

Model de arhitectură pentru traducerea automată și migrarea aplicațiilor legacy în medii de calcul cloud

(Raport științific - Activitatea 1.3)

Andrei Ionut DAMIAN

Cloudifier SRL
București, România
Damian@cloudifier.net

Nicolae TAPUS

Departamentul de Informatică și Inginerie
Universitatea POLITEHNICA din București
București, România
Ntapus@cs.pub.ro

Rezumat calcul -on-cerere, SaaS, PaaS, și , în general , Cloud Computing este în prezent principala abordare prin care domenii atât academice și comerciale sunt sisteme și livrarea conținutului. Cu toate acestea, rămâne un segment imens de sisteme și aplicații vechi, de la sisteme de contabilitate și management, până la software științific bazat pe arhitecturi desktop clasice sau simple client-server. Deși în ultimii ani tot mai multe companii și organizații au investit bugete importante în traducerea aplicațiilor moștenite în mediul cloud-enabled online, rămâne încă un segment important de aplicații care, din motive nu au fost traduse. Acest document propune o arhitectură inovatoare pentru conducte pentru traducerea automată și migrarea aplicațiilor moștenite în mediul compatibil cu tehnologia cloud cu costuri minime de dezvoltare software.

Cuvinte cheie - programare automată; Cloud computing; Migrație; învățare mecanică; Traducerea automată

I. INTRODUCERE

Traducerea automată a aplicațiilor moștenite a reprezentat întotdeauna un domeniu important de cercetare pentru comunitatea științifică cu mare impact economic, în special în domeniul întreprinderilor mici și mijlocii. Cu fiecare sosire a unor noi limbi de programare, cadre sau limbi oficiale de modelare, omul de știință de calculator a urmărit cercetarea și dezvoltarea diferiților traducători de cod care ar migra automat din codul sursă al aplicației vechi în noul limbaj de programare. Cele mai reușite proiecte de cercetare și de producție de-a lungul anilor s-au aflat în domeniul generării codurilor automate din limbile oficiale de modelare [1] [2] [3] [4] și în domeniul traducerilor automate de coduri [5] [6] [7]]. Cu toate acestea, procesul de migrare a codului sursă

a fost întotdeauna însoțit de procese de re-engineering care ar fi trebuit să optimizeze mecanica internă a sistemelor software nou-migrate. Un alt obiectiv al traducerii software a fost și este totuși migrarea aplicației moștenite de la un mediu depreciat la unul mai modern, robust și mai fiabil - cum ar fi migrarea de la aplicații bazate pe desktop la client-server și apoi la mediile Cloud Computing. Cu toate acestea, în mai multe ocazii, datorită lipsei codului sursă veche, migrarea reală a fost forțată prin utilizarea terminalelor sau a desktop-urilor virtuale.

În lucrarea noastră ne propunem un model de arhitectura inovatoare concepute pentru provocările reprezentate în special de către interfața de utilizator de aplicații și fluxul de date re-inginerie de aplicații moștenite în cazul în care codul sursă **nu** este disponibil. Modelul nostru va folosi tehnici de învățare a mașinilor și în acest fel va putea "observa" și va deduce comportamentul funcțional al aplicației moștenite. La sfârșitul procesului inferențial, modelul nostru de conducte va fi capabil să genereze o definiție intermediară scenariată a aplicației moștenite observate. În cele din urmă, definițiile deduse și scripturi definite de propriul script de limbă de interpretare intermediară (ITDL pe scurt) vor fi folosite de ultimul strat al modelului nostru de conducte pentru a genera codul țintă al platformei.

II. MUNCA ÎNRUDITĂ

A. *Stadiul actual al tehnicii de segmentare a imaginii semantice*

Una dintre primele domenii în care lucrează munca noastră constă în actuala tehnologie de segmentare semantică a imaginii. Pentru ca conducta noastră de model să realizeze rezultatele dorite, este necesar ca stratul de învățare a mașinii să aibă capacitatea de a identifica corect controalele vizibile ale interfeței utilizator pentru aplicația moștenită observată și locațiile acestora. În cele din urmă, modelul nostru de învățare a mașinilor va putea deduce aspectul general al scenei pentru fiecare ecran de interfață utilizator.

Problemele de ultimă oră și de cercetare, cum ar fi clasificarea imaginilor [8] și segmentarea semantică a imaginii [9], oferă noi oportunități pentru diverse sarcini legate de învățarea mașinilor, de la recunoașterea unei pisici la o imagine, a poziției sale efective în scenă și definirea tuturor atributelor scenei, cum ar fi toate obiectele cu dimensiuni și poziții.

