Nr.: 139/30.12.2016

Raport stiintific

de cercetare-dezvoltare in cadrul Cloudifier SRL

Nume proiect	Platforma de migrare automatizată în cloud a aplicațiilor și sistemelor informatice clasice cloudifier.net
Beneficiar	CLOUDIFIER SRL
Cod MySMIS	104349
Nr. iregistrare	P_38_543
Director Proiect	Andrei Ionut DAMIAN
Activitate conform planului de proiect	1. Activităţi de cercetare-dezvoltare (cercetare industrială şi/sau dezvoltare experimentală) - 1.1 State-of-the-art
Echipa de cercetare-dezvoltare	Andrei Ionut DAMIAN Octavian BULIE
Livrabil	Raport State-of-the-Art / 74 pag

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

Contents

1 Abstract

In prezentul document este descris rezultatul etapei de cercetare industriala aferenta Activitatii 1 de cercetare industrială şi/sau dezvoltare experimentală, sub-activitatea “State-of-the-Art”. Cercetarea industriala realizata in aceasta etapa a constat in revizuirea stadiului actual al tehnicii/tehnologiilor atat in domeniul Cloud Computing / virtual desktop dar mai ales in domeniul recunoasterii si segmentarii semantice a imaginilor prin algoritmi avansati de inteligenta artificiala, domeniu de cercetare ce a luat o deosebita amploare in ultimile 12 luni.

Scopul analizei, recunoasterii, segmentarii semantice si a constructiei unei harti a imaginilor este de incorporare a acestor tehnologii avansate de inteligenta artificiala (Machine Learning) in motorul de migrare automatizata a aplicatiilor in mediul cloud. Intuitiv acest mecanism functioneaza similar unui actor/observator uman care analizeaza o imagine dupa care reproduce intr-un alt mediu din memorie elementele principale ale imaginii respective impreuna cu pozitia, forma si functionalitate lor in contextul analizat initial.

2 Analiza contextului cercetarii

2.1 Acronime

Machine Learning = domeniu de cercetare-dezvoltare din cadrul Inteligentei Artificiale, aflat la fundamentul acesteia, prin care un sistem computerizat este capabil sa invete automatizat din propriile greseli cu sau fara supervizare sau orice forma de factor uman

Desktop computing = mod de lucru in care aplicatiile utilizate sunt rulate exclusiv pe o statie de lucru sau laptop fara a exista acces la o baza de date centralizata sau la sisteme online

Web- based computing = mod de lucru in care aplicatiile sunt rulate in cadrul unui server de web iar utilizatorii acceseaza functionalitatile respective prin intermediul browser-elor de internet

Legacy = Aplicatie sau modul dezvoltat in tehnologii invechite si/sau perimate care ruleaza in mediu de tip desktop computing

Client-Server = mod de lucru cu masive de date prin care acestea sunt centralizate pe un server de baze de date relationale

Segmentare semantica = analiza unei multimi, imagini, etc si identificarea elementelor componente ale acesteia

Cloud computing = tehnologie si mediu de rulare a sistemelor informatice prin care toate datele si functionalitatile aplicatiilor se acceseaza prin intermediului internetului si a aplicatiilor de tip web-based

Bot = aplicatie care simuleaza un interlocutor cu care utilizatorul comunica in limbaj natural (roBot) pentru rezolvarea unui set de probleme analizate cu ajutorul algortmilor de tip Machine Learning

Python = limbaj de nivel inalt construit in special pentru mediul stiintific ce include capabilitati avansate pentru dezvoltare atat la nivel de experimentare cat si la nivel de productie

2.2 Obiectivele proiectului

In conformitate cu Cererea de finantare aferenta contractului de finantare nr 98/09.09.2016 obiectivul proiectului „PLATFORMA DE MIGRARE AUTOMATIZATA IN CLOUD A APLICATIILOR SI SISTEMELOR INFORMATICE CLASICE- Cloudifier.NET" este cercetarea, dezvoltarea si punerea in functiune in mediul comercial a produsului platforma inovativ Cloudifier.NET, ce se adreseaza domeniului tehnologiilor informatiei si comunicatiilor. In cadrul acestui obiectiv mentionam si intentia de diseminare publica partiala a rezultatelor proiectului sub licenta European Public License.

