**Landmark**

1. **Introducere**

Principala atracție într-o țară straină este reprezentată de obiectivele turistice, acestea atrăgând anual cei mai mulți turiști din toat colțurile lumii. Una dintre cele mai apreciate capitale europene când vine vorba de obiective turistice, și nu numai, este considerată Paris. Când spui Paris automat te gandești la minunatul Turn Eiffel sau la Arcul de Triumf, însă acestea nu sunt singure puncte atractive pe care le poți descoperi în capitala Franței. Din păcate însă, nu mulți sunt cei care știu cum arată Pantheon, Musee d’Orsay sau Popidoum, sau dacă știu, nu asociază imaginea obiectivelor cu numele acestora. Aceasta a fost ideea de la care a pornit Landmark.

Landmark reprezintă o aplicație de identificare a obiectivelor turistice principale din Paris, având ca bază de pornire o aplicație realizată de Google, TensorFlow for Poets. [1]. Cu ajutorul acesteia se pot identifica unul din cele 11 obiective consacrate ale Parisului: Defense, Turnul Eiffel, Hotel des Invalides, Lovre, Moulin Rouge, Musee d’Orsay, Notre Dame, Pantheon, Popidoum, Sacre Coeur, Arcul de Triumf. Majoritatea străinilor admiră frumusețile unui oraș însă nu toți cunosc denumirile obiectivelor pe lângă care trec. Astfel, utiliând Landmark, cu o simplă poză realizată la obiectivul dorit din Paris, utilizatorul va putea afla în timp real care este numele acestuia.

Turnul Eiffel este unul dintre obiectivele pe care nu ai cum sa nu le vezi din orice colț al Parisului. Cu ajutorul aplicației Landmark vei putea afla traseul de la locația curentă până la acesta, realizând o singura poză a obiectivului. Această funcționalitate a aplicației este posibilă cu ajutorul integrarii Google Maps-ului.

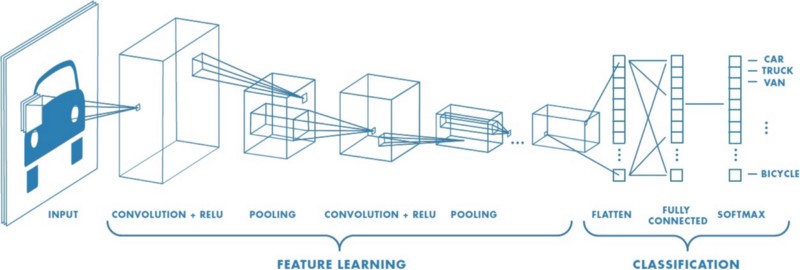
1. **Identificarea obietivelor**

**2.1 Rețele neuronale convoluționale**

Rețelele neuronale de convoluție sunt clasificatori special concepuți pentru lucrul cu imagini. Structura acestora (Figura 1) facilitează clasificarea datelor cu un număr mare de parametri, deoarece aceste tipuri de rețele permit prelucrarea eficientă a acestor categorii de date.

Primul strat este cel de concoluție unde se realizează convoluția dintre imaginea de intrare și un filtru(o matrice de MxN de dimensiuni mici). Acest strat produce una sau mai multe harți de trăsături(engl. *feature maps)*, acestea fiind imagini care conțin anumite caracteristici ale imaginii de intare. Acest strat este supus mai apoi unei funcții de activare neliniară (ReLU). Valorile filtrelor utilizate constituie ponderile conexiunilor, acestea modificându-se în etapa de antrenare în vederea micșorării erorii dintre rezultatul de la ieșire obținut și cel dorit. Următorul strat este unul de agregare. Acesta relizează o reducere a dimensiunii hărții determinate anterior pe fiecare grup de NxN pixeli. Valoarea unui astfel de grup este înlocuită cu un singur pixel, determinat ca valoarea maximă, suma pixelilor sau media acestora. Aceste două straturi se pot repeta de mai multe ori în funcție de aplicația dorită.