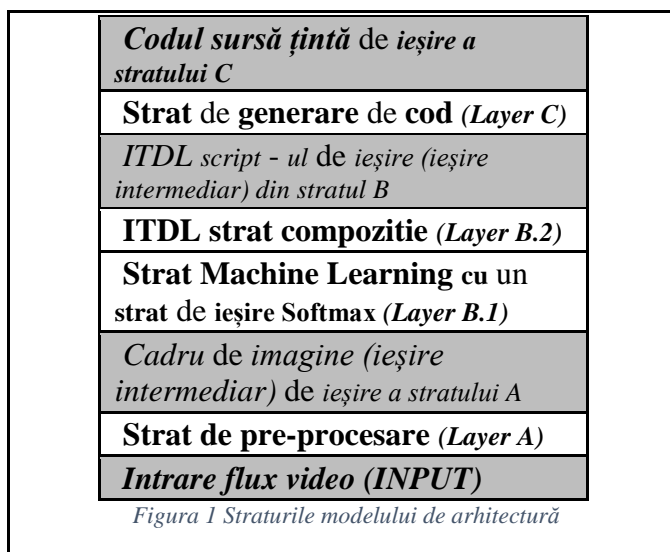
B. *Avansuri în performanța inferențelor de învățare a mașinilor*

Progresele recente în domeniul prelucrării informatice, a capacității de stocare și de comunicare - și în special în domeniul prelucrării paralele masive numerice / științifice pe bază de GPU [10], au adus o îmbunătățire extraordinară abordărilor existente sau în curs de desfășurare, cum ar fi modele neuronale profunde. Tehnologiile GPU, cum ar fi arhitectura NVIDIA Pascal, oferă la niveluri acceptabile de prețuri pentru o putere de calcul de înaltă performanță capabilă să livreze peste 10 TFLOPS de calcul paralel pe mii de nuclee computaționale numerice. Concursurile de recunoaștere a imaginilor de prestigiu, cum ar fi competiția ImageNet [11], demonstrează o rată uimitoare de îmbunătățire a ultimilor ani, legate atât de utilizarea învățării profunde, cât și de rețelele neuronale deosebit de profunde. Folosirea formării masive de calcul paralel a modelelor de învățare bazată pe GPU a fost o abordare câștigătoare în competițiile de recunoaștere a imaginii - de la câștigătorul competiției din 2010 care a obținut o precizie de 72% până la câștigătorul competiției din 2015, cu o precizie de peste 96% .

III. MODELUL ARCHITECTURE

Arhitectura generică a conductei noastre propuse prezentată în figura (1) se bazează pe o perioadă de trei straturi principale: (a) **stratul A** - un strat de bază preprocesare date constând dintr-un / flux video de imagine captura motor care va fi responsabil de „Respectand” sistem de moștenire , în scopul de a alimenta informații de intrare la stratul următor (b) **stratul b** - învățare

strat predictiv responsabil de analiza a fluxului de intrare cadru cu cadru și reconstruind prin intermediul ITDL complet interfață utilizator aspectul scenă și deducție modele și controlează mașina structura; (c) **stratul C** - pe baza de ieșire mașină de învățare strat ITDL stratul final va genera un cod sursă într - o limbă de alegere (bazate pe reguli de dicționar) pentru opiniile deduse de interfață utilizator cu propriile lor modele și controlere.



A. De la fluxul video la ITDL la codul sursă

Principala inovație propusă a lucrării noastre în domeniul programării automate este introducerea unei conduite de învățare a mașinilor, care va fi capabilă să realizeze analiza și reconstrucția interfeței utilizator în timp real pentru cazul pornirii sistemelor vechi în mediile Cloud computing.

Stratul de învățare al mașinii propus în cadrul arhitecturii modelului nostru de conduite este responsabil pentru inferența scenariului interfeței utilizator, așa cum am menționat anterior. Modelul de învățare predictivă a mașinilor trebuie să analizeze un flux de imagini livrate (ca film / flux sau mediu real-time de aplicație) dintr-o interacțiune reală între un utilizator și sistemul vechi. Procesarea reală constă în citirea unui flux de ecrane de înaltă definiție la peste Caracteristici de predicție pe ecran și cel puțin 10 cadre pe secundă, posibil pentru mai multe ore. În cele din urmă, modelul de învățare a mașinilor poate deduce ecranele, generând rezultate în timp real pe baza limbii scriptului ITDL. Obiecte individuale din cadrul fluxului video analizate sunt clasificate pe baza ecuației de clasificare Softmax (1) , în cazul în care *este vectorul transpusa ponderilor*

caracteristicilor învățate pentru un anumit tip de clasă de control al interfeței cu utilizatorul aplicației (dintr-un total o clasă N formate).