Obiectivul proiectului "PLATFORMA DE MIGRARE AUTOMATIZATA IN CLOUD A APLICATIILOR SI SISTEMELOR INFORMATICE CLASICE - Cloudifier.NET" raspunde uneia dintre prioritatile stabilite de Comisia Europeană prin Agenda Digitala 2020 si anume stimularea si facilitarea comertului electronic la nivelul tarilor membre. Acest proiect va contribui major la sporirea utilizarii, calitatii si a accesului la tehnologia informatiei si comunicatiilor atat la nivelul local al Romaniei cat si la nivelul comunitatii de viitori potentiali utilizatori din intreaga Uniune Europeana iar implementarea cu succes a platformei va duce la sporirea contributiei sectorului TIC pentru competitivitatea economica a Romaniei.
De asemenea, proiectul are ca obiectiv integrarea federalizarii cu conceptul de virtual desktop online, oferind astfel utilizatorului posibilitatea de a integra toate aplicatiile pe care le utilizeaza in mediul online – aplicatii de tip Cloud– intr-un singur spatiu virtual, in care sa dispuna de servicii de securizare si confidentialitate avansata a informatiilor.

La finalizarea implementarii produsul final va functiona ca o platforma de tip desktop online, bazata pe tehnologii de tip Cloud Computing, cu functionalitati multi-scop, care va

permite sustinerea proiectelor inovative din domeniul tehnologiei informatiilor si comunicatiilor si va facilita dezvoltarea sustenabila a acestora.

2.3 Contextul proiectului

Produsul Cloudifier.NET revendica prin unicitatea conceptului sau o serie de functionalitati total inovative, ce nu se regasesc la produsele comerciale deja existente in piata. In procesul de analiza preliminara a stadiului actual al tehnologiei (state-of-the-art), s-au luat in considerare o serie de produse comerciale relativ similare, pentru a se face o paralela intre acestea si Cloudifier.NET si a se determina detaliile avansului tehnologic dincolo de stadiul actual al tehnologiei (advances beyond state-of-the-art).

Implementarea proiectului „Platforma de migrare automatizata in Cloud a aplicatiilor si sistemelor informatice clasice- Cloudifier.NET" va permite:

- Sustinerea proiectelor inovative din domeniul tehnologiei informatiilor si comunicatiilor;
- Va facilita dezvoltarea sustenabila a acestora, prin oferirea accesului la o platforma de tip comunitate, in care sa poata gasi si regasi aplicatii si sisteme la cerere, utilizand astfel exclusiv structura de costuri de tip OPEX, fata de structurile clasice de tip CAPEX;
- Sprijinirea companiilor mari prin servicii "cost-effective" pentru migrarea de la aplicatii legacy- (aplicatii dezvoltate prin metode clasice de programare si implementare de tip desktop sau client-server, ce utilizeaza resurse bazate pe cheltuieli de capital, cum ar fi echipamente de calcul locale, licente locale, s.a.m.d - spre deosebire de modelul de aplicatii bazat pe tehnologia Cloud Computing care utilizeaza resurse la cerere scalabile, elastice si bazate aproape exclusiv pe modelul de cheltuieli operationale OPEX)" - la aplicatii in Cloud;
- Sprijinirea atat a mediului IMM, cat si a utilizatorilor privati, in vederea accesului la un mediu de tip spatiu virtual, personal de lucru online, in continua dezvoltare.

Unificarea avansului tehnologic adus de subplatforma avansata de translatare bazata de Machine Learning a aplicatiilor clasice in medii de tip Cloud cu avansul tehnologic propus de subplatforma de federalizare, brokeraj de date si spatiu virtual de lucru (virtual desktop), face ca platforma sa ofere o arie de inovare deosebit de generoasa, acoperind mai multe nevoi orizontale in domeniul tehnologiilor informatiilor si comunicatiilor, ce au impact asupra unor arii multiple tehnologice si variate industrii.

