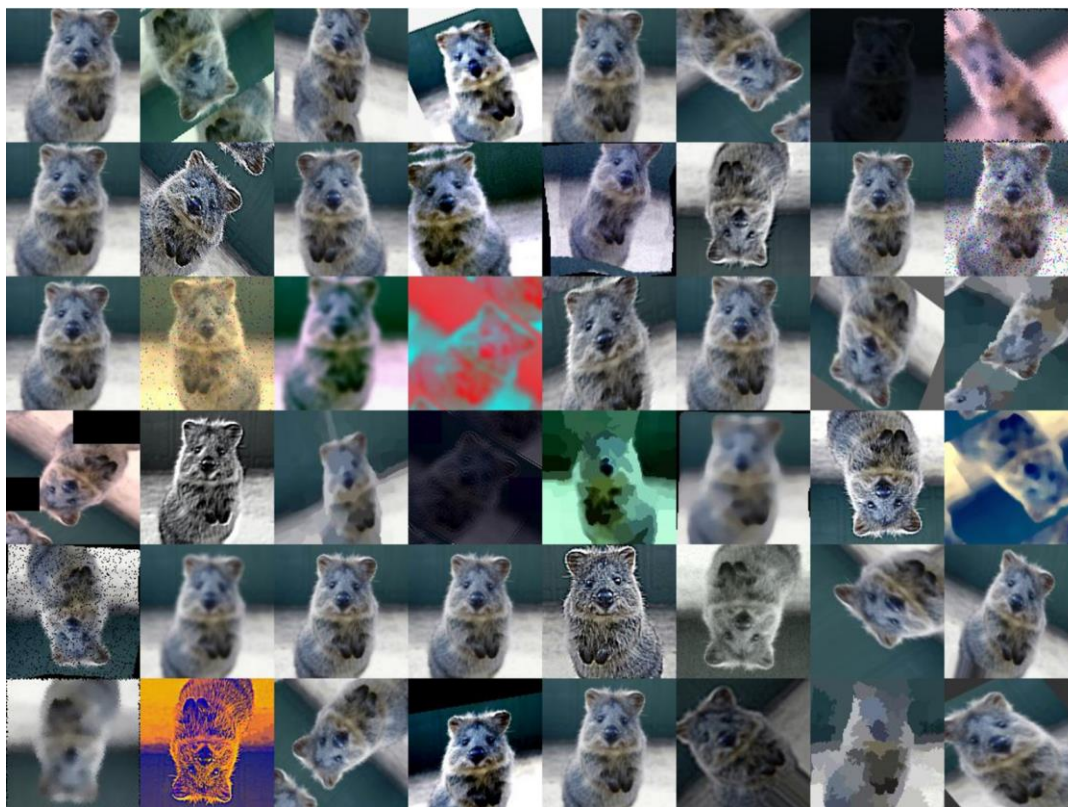
4.2 Preprocesarea setului de date

Pentru pregătirea setului de imagini pe baza căruia modelul va fi antrenat, validat și testat, s-a folosit operația de redimensionare a întregului set de imagini. Astfel, după această modificare, fiecare imagine din set are o dimensiune de 224x224. Dimensiunea a fost aleasă pentru a se potrivi cu inputul de care rețeaua pe care o vom implementa îl cere.



4.3 Augmentarea setului de date de antrenare

Performanța rețelelor neuronale de învățare profundă se îmbunătățește adesea odată cu cantitatea de date disponibile. Augmentarea setului de date de antrenare reprezintă procesul de aplicare a unor transformări simple sau mai complexe (flipping, transfer de stil pentru setul de date) asupra acestuia. Pe măsură ce rețelele neuronale convoluționale învață caracteristicile imaginii, dorim să ne asigurăm că aceste caracteristici apar la o varietate de orientări, pentru ca un model antrenat să poată recunoaște, de exemplu, o leziune care este fotografiată din alt unghi sau mai de aproape.



Pe lângă beneficiul de a fi capabili să ne antrenăm modelul pe un set de date mult mai numeros, prin augmentarea setului de date putem preveni și overfitting-ul. Prin overfitting înțelegem fenomenul de a obține rezultate foarte bune atunci când ne antrenăm modelul, dar în momentul în care acesta trebuie să clasifice o imagine pe care o întâlnește pentru prima oară, nu se mai obțin rezultate la fel de bune.