$$h_{softmax}(x_i^{(j)} | \theta, j \in M, i \in N) = \frac{e^{\theta^T x_i^{(j)}}}{\sum_k^N e^{\theta^T x_k^{(j)}}} \quad (1)$$

Datorită utilizării funcției de ipoteze în stratul mașină de învățare a modelului de conducte, vom fi în măsură să determinăm reale „probabilitatea de interfață de control”, că un anumit control interfață de utilizator identificat aparține unei anumite clase de ls contro, cum ar fi buton, etichetă, și alte controale ale interfeței utilizator.

Pentru inferența poziției individuale a obiectului și obiectivul ansamblului de scena global, va fi utilizat un mecanism de ferestre glisante cu direcție simplă. În ceea ce privește complexitatea, modelul conduitei va utiliza o gamă variată de ferestre de alunecare pe baza datelor de instruire furnizate de conducte. Cu toate acestea, adaptarea modelului de conductă într-un mediu de învățare online va oferi un flux continuu de cazuri de formare (cum ar fi forme și comportamente noi de control) care vor permite modelului conduitei să se adapteze automat.

Definiția reală a ecranului ecranele de interfață cu utilizatorul, inclusiv modelul și informațiile controler se bazează pe un script ITDL JSON / dicționar stil așa cum se arată în *Output 1*.

Ieșire 1 ITDL de ieșire (generic)

```
Rezultat = {
  <Nume element șir>: {
    "TYPE": {[<valoare>: <procent>],}
    "LABEL": <valoare>,
    "COORD": {<valori>}
    "SIZE": {<values>}
    "ACȚIUNE": {<valoare>}
    [<Numele sub-elementului șir> {
      [<Definition>]],
      .....]
  }
}
```

B. Modelul algoritmului de traducere

Procesul generic care descrie fluxul de lucru global al modelului nostru de conducte este prezentat în următorul algoritm:

Algoritmul 1 Pipeline Modelul general Algoritmul

```
S0: Faza de instruire
  S0.1: B.2 În cazul în care pre-instruit de încărcare pre-instruit
    Altfel
  S0.2: preprocesa date CCD de formare cu strat A
  S0.3: Strat de tren B.1 model de mașină de învățare
S1: stratul A se conectează la fluxul video de aplicație veche
S2: Pentru fiecare cadru video observate de către Layer A
  Date preprocesa din cadrul stratului A: S2.1
  S2.2: Layer B.1 primește date și prezice scena
  S2.3: Layer B.2 a primit predicții și se acumulează
  S2.4: Dacă Layer B.2 a primit nouă scenă din prev
    S2.4.1: Generarea scena anterioară ITDL
    S2.4.2: Trimite scena ITDL la Layer C
```

S2.4.3: *Strat C generează cod sursă scenă*

S3: Faza de formare online

S3.1: În cazul în care *etichetarea on - line* disponibile

S3.1.1: Train-actualizare modelul *B.1*

S3.1.2: Adăugați eșantion la depozitul de antrenament

S3.1.3: Salvați de *pre-instruit* model de mașină de învățare *B.1*

Sfârșit
Întoarcere

C. Utilizare caz

O prezentare a unei cereri de moștenire caz de utilizare în viața reală și de ieșire ITDL reală a modelului se va clarifica mai bine procesul real descris în *algoritmul 1*. În acest scop, vom folosi o reală aplicație desktop ferestre de viață prezentate mai jos în *Figura 2*. Din motive de confidențialitate evidente în *Figura 2* detaliile / date în cadrul unora dintre domeniile a fost disimulat.