Cele doua subsisteme principale ale platformei Cloudfier sunt:

A. subsistemul bazat pe Machine Learning de translatare inteligenta automatizata a aplicatiilor clasice desktop in aplicatii online in mediu de tip Cloud computing cu accent pe trecerea de la modele de sisteme informatice bazate pe CAPEX la modele de sisteme informatice bazate pe OPEX;
B. subsistemul de federalizare a aplicatiilor de tip Cloud Computing provenite din surse multiple in vederea realizarii unei platforme de tip spatiu de lucru personal virtual online (online personal virtual desktop).

In concluzie directiile principale de utilitate ale platformei propuse sunt axate pe trei mari categorii distribuite in doua zone de inovatie dupa cum urmeaza:

1. Inovare in domeniu federalizarii platformelor, migrarii datelor si spatiilor personale virtuale:
   1.1. broker de servicii de Cloud pe care se inregistreaza furnizorii de servicii noi si inovative de Cloud;
   1.2. agregator si federalizator de servicii de Cloud prestandardizate
2. Inovare in domeniul migrarii aplicatiilor construite pe principiile clasice ale sistemelor informatice catre noile paradigme tehnologice definite de Cloud Computing:
   2.1. sistem inteligent bazat pe tehnici avansate de Machine Learning destinat traducerii applicatiilor clasice desktop sau aplicatiilor client-server in aplicatii de tip Cloud Computing;
   2.2. provizionarea automatizata a aplicatiilor translatate in mediul platformei inovative

2.4 Sumarul activitatii de cercetare si rezultatele

Activitatea de cercetare a stadiului actual al tehnologiei s-a axact atat pe o serie de proiecte depuse in cadrul apelurilor de proiecte de cercetare beyond-state-of-the-art din cadrul Horizon 2020 dar mai ales pe lucrari publicate in cele mai renumite jurnale stiintifice cum ar fi:

- ✓ Journal of Machine Learning Research, www.jmlr.org (ISSN 1533-7928)
- ✓ International Journal of Neural Systems, ISSN:0129-0657
- ✓ IEEE Transactions on Pattern Analysis and Machine Intelligence, ISSN:0162-8828
- ✓ IEEE Transactions on Neural Networks and Learning Systems, ISSN:2162-237X
- ✓ Machine Learning, ISSN:0885-612
- ✓ International Journal of Intelligent Systems, ISSN:0884-8173
- ✓ Expert Systems with Applications, ISSN:0957-4174

De mentionat este faptul ca au fost analizate in mod particular lucrari ale celor mai cunoscuti cercetatori din domeniul inteligentei artificiale ca:

- ✓ Yoshua Bengio: http://www.iro.umontreal.ca/~ben...
- ✓ Geoffrey Hinton: http://www.cs.toronto.edu/~hinton/
- ✓ Alex Smola: http://alex.smola.org/
- ✓ Andrew Ng: http://ai.stanford.edu/~ang/
- ✓ Alex Krizhevsky: http://www.cs.toronto.edu/~kriz/
- ✓ Ilya Sutskever: http://www.cs.toronto.edu/~ilya/
- ✓ Andrej Karpathy: http://cs.stanford.edu/people/karpathy/
- ✓ Francois Chollet: https://blog.keras.io/author/francois-chollet.html
- ✓ Chris Meek: http://research.microsoft.com/en...
- ✓ Hugo Larochelle: http://www.dmi.usherb.ca/~larocheh/index_en.html
- ✓ Ian Goodfellow: https://en.wikipedia.org/wiki/Ian_Goodfellow
- ✓ Aaron Courville: https://aaroncourville.wordpress.com/
- ✓ Leo Breiman: http://www.stat.berkeley.edu/~br...
- ✓ Andrew McCallum: http://www.cs.umass.edu/~mccallum/
- ✓ Chris Meek: http://research.microsoft.com/en...
- ✓ Trevor Darrell: https://people.eecs.berkeley.edu/~trevor/

- ✓ Jonathan Long: http://people.eecs.berkeley.edu/~jonlong/

Din lucrarile analizate au fost extrase si analizate elemente cum ar fi:

1. Modelele de reconstructie arhitecturala si regenerare cod sursa
   1.1. Modele si metode de analiza a segmentarii ecranelor din cadrul aplicatiilor legacy bazate pe algoritmi clasici interativi si modele de Machine Learning fara retele neurale adanci
   1.2. Modele si metode de segmentare semantica cu ajutorul retelelor neurale adanci convolutionale a componentelor din cadrul ecranelor aplicatiilor legacy