În TensorFlow, augmentarea setului de date se realizează cu ajutorul clasei `ImageDataGenerator`. Transformările pe care le facem sunt efectuate în memorie, deci nu avem nevoie de o stocare suplimentară, deși cu ajutorul parametrului clasei `save_to_dir`, putem realiza acest lucru în cazul în care avem nevoie.

Pentru setul de validare și testare, nu este indicat să folosim operațiuni de augmentare. În schimb, putem folosi clasa `ImageDataGenerator` pentru a scala imaginile, fără nicio augmentare suplimentară.

```
test_datagen=ImageDataGenerator(rescale=1./255)
```

Un exemplu prin care ne putem augmenta setul de date cu ajutorul clasei `ImageDataGenerator` este:

```
train_datagen = ImageDataGenerator( rotation_range=20,  
                                   width_shift_range=0.2,  
                                   height_shift_range=0.2,  
                                   shear_range=0.2,  
                                   zoom_range=0.2,  
                                   horizontal_flip=True,  
                                   vertical_flip=True,  
                                   fill_mode='nearest')
```

`rotation_range` – interval de grade pentru rotații random;

`width_shift_range` - fracțiune din lățimea totală (dacă valoarea <1, ca în acest caz) pentru a transla pe orizontală random imaginile;

`height_shift_range` - fracție din înălțimea totală (dacă valoarea <1, ca în acest caz) pentru a translate pe verticală random imaginile;

`shear_range` - unghiul de forfecare în sens contrar acelor de ceasornic în grade, pentru transformări de forfecare;

zoom_range - interval pentru zoom random;

horizontal_flip - valoarea booleană pentru întoarcerea aleatorie a imaginilor pe orizontală;

vertical_flip - valoarea booleană pentru întoarcerea aleatorie a imaginilor pe verticală;

fill_mode - punctele din afara limitelor de intrare sunt completate fie în funcție de „constant”, „nearest”, „reflect” sau „wrap”.

4.4 Detalii despre implementare

Pentru o rulare mult mai rapidă, am utilizat rularea pe GPU, și nu pe CPU. Motivul pentru care rularea pe GPU este mai rapidă decât cea pe CPU este că GPU conține mult mai multe nuclee decât un CPU, care are 4,6 etc. nuclee. Faptul că avem la dispoziție un număr mai mare de nuclee ne permite să rulăm mai multe procese în paralel, iar acest lucru face ca rularea să dureze mult mai puțin.

Deoarece am aflat destul de târziu de metoda rulării pe GPU, inițial rulând pe CPU, dar și pentru că obțineam rezultate care nu mă mulțumeau deplin, am căutat soluții pentru aceste probleme. Astfel, am aflat de metoda extragerii caracteristicilor de tip bottleneck și implicit de transferul de învățare. Încărcarea modelului cu arhitectura de tip VGG16, model deja antrenat cu ajutorul ImageNet, proces prin care se realizează transferul de învățare:

```
def bottleneck_features():  
    # Loading vgg16 model  
    # here we are using the transfer learning  
    vgg16 = applications.VGG16(include_top=False, weights='imagenet')
```

Adăugarea unui model mai mic pe care îl vom folosi pentru a putea fi posibilă antrenarea modelului cu propriul set de imagini:

```
#On top of the bottleneck features we are going to train a small model which will have target labels specific to our dataset
model = Sequential()
#we don't use the last layer
model.add(Flatten(input_shape=train_data.shape[1:]))
model.add(Dense(256, activation='relu'))

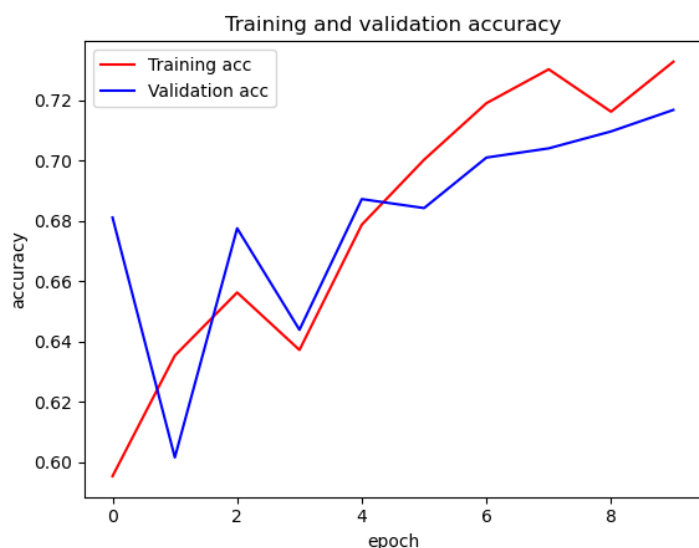
# adding Dropout to avoid overfitting
model.add(Dropout(0.5))
model.add(Dense(128, activation='relu'))

# adding Dropout to avoid overfitting
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='sigmoid'))
opt = keras.optimizers.Adam(lr=0.0001, beta_1=0.9, beta_2=0.999, epsilon=1e-08, decay=0.0)
```