Cazul de utilizare a testelor propuse se bazează pe un interval de timp scurt "de analiză" pe care modelul nostru de conducte a realizat-o pe ecranele interfeței de aplicații vechi. Prima etapă a modelului nostru de conducte presupune că stratul de învățare a mașinilor este deja pre-instruit într-un set de date de antrenament specializat în controlul interfeței utilizatorilor de aplicații. *Layer A* modelului nostru de conducte va capta ecranul aplicație așa cum este descris în *figura 2* și se va aplica de extracție segment imagine bazată pe diverse glisante-ferestre (pas *S2.1* algoritmului nostru model de conducte). Pentru fiecare fereastră extrasă, clasificatorul Softmax va deduce opțiunile de vârf pentru segmentul respectiv în ceea ce privește clasa de control al interfeței utilizator. Având în vedere o probabilitate mai mare decât media pentru o anumită fereastră de alunecare, putem să atribuim diferite proprietăți, cum ar fi dimensiunea și locația, urmate de inferența conținutului de control (cum ar fi legătura cu butonul interfeței utilizator sau textele de opțiune)

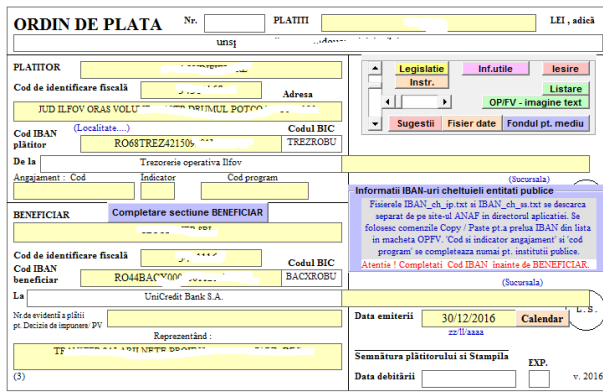


Figura 2 Aplicație moștenită de pe desktop

În următorul scenariu de ieșire (ieșire 2) vom prezenta proprietățile scena pentru cazul deduse de utilizare de mai sus:

Ieșire 2 Utilizați Cauza ITDL ieșire

Rezultat = {

"LbIORDIN": {

"TIP": {

"CAPȚIUNE": 0,97

},

"LABEL": "ORDIN DE PLATA",

"COORD": {X: 5, Y: 5};

"SIZE": {X: 150, Y: 20};

"ACȚIUNE": "Niciuna",

"BOUNDDATA": "Niciuna",

"COPII": "Niciuna"

},

"LbINr": {

"TIP": {

"AUTOEDIT": 0,55,

"EDIT": 0,41,

},

"LABEL": "Nr.",

"COORD": {X: 165, Y: 4};

"SIZE": {X: 40, Y: 20};

"ACȚIUNE": "Niciuna",

"BOUNDDATA": "LbINr_Data",

"COPII": "Niciuna"

},

"LbIPLATITI": {

"TIP": {

"EDIT": 0,89

},

```

    "LABEL": "PLATITI:",
    "COORD": {X: 215, Y: 4};
    "SIZE": {X: 180, Y: 20},
    "ACTION": "on_lblPLATITI_Change"
    "BOUNDADATA": "Niciuna",
    "COPII": "Niciuna"
  },
  "LbLEIadica": {
    "TIP": {
      "LABEL": 0,91
    },
    "LABEL": "LEI, adică",
    "COORD": {X: 395, Y: 5};
    "SIZE": {X: 35, Y: 20};
    "ACȚIUNE": "Niciuna",
    "BOUNDADATA": "Niciuna",
    "COPII": "Niciuna"
  },
  "GidUnnameGridArea":
  {{
    "TIP": {
      "VIEWGRID",: 0,99
    },
    "LABEL": "Niciuna",
    "COORD": {X: 5, Y: 35};
    "SIZE": {X: 450, Y: 350};
    "ACȚIUNE": "Niciuna",
    "BOUNDADATA": "Niciuna",
    "COPII": {
  "LbPLATITOR": {
    "TIP": {
      "COMBOEDIT": 0,59,
      "AUTOEDIT": 0,30,
      "EDIT": 0,10
    },
    "LABEL": "PLATITOR:",
    "COORD": {X: 3, Y: 3};
    "SIZE": {X: 185, Y: 20};
    "ACTION": "on_lblPLATITOR_Change"
    "BOUNDADATA": "lPlATITOR_Data",
  },
  ...
}

```

După cum este prezentat în *Output 1* modelul nostru de conducte generează o inferență completă scena pentru aplicația bazată pe desktop cu ecran capturate în Figura 2. Pe baza interacțiunii utilizatorului real cu sistemul de moștenire conducta generează o probabilitate maximă de

probabilitate pentru fiecare clasă de utilizator de interfață de control. Clasa de control a interfeței utilizator inferioare este prezentată împreună cu alți potențiali candidați potențiali, dacă există.