2. Modelele de migrare aplicatie
   2.1. Utilizarea retelelor neurale recurente si in particular a LSTM (Long Short Term Memory) pentru realizarea de sisteme expert de tip bot care sa fie capabile de a asista utilizatorul in procesul de dezasamblare in subcomponente a aplicatiilor legacy si reasamblarea automatizata a acestora in medii de tip Cloud computing

Deasemenea in vederea atingerii tuturor elementelor de analiza propuse au fost analizate tehnologii si abordari pentru probleme de inginerie si dezvoltare experimentala cu complexitate stiintifica mai redusa cum ar fi:

3. Modelele de analiza ale tehnologiei de tip baza de date utilizata in aplicatia legacy ce urmeaza a fi migrata
   3.1. Baze de date flat-file
   3.2. Baze de date relationale nebazate pe tehnologii client-server
4. Modelele de analiza ale entitatilor de date
   4.1. Analiza bazata pe parcurgerea si recunoasterea grafurilor
5. Modelele de analiza ale relationarii intre entitatile de date
   5.1. Analiza bazata pe parcurgerea si recunoasterea grafurilor
6. Modelele de recunoastere a tehnologiei componentelor/modulelor sistemului
   6.1. Analiza bazata pe citirea componentelor sistemului de operare gazda
7. Modelele de recunoasterele ale layere-lor de comunicare/API/etc.
   7.1. Analiza bazata pe citirea componentelor sistemului de operare gazda

2.5 Alte remarci

Directorul de proiect intentioneaza ca in baza cercetarii realizate in primul trimestru de implementare a proiectului “Platforma de migrare automatizată în cloud a aplicațiilor și sistemelor informatice clasice Cloudifier.net”, MySMIS: 104349, nr.: P_38_543, sa pregateasca si sa publice o lucrare stiintifica intr-o publicatie stiintifica cu recunoastere internationala.

3 Rezultatele cercetarii

3.1 Problematica concreta analizata

Scopul analizei stadiului actual al tehnologiei este acela de a determina metodele cele mai moderne/actuale de realizare a predictiilor/inferentelor in imagistica – in particular in cazul proiectului CLOUDIFIER referindu-ne la analiza imaginilor captate in timp real in timpul functionarii aplicatiilor si implicit analiza automatizata cu ajutorul recunoasterii avansate de forme/imagini a aplicatiilor “legacy” in vederea transalatarii acestora automatizate.

In decursul lunii octombrie 2016 au fost analizate cele mai recente si avansate lucrari de cercetare fundamentala si industriala provenite de la cele mai prestigioase institute si universitati printre care enumeram:

- ✓ Caltech – California Institute for Technology
- ✓ MIT – Massachusetts Institute for Technology
- ✓ Stanford
- ✓ University of Toronto
- ✓ Harvard
- ✓ University of Washington

Principalele zone analizate au fost:

- ✓ Metodele de tip Deep Learning bazate pe Retele Neuronale Convolutionale – Deep Convolutional Neural Networks
- ✓ Metode de tip shallow learning pentru invatarea supervizata a structurilor si a elementelor de imagistica utilizand modele de invatare in timp real (online learning)
- ✓ Cele mai moderne abordari in Deep Learning – Tensor Flow
- ✓ Cele mai moderne abordari in shallow learning – Extreme Boosted Decision Trees / Random Forests - XGBoost

In decursul lunii noiembrie 2016 au fost analizate cele mai recente si avansate lucrari de cercetare fundamentala in vederea determinarii unui set de algoritmi de Machine Learning ideali pentru identificarea primitivelor de interfata grafica (butoane, campuri, ferestre, texte statice, etc) si a pozitiei acestora in cadrul ecranelor interfetelor grafice

- ✓ Determinarea unui algoritm de tip Machine Learning pentru generarea AUTOMATA de interfe grafice si cod sursa aferente pe baza schitelor facute manual pe suport de hartie, tabla, etc
- ✓ Analiza TensorFlow
- ✓ Analiza XGBoost
- ✓ Analiza metode si propuneri pentru biblioteci interne
- ✓ Analiza si testarea experimentala a mediilor de procesare numerica masiv paralela cu ajutorul GPU (tehnologiile bazate pe nuclee de calul masiv paralel CUDA)