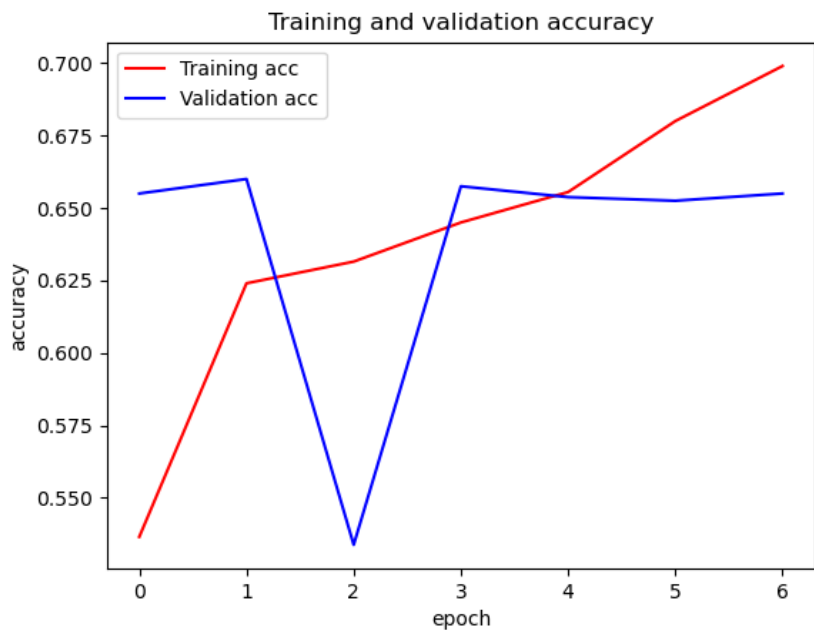
Capitol 5 – Experimente și rezultate

Atunci când vine vorba despre o aplicație care are nevoie de un model pe baza căruia să fie clasificate anumite imagini, procesul care durează cel mai mult, care este cel mai important și pe baza căruia trebuie să ne focusăm este acela de a găsi arhitectura modelului care este potrivit pentru problema noastră și tehnicile pe care să le folosim. Un alt aspect foarte important pentru crearea modelului care să ne clasifice imaginile cât mai bine este cel al alegerii corespunzătoare ale hiperparametrilor care controlează procesul de învățare. Astfel, pentru realizarea acestei lucrări de licență a fost nevoie de mai multe experimente în vederea obținerii celor mai bune rezultate, unele dintre acestea prezentându-le în cele ce urmează.