IV. CONSIDERAȚII PRIVIND PUNEREA ÎN APLICARE

După cum sa menționat anterior, stratul de învățare a mașinilor al modelului nostru de conducte se va baza pe o arhitectură de model superficială. Motivul principal al utilizării unui model de învățare mecanică superioară în locul unei rețele clasice neuronale profilactice pentru recunoașterea imaginii se bazează pe ghilimele datelor de antrenament pentru fiecare element de interfață individuală utilizator în momentul experimentării sistemului. Pentru a instrui o rețea de convoluție profundă pentru inferența completă a scenei, descrisă de [9] [8] [12], am avea nevoie de un model pre-instruit sau de un set de date de antrenament masiv. Datorită naturii scenelor țintă pe care trebuie să le deducem, nu vom putea folosi modele pre-instruite pe imagini / fotografii, cum ar fi modele bine cunoscute de tip convoluție, cum ar fi AlexNet [8] sau Inception [13]. În prezent, sunt disponibile metode bine cunoscute pentru îmbogățirea setului de date, cum ar fi generarea de eșantioane și etichetarea bazate pe SVM [14] sau alte tehnici de regresie simplă. Cu toate acestea, în cazul nostru, observațiile pentru eșantioanele din clasa de control ale interfeței individuale sunt doar prea puține pentru a aplica orice fel de regresie sau îmbogățire a setului de date pe bază de zgomet.

Ca rezultat, un set mic de formare dimensiune variabilă pot fi furnizate efectiv așa cum este descris datele eșantionate în *tabelul 1* și *ecuația 2*, unde *este vectorul caracteristica pentru* m^{th} probei de formare de un set de date de formare oferite de probe M și o probă maximă dimensiuni C , este real cls eticheta de clasa de control (de exemplu, „B” pentru butonul de interfață și „L” pentru eticheta vizuală). Rețineți că fiecare probă de formare *aparține unei anumite clase și fiecare clasă are un spațiu dimensional particular*.

Nr	Nr F	F 1	F 2	Cls
...
K				$\frac{N}{A}$	$\frac{N}{A}$.	B
...

J					$\frac{N}{A}$.	L
...

Tabelul 1 - Eșantionarea datelor de antrenament

(2)

În principiu, setul de date de instruire pentru modelul conduitei va conține o multitudine de sub-seturi de date, fiecare sub-set de date cu spațiul propriu de dimensiuni. Faza preprocesare a *stratului A* în cadrul modelului nostru de conduite este responsabil de scalare a probelor la spațiul dimensional fereastră glisantă actuală.

Datorită potențialului ridicat al naturii paralelizabil straturilor noastre în cadrul arhitecturii pipeline - cu accent special pe *B.1 Layer*, *B.2 Layer* și *stratul C* - folosim o abordare mapreduce pentru fiecare dintre straturile menționate. La nivelul modelului stratului de învățare a mașinilor, putem genera concluzii în paralel pentru mai multe ferestre glisante observate în fluxurile video. Ca rezultat, modelul de conduite propus este o scalare grațioasă a tuturor calculelor cerute de modelul de învățare a mașinilor pe resursele disponibile. Utilizând fie OpenCL [15], fie CUDA [10], implementăm kernel-uri pe o infrastructură GPU accesibilă momentan, care va calcula inferențele la nivelul ferestrei de alunecare pentru fiecare și toate controalele cunoscute și etichetate.

Utilizarea feedback - ul-bucloa propusă descrisă de *S3.1.1*, *S3.1.2* și *S3.1.3* sau *Algoritm 1* putem alimenta etichete corecte înapoi la modelul deja pregătit pentru a-on - line de actualizare a parametrilor de model și , astfel , a îmbunătăți acuratețea inferență scena. În cele din urmă, bazându-se pe un mediu Cloud Computing, modelul conduitei ar trebui să poată actualiza continuu greutatea formate pe baza medierii modelului de la lucrările de inferență scena secvențială sau paralelă și să creeze un model puternic de învățare mecanică pre-instruit.

V. CONCLUZII

Arhitectura modelului de gazoducte propusă este capabilă să realizeze un important proces automatizat cu o valoare ridicată pentru o multitudine de entități și în special pentru întreprinderile mici și mijlocii. Impactul propus pentru comunitatea întreprinderilor mici și mijlocii, care este motorul economiei mondiale, este ridicat în ceea ce privește natura sa de economisire a costurilor. Practic, procesul nostru automatizat propus, folosit de modelul conduitei, economisește peste 50% din costurile de traducere și migrare legate de transformarea unei aplicații moștenite într-o aplicație bazată pe Cloud.