In decursul luni decembrie 2016 a fost continuat procesul de analiza a stadiului curent al tehnologiei in domeniul sistemelor de tip Machine Learning cu accent pe zona de Deep Learning si in particular a sistemelor de analiza si recunoastere bazata pe inteligenta artificiala a imaginilor. In decursul acestei luni analiza stadiului curent al cercetarii fost fost axat in principal pe lucrarea stiintifica publicata recent de J. Long et al “Fully Convolutional Networks for Semantic Segmentation”, lucrare considerata actualmente state-of-the-art in ceea priveste metodele de recunoastere si segmentare a componentelor in cadrul imaginilor. Pentru referinta prezentam anexat un scurt rezumat in limba engleza a lucrarii de referinta.

Principalele puncte pe care le urmarim in cercetare sunt urmatoarele:

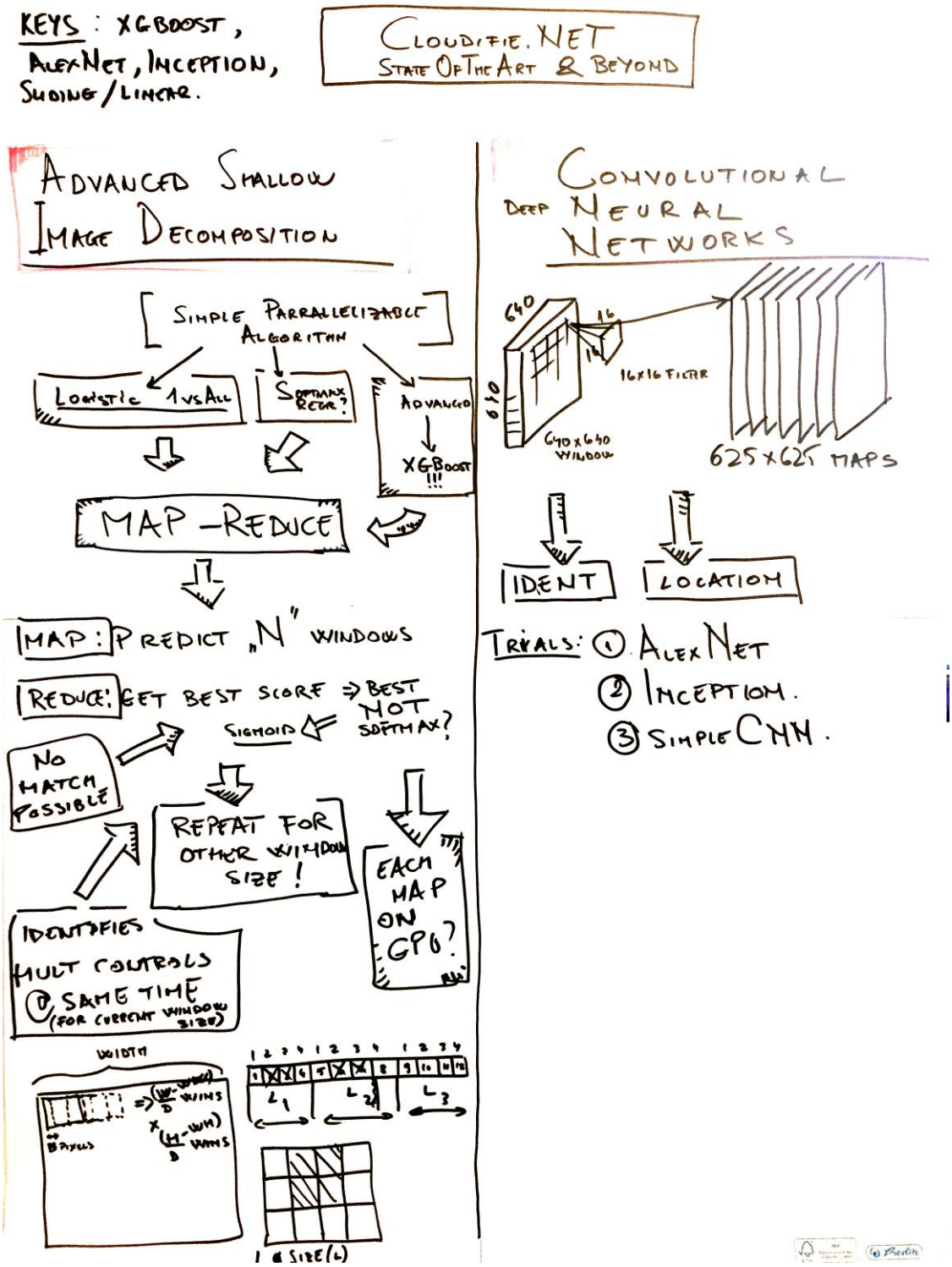
1. Determinarea metodelor optime bazate pe Deep Learning pentru recunoasterea si segmentarea (identificarea locatiei spatiale) a elementelor de interfata grafica pe care Cloudifier.NET va trebuie sa le translateze automatizat din aplicatiile legacy in aplicatiile din mediul cloud computing.
2. Aplicarea de metode simple bazate pe algoritmi de machine learning superficiali (regresie logistica, arbori de decizie, clasificare naiva bazata pe teorema lui Bayes, clusterizare cu analiza distantelor euclidiene) precum si metode de segmentare