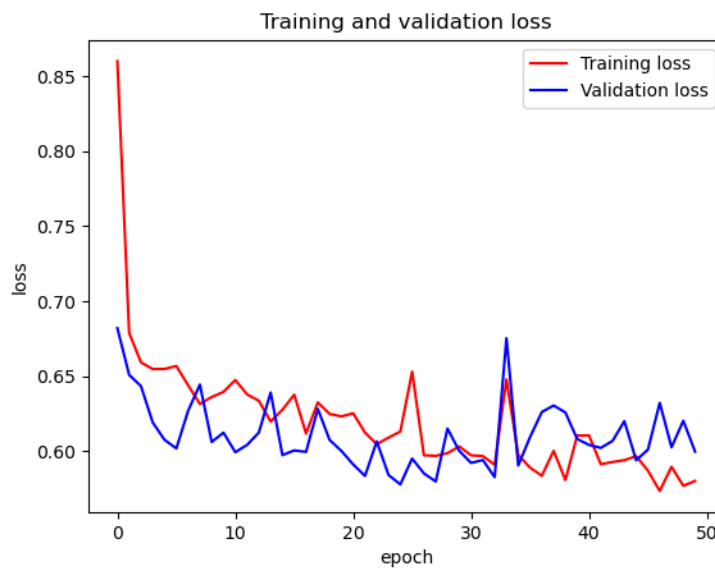
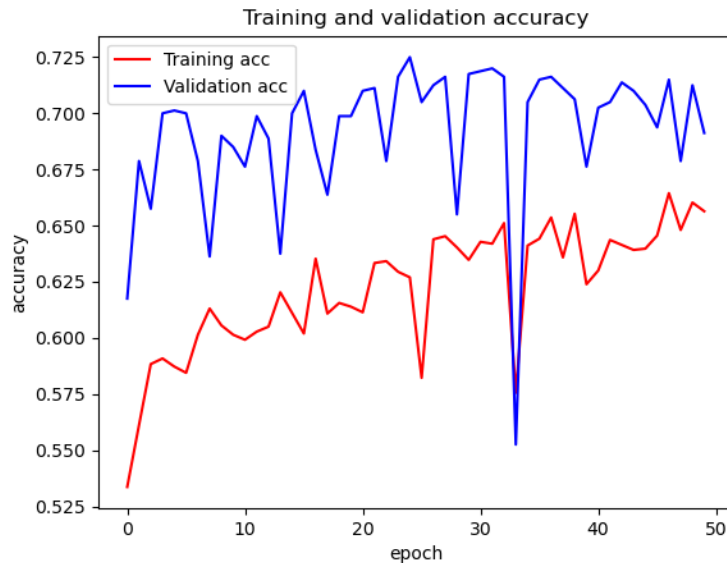
Ideea de la care am plecat a fost să începem cu un model simplu, cu mai puține straturi, ulterior, în funcție de rezultatele obținute să mergem în adâncime. Astfel, pentru început am construit un model cu o arhitectură de tip AlexNet, cu doar 5 straturi convoluționale. Cu acest model am obținut o acuratețe de 51.40%.



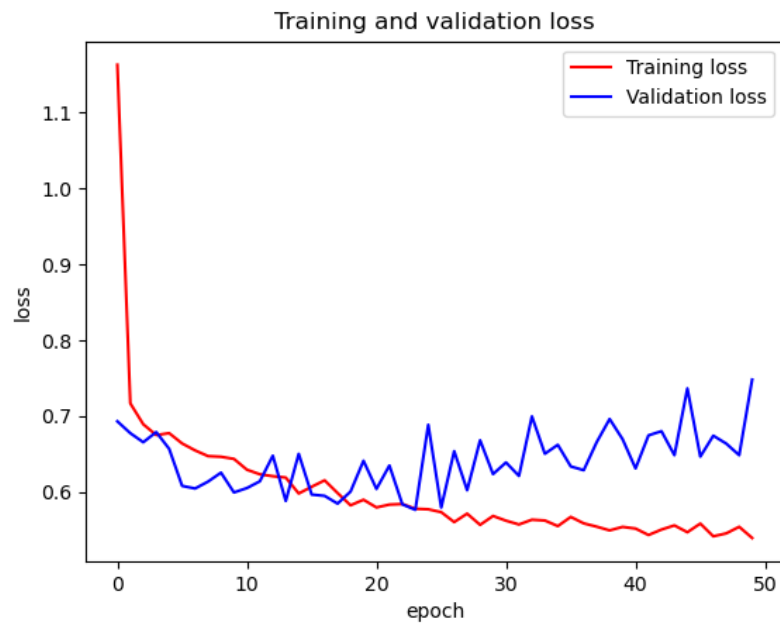
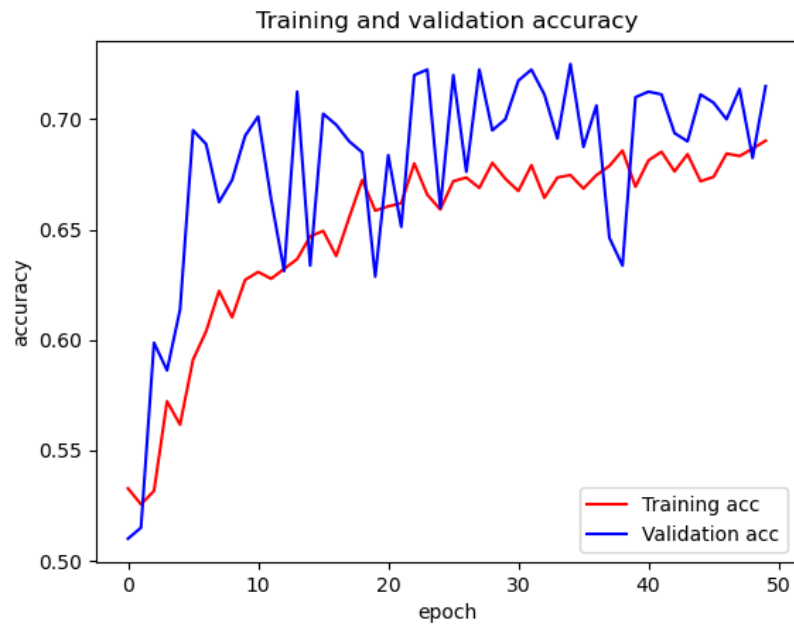
Mai departe, am construit de la zero un model de tip VGG16. Cu aceasta am obtinut rezultate mai bune decât cele precedente: o acuratețe de 61.88% și o pierdere de 0.5888721203804016.