Urmează etapa de clasificare. Pentru început, matricea rezultată în urma procesului de învațare a trăsăturilor este vectorizară. Pe baza acstui vector se realizează clasificarea cu ajutorul stratului *fully connected*. La intrarea acestuia se furnizează hărțile de trăsături, urmând ca la ieșire să se aplice funcția de activare *softmax*. Ieșirea acestui strat este reprezentată de un vector de probabilități cu un număr de componente egal cu numarul de clase, fiecare reprezentând probabilitatea ca imaginea de la intrare sa se încadreze în clasa corespunzătoare. [2]



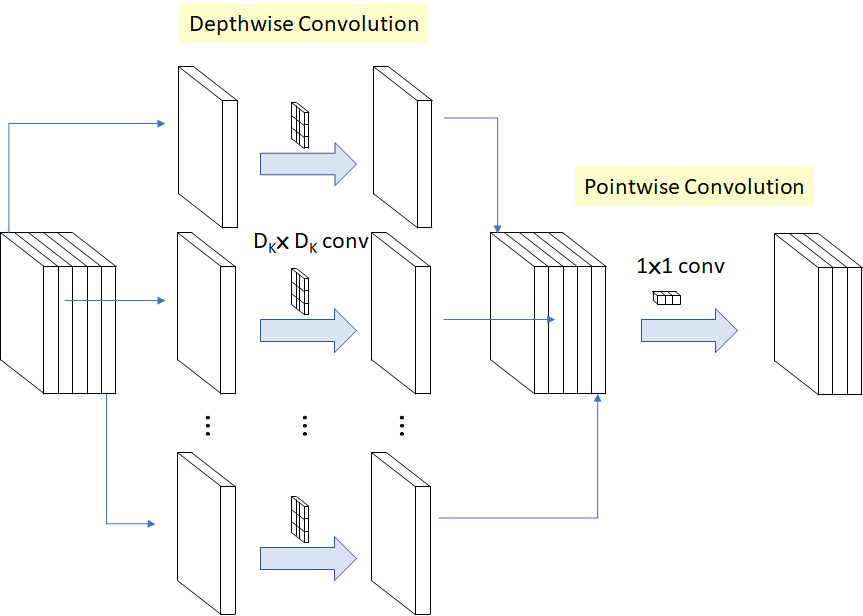
Figură 1 - Arhitectura unei rețele convoluționale standard [3]

**2.2 MobileNet**

MobileNet este o rețea neuronală convoluțională utilizată în special pentru aplicațiil mobile. Aceasta are la baza 28 de straturi și se diferențiază prin stratul convoluțional. În mod uzual, acesta este are dimensiunea filtrului de 3x3, fiindu-i aplicat ulterior o funcție de normalizare (Batchnorm) urmată de o funcție de activare neliniară (ReLU). MobileNet însă separă stratul convoluțional într-un strat de convoluție pe adâncime și de un strat de convoluție punctuală (Figura 2), ambele fiind urmate de Batchnorm și Relu. Prin această separare se reduce numărul de operații de multiplicare din timpul antrenării și testării, reducăndu-se asfel timpul de calcul necesar. De asemena, prin această arhitectură se face un compromis între acuratețe și complexitatea rețelei, acesta fiind necesar în cazul unei aplicații mobile. [4]

Pentru detectarea obiectivelor turistice am folosit o rețea MobileNet cu reducerea dimensiunii imaginilor la 224x224. Baza de date utilizată este cea pusă la dispoziție de Universitatea Oxford, aceasta conținând 6412 imagini cu cele 11 obiective turistice menționate. [5] După o curățare a bazei de date am rămas cu aproximativ 800 de imagini ce au fost folosite pentru antrenarea rețelei. Majoritatea claselor au avut un număr de aproximativ 80 de imagini, existând însă și excepții (Musee d’Orsay) pentru care baza de date a conținut doar 6 imagini.

În urma mai multor antrenări în care am variat mai mulți parametrii ai rețelei în vederea îmbunătățirii acureteței acesteia, am stabilit numărul de epoci la 1500 iar rata de antrenare la valoarea de 0.01. Utilizând acești parametrii, etape de antrenare a rețelei a durat aproximativ 6 ore, având în vedere și numărul de imagini din fiecare clasă. În urma antrenării cu imaginile din baza de date și validarea acestora cu ajutorul imaginilor de testare am obținut un graf cu ponderile asociate rețelei. Acest graf a fost încărcat în aplicația Android, pe baza lui clasificându-se ulterior imaginile analizate.