iterativa a imaginilor analizate cum ar fi metoda ferestrelor deplasate continuu (“ferestre alunecatoare” sau sliding-windows algorithm)

In Imaginea nr. 1 (“Schita de principiu a abordarii stadiului actual al tehnologiei si avansurile aferente”) este prezentata abordarea celor doua metode care vor fi proiectate in paralel in procesul de proiectare a modelelor arhitecturale ale etapei 1.2 din activitatile proiectului si ulterior experimentate in cadrul procesului de dezvoltare experimentala. Detaliile acestei abordari vor fi prezentate in sectiunea “3.3 Doua directii de abordare avansata a cercetarii”

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

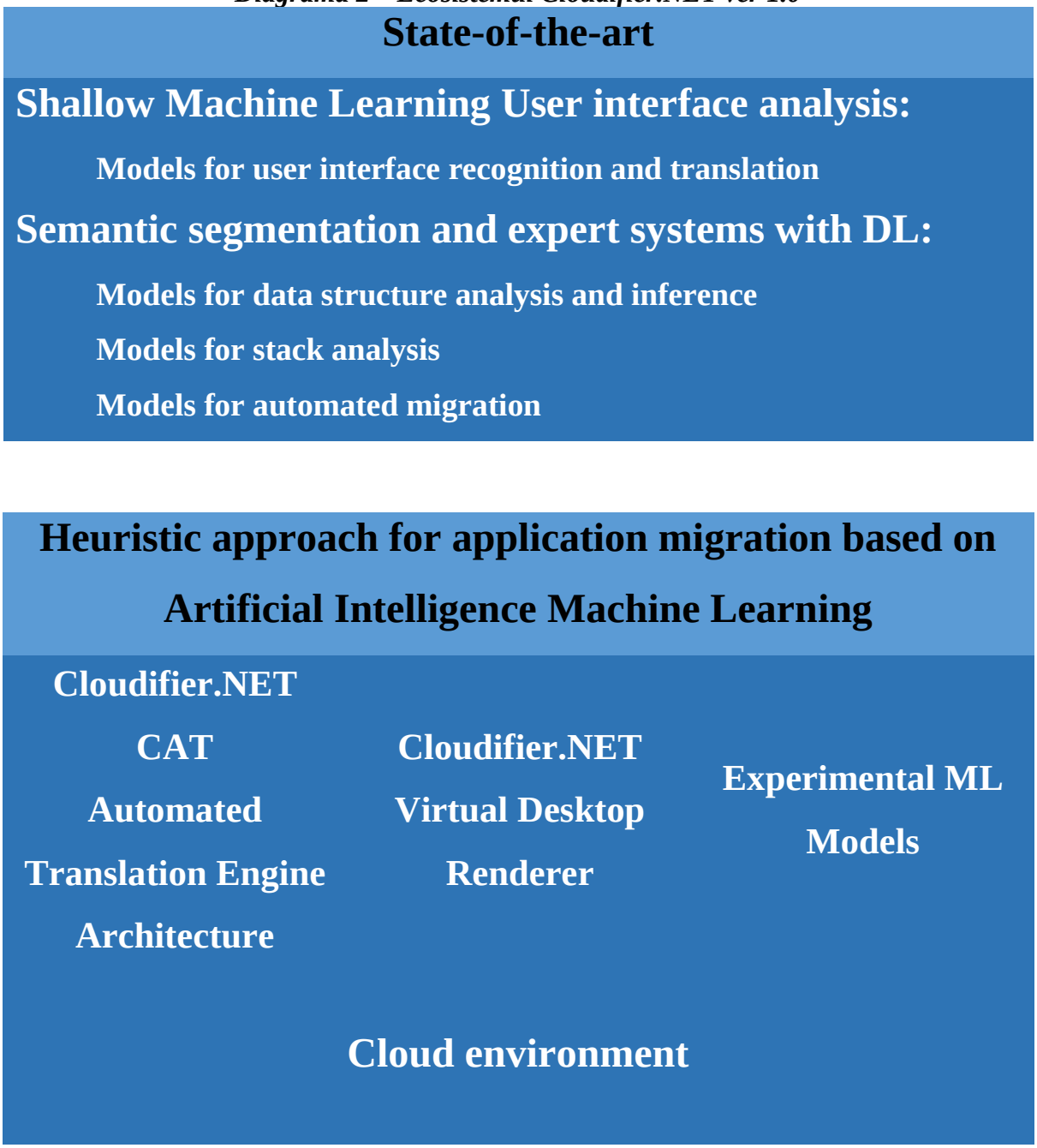
Imaginea 1 "Schita de principiu a abordarii stadiului actual al tehnologiei si avansurile aferente"