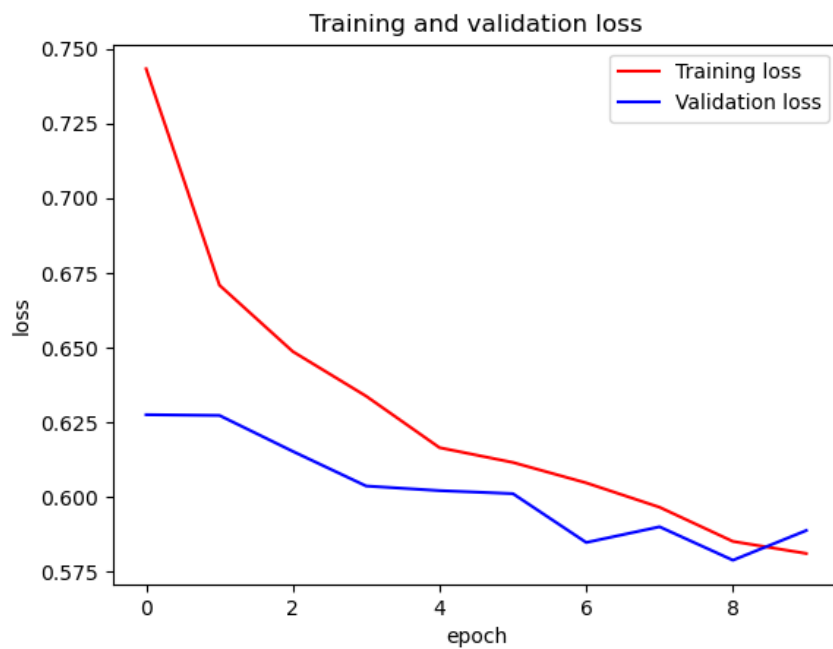
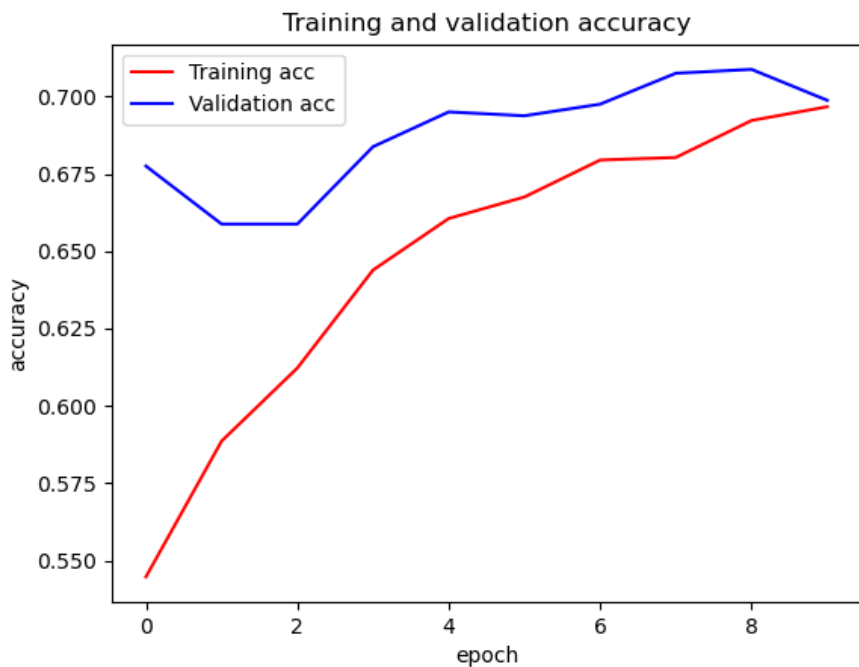
Fluctuațiile și salturile mari care aveau loc pe grafic m-au făcut să mă gândesc că modelul nu este stabil și este posibil să apară fenomenul de overfitting. Metoda următoare a fost cea cu extragerea caracteristicilor de tip bottleneck. Pentru început am folosit optimizatorul RMSprop. Initial, nu am folosit augmentarea datelor de antrenament, obținând astfel o acuratețe de 67.50% și o pierdere de 0.6499205198138952.