Lucrând într-un mediu în timp real și on-line, modelul conduitei este capabil să deservească mai mulți clienți pe baza arhitecturii de procesare cloud computerizată. Angajarea de instruire și inferență de învățare automată online se bazează pe modelul nostru de conduite cu puterea de adaptare automată. Cu toate acestea, procesul nostru de cercetare în curs de desfășurare va continua să transforme arhitectura propusă pentru a adopta noi tehnologii, limbi și medii de execuție bazate pe cloud.

REFERINȚE

- S. Viswanathan și P. Samuel, „generarea automată a codului , folosind modele de activitate limbaj de modelare și de secvență unificate,“ *Software - ul IET*, voi. 10, nr. 6, 2016.
- [1] R. Campos-Rebelo și F. Pereira, " De la plase lopt Petri la C: Un instrument automat generator de cod," în *Informatică Industrială (INDIN) 2011 nouă IEEE International Conference on* 2011.
- [2] S. Diswal, PWW Tran-Jørgensen și PG Larsen, "generarea automata de C # și .NET Code Contracte din VDM-SL Modele" , în *data de 14 Uvertura Workshop: Către Analitic Tool Chains: Raport tehnic ECE - TR -* 28, 2016.
- [3] B. Milosavljević, M. Vidaković și Z. Konjović, „generare automată de cod pentru aplicații web orientate pe baze de date“ , în *PPPJ '02 / '02 IRE lucrările conferinței inaugurale privind principiile și practicile de programare*, 2002.
- [4] M. Trudel, M. Oriol, CA și M. Nordio Furia " traducere automată a Java Source Code la Eiffel" , în *49th International Conference, TOOLS 2011, Zurich, Elveția, 28-30 iunie, 2011. Proceedings*, 2011.
- [5] G. Paulsen, J. Feinberg și X. Cai, "Matlab2cpp: A Matlab-to-C ++ cod traducător" , în *Sistemul Sistemelor Engineering Conference (sose)*, 2016 11, 2016.
- [6] S. Chadha, A. Byalik și E. Tilevich, "Facilitarea dezvoltării de software cross-platform prin sinteza automata de cod de resurse de programare bazate pe web," *Limbaje, Sisteme si Structuri*, vol. 48, 2017.
- [7] A. Krizhevsky, I. Sutskever și G. Hinton, "Clasificarea ImageNet Deep convoluțională Networks," *Advances in sistemele de prelucrare a informațiilor neuronale* 1097-1105, 2012.
- [8] J. lung, E. Shelhamer și T. Darrell, "Rețele complet convoluțional pentru Segmentarea semantic," *IEEE model de analiză și Machine Intelligence ISSN: 0162-8828*, nr. Mai 2016, 2016.
- [9] J. Ghorpade, J. Parande și M. Kulkarni, "PROCESAREA GPGPU IN ARHITECTURA CUDA," *Advanced Computing: An International Journal (ACIJ)*, vol. 3, 2012.
- [10] ImageNet, "Large Scale ImageNet vizuală Concurență Recunoaștere (ILSVRC)," în *www.image-net.org/challenges/LSVRC/*.
- [11] A. Karpathy și L. Fei-Fei, "aliniamente vizual-semantic
- [12] profundă pentru generarea de descrieri imagine (arXiv:

1412.2306)," *Computer Vision si Pattern Recognition* 2015.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke și A. Rabinovich, "Mergând mai aprofundate cu convoluții,"

[13] *Recunoașterea Computer Vision și model* 2014 .

R. Mao, H. Zhu și L. Zhang, „o metodă nouă pentru a ajuta mici setul de date neuronale de învățare de rețea“ , în *sisteme inteligente de proiectare și aplicații*, 2006. *ISDA '06. A șasea Conferință internațională privind*

[14] 2006.

K. Wang, J. Nurmi și T. Ahonen, „Accelerarea calculării pe un telefon Android cu OpenCL Paralelism și distribuție Optimizarea volumului de lucru între un telefon și un serviciu Cloud,“ în *ubicuu Intelligence & Computing, Advanced și Trusted Computing, Computing scalabilă și Comunicațiilor , Cloud Computing și Big de date, Internet de Oameni, și Smart World Congress (UIC / ATC / ScalCom / CBDCOM / PIO / SmartWorld)*, 2016

[15] *Conferințe Intl IEEE*, 2016.

P a g i n ă | 6

Proiect ID: P_38_543, Nr. Ctr. 98 / 09.09.2016, MySMS: 104349, Apel - POC-A1-A1.2.1-C-2015,