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

3.2 Ecosistemul Cloudifier.NET

Diagrama 2 – Ecosistemul Cloudifier.NET ver 1.0



Ecosistemul Cloudifier.NET descris in Diagrama 2 cuprinde toate elementele descrise in documentatia Cererii de Finantare si proiectului tehnic aferente proiectului "Platforma de migrare automatizată în cloud a aplicațiilor și sistemelor informatice clasice cloudifier.net" al

CLOUDIFIER SRL cu codul MySMIS 2014 nr 104349 si numarul de inregistrare online P_38_543. La elementele initiale au fost adaugate si vor fi detaliate pe parcursul activitatilor de cercetare-dezvoltare elementele principale bazate pe tehnologii de Machine Learning.

4 Doua directii de abordare avansata a cercetarii

Conform diagramei descrise anterior si a imaginii 1 ce descrie schita de principiu a abordarii proiectului, in urma analizei stadiului actual al tehnologiei au fost determinate doua directii principale in care se vor desfasura activitati de modelare arhitecturala si dezvoltare experiementala in urmatoarea perioada:

1. Modele avansate de recunoastere si decompoziei a cadrelor (imaginilor) din aplicatiile legacy prin utilizarea de algoritmi paralelizabili de predictie bazata pe tehnici de machine learning fara retele neurale adanci
2. Modele de segmentare semantica precum si proiectare de sisteme automatizate expert de tip “Bot” cu ajutorul retelelor neurale adanci convolutionale si recurente (LSTM)

4.1 Tehnici bazate pe modele avansate Shallow Machine Learning

4.1.1 Paralelizarea masiva a modelelor superfiale de invatare

Prin utilizarea tehnicilor de ultima generatie si in particular a tehnologiei state-of-the-art PASCAL creeata si lansata in 2016 de NVidia se vor utiliza structuri de calcul masiv paralel cu peste 1500 de nuclee de calcul numeric paralelizat.

Aceasta abordare va permite aplicarea de algoritmi de invatare a modelelor superficiale Machine Learning pe structuri de date cu complexitate si dimensionalitate foarte mare – spre exemplu imagini cu adancime de culoare pe 32 de biti (4 octeti) si rezolutii reale de peste W:2000 H:2000. Pentru cazul particular al proiectului Cloudifier.NET este necesara analiza in timp real a imaginilor cu rezolutii inalte peste 1080x786x4 ceea ce genereaza un minim de 829,000 de dimensiuni (variabile predictor) . Astfel se vor putea aplica algoritmi de tip regresie logistica softmax antrenata prin mini-calupuri informationale pe structura de calcul masiv paralel.