Rezultatele cu augmentarea datelor și tot cu optimizatorul RMSprop sunt următoarele: acuratețe: 71.50% și pierdere: 0.74826873421669



În final, după numeroase alte teste și documentare am folosit optimizatorul Adam, testând diferiți hiperparametri. Am obținut o acuratețe de 82,33% și o pierdere de 0.398486837314234



Capitol 6- Concluzii

Aplicația care are la bază această lucrare de licență are capacitatea de a identifica dacă o leziune a pielii, mai exact, o aluniță prezintă semne pentru un posibil cancer de piele, adică dacă este malignă sau dacă este benignă și nu este nevoie să ne face nicio grijă.

Consider că este o aplicație de actualitate și de foarte multă necesitate, în contextul în care foarte multe persoane se expun foarte mult la soare, fără a se proteja cum se cuvine, nefiind conștinete de efectele pe care le au razele soarelui asupra pielii noastre, căreia, după părerea mea trebuie să îi acordăm o atenție foarte mare. Cred că aplicația poate fi utilizată chiar și în domeniul dermatologic, ca un prim pas al unui consult de specialitate.

Există foarte multe îmbunătățiri care pot fi aduse acestei aplicații, iar unele dintre acestea le voi numi în cele ce urmează. Aplicația poate fi antrenată pe un set mult mai mare de date, astfel obținându-se cu siguranță niște rezultate mult mai bune. Dacă vom avea un set de date mai numeros, atunci consider că putem testa și o altă arhitectură, cu mult mai multe straturi, fapt ce va îmbunătăți rezultatele. De asemenea aplicația poate fi transformată într-o aplicație de mobil, lucru care ar face-o mult mai practică și la îndemână după părerea mea. Nu în ultimul rând, se poate mări numărul de leziuni care pot fi identificate cu această aplicație, lucru ce o va face mai variată și disponibilă pentru un număr mai mare de oameni, nu doar pentru cei care au de exemplu o aluniță suspectă.

Bibliografie

<https://keras.io/api/preprocessing/image/>

<https://towardsdatascience.com/how-to-use-data-augmentation-to-10x-your-image-datasets-dab42858be55>

<https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>

<https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148>

<https://colah.github.io/posts/2014-07-Understanding-Convolutions/>

http://rovislab.com/courses/ml/08_NeuralNetworks-CNN/08_Retele_Neurale_Convolutionale.pdf

https://pyimagesearch.com/wp-content/uploads/2018/12/keras_conv2d_activation_functions.png

<https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/>

<https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148>

<https://www.geeksforgeeks.org/vgg-16-cnn-model/>

<https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a>

<https://revistaurbania.ro/mitul-alunitelor-crescut-alarmant-mortalitatea-cancerului-de-piele/>

<https://docs.python.org/3/library/tkinter.html>

<https://neurohive.io/en/popular-networks/vgg16/>