Figură 2 – Statul convoluțional la rețeaua MobileNet [6]

1. **Funcționarea aplicației**

Aplicația are la baza 3 activități principale: Start, Camera și Path. Trecerea de la o activitate la alta se face la comanda utilizatorului prin intermediul interfeței vizuale. Resursele hardware necesare rulării aplicației sunt: conexiunea la internet(Wifi sau date mobile), accesul la GPS și la camera frontală a terminalului mobil. În continuare vom descrie fiecare din cele 3 aplicații și vom vedea cum se îmbină acestea. [7]

**2.1 Start**

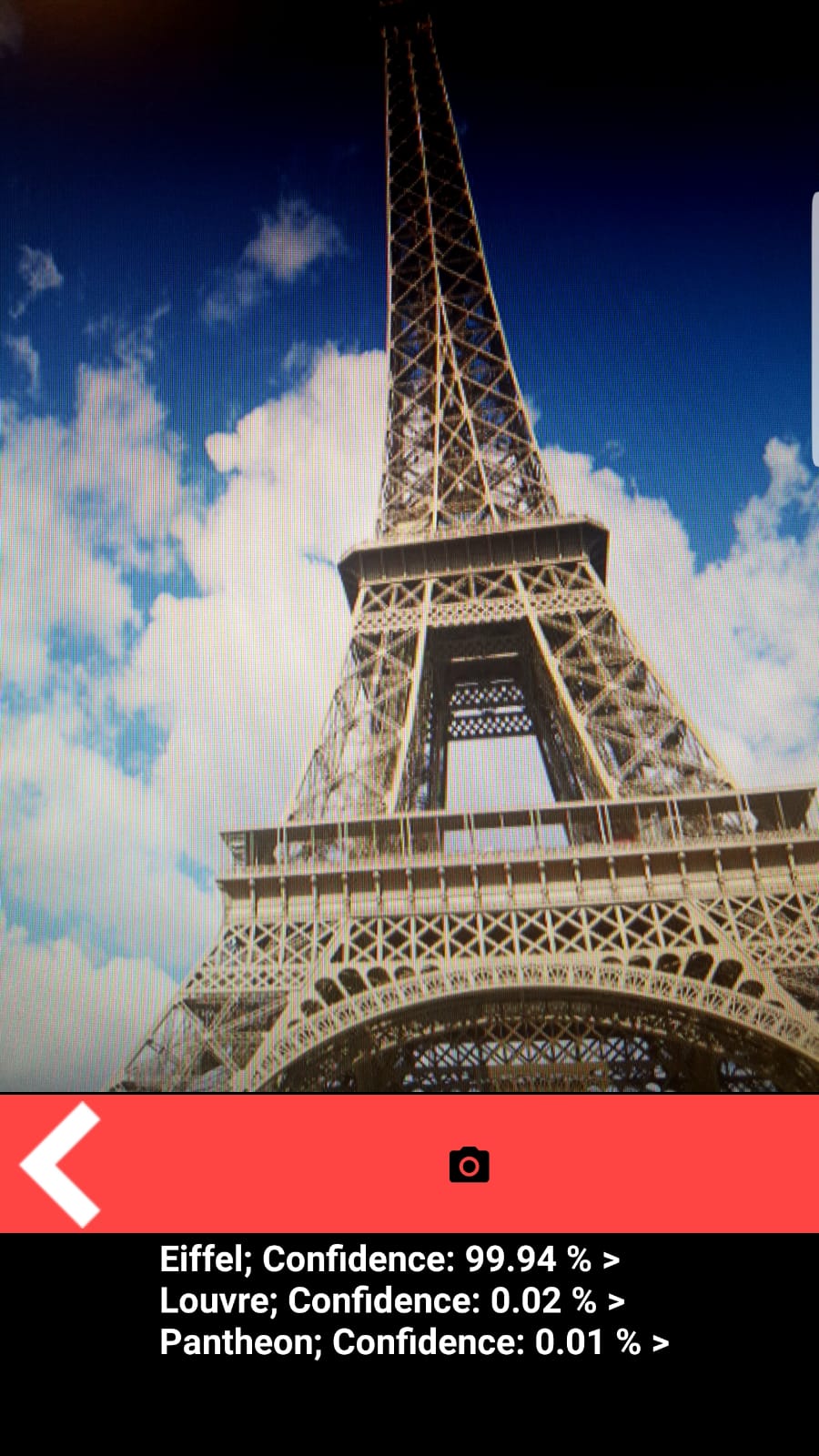


În momentul pornirii aplicației utilizatorul este întâmpinat de meniul de mai sus ce oferă acestuia informații legate de locația in care se afla, date de latitudine si longitudine, cât și date referitoare la starea vremii. Locația cat și starea vremii sunt obținute prin intermediul unui request la un serviciu WEB (<http://api.openweathermap.org/data/2.5/weather>) în care noi vom transmite datele de latitudine și longitudine, iar în schimb vom primi datele menționate anterior. Atât serviciul cât și noi vom transmite datele sub formatul JSON.

Pentru conversia temperaturii în grade Celsius și schimbarea pictogramei in funcție de starea vremii s-a utilizata o clasă numită WeatherDataModel.

Nu în ultimul rând activitatea are in colțul din dreapta sus un buton ce va permite utilizatorului trecerea către următoarea activitate (Camera).

* 1. **Camera**



Din fluxul video provenit de la camera foto a terminalului mobil, ia periodic câte un frame și îl trece prin rețeaua graful rețelei neuronale. Din răspunsul rețelei vom afișa pe ecran primele 3 clase în ordinea descrescătoare a valorii neuronilor din stratul de ieșire. Fiecare clasă are asociată în dreapta gradul de încredere. În momentul în care utilizatorul alege sa facă o poză aplicația trece poza prin rețea și se va alege clasa corespunzătoare neuronului din stratul de ieșire ce returnează valoarea maximă. Obținerea pozei determină de asemenea trecerea la ultima activitate. Pentru fiecare din cele 11 clase cu care a fost antrenată rețeaua noastră există date de latitudine și longitudine asociate.

* 1. **Path**



Imaginea cu obiectivul cât si coordonatele sale sunt afișate utilizatorului în această activitate iar utilizatorul este prezentat cu 3 opțiuni. Să se întoarcă la activitatea Start prin intermediul săgeții aflate in colțul din dreapta sus, să afișeze locația obiectivului în GoogleMaps sau să seteze o rută către obiectivul din imagine.

# **Referințe**

|  |  |
| --- | --- |
| [1] | Google. [Interactiv]. Available: https://codelabs.developers.google.com/codelabs/tensorflow-for-poets-2-tflite/#0. [Accesat 25 01 2019]. |
| [2] | „Rețele neuronale de convoluție,” [Interactiv]. Available: http://myac.xhost.ro/inva/Lab11/InvA\_Lab11.pdf. [Accesat 27 01 2019]. |
| [3] | „Towerds Data Science,” [Interactiv]. Available: https://towardsdatascience.com/classifying-skin-lesions-with-convolutional-neural-networks-fc1302c60d54. [Accesat 25 01 2019]. |
| [4] | A. G. Howard, *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision,* Google Inc.. |
| [5] | J. Philbin și A. Zisserman, „The Paris Dataset,” [Interactiv]. Available: http://www.robots.ox.ac.uk/~vgg/data/parisbuildings/. [Accesat 27 01 2019]. |
| [6] | „Towards Data Science,” [Interactiv]. Available: https://towardsdatascience.com/review-mobilenetv1-depthwise-separable-convolution-light-weight-model-a382df364b69. [Accesat 27 01 2019]. |
| [7] | P. Muellauer, „Android O & Java - The Complete Android Development,” [Interactiv]. Available: https://www.udemy.com/android-app-development-with-java/. |