Detaliile modului de aplicare a algoritmilor vor fi definite in etapa dezvoltarii subactivitatii 1.2 de proiectare a modelelor arhitecturale

Print utilizarea celor mai recente cercetarii in domeniul Machine Learning pentru ansamble de modele se urmareste testarea experimentala a potentialei utilizari de modele simple si eficiente pentru toate componentele de inteligenta artificiala ale proiectului Cloudifier.NET. Aceste componente vor fi asociate cu algoritmi clasici de tip “Sliding Windows” – utilizati actualmente de camerele de luat vederi de ultima generatie - care vor genera mecanic segmentele ce urmeaza sa fie analizate de algoritmii de tip Machine Learning.

In cele ce urmeaza este prezentat un test realizat cu ajutorul tehnologiei CUDA a NVidia pentru calcul masiv paralel utilizand limbajul de nivel inalt Python

```
1 # Exercise 1 from http://webapp.dam.brown.edu/wiki/SciComp/CudaExercises
2
3 # Transposition of a matrix
```

```
# by Hendrik Riedmann <riedmann@dam.brown.edu>

from   future   import division, print function

import pycuda.driver as cuda
import pycuda.gpuarray as gpuarray
import pycuda.autoinit
from pycuda.compiler import SourceModule

import numpy
import numpy.linalg as la

from pycuda.tools import context dependent memoize

block size = 16

@context_dependent_memoize
def _get_transpose_kernel():
    mod = SourceModule("""
    #define BLOCK SIZE %(block size)d
    #define A BLOCK STRIDE (BLOCK SIZE * a width)
    #define A T BLOCK STRIDE (BLOCK SIZE * a height)

    __global__ void transpose(float *A_t, float *A, int a_width, int a_height)
    {
        // Base indices in A and A t
        int base idx a   = blockIdx.x * BLOCK SIZE +
    blockIdx.y * A_BLOCK_STRIDE;
        int base_idx_a_t = blockIdx.y * BLOCK_SIZE +
    blockIdx.x * A T BLOCK STRIDE;

        // Global indices in A and A t
        int glob idx a   = base idx a + threadIdx.x + a width * threadIdx.y;
        int glob_idx_a_t = base_idx_a_t + threadIdx.x + a_height * threadIdx.y;

          shared   float A shared[BLOCK SIZE][BLOCK SIZE+1];

        // Store transposed submatrix to shared memory
        A_shared[threadIdx.y][threadIdx.x] = A[glob_idx_a];

        __syncthreads();

        // Write transposed submatrix to global memory
        A t[glob idx a t] = A shared[threadIdx.x][threadIdx.y];
    }
    """% {"block_size": block_size})

    func = mod.get function("transpose")
    func.prepare("PPii")

    from pytools import Record
    class TransposeKernelInfo(Record): pass

    return TransposeKernelInfo(func=func,
            block=(block size, block size, 1),
            block_size=block_size,
            granularity=block size)


def _get_big_block_transpose_kernel():
    mod = SourceModule("""
    #define BLOCK SIZE %(block size)d
    #define A BLOCK STRIDE (BLOCK SIZE * a width)
    #define A T BLOCK STRIDE (BLOCK SIZE * a height)

    __global__ void transpose(float *A, float *A_t, int a_width, int a_height)
    {
        // Base indices in A and A t
        int base idx a   = 2 * blockIdx.x * BLOCK SIZE +
    2 * blockIdx.y * A_BLOCK_STRIDE;
```


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

```
        int base_idx_a_t = 2 * blockIdx.y * BLOCK_SIZE +
    2 * blockIdx.x * A_T_BLOCK_STRIDE;

        // Global indices in A and A_t
        int glob_idx_a   = base_idx_a + threadIdx.x + a_width * threadIdx.y;
        int glob_idx_a_t = base_idx_a_t + threadIdx.x + a_height * threadIdx.y;

          shared   float A_shared[2 * BLOCK_SIZE][2 * BLOCK_SIZE + 1];

        // Store transposed submatrix to shared memory
        A_shared[threadIdx.y][threadIdx.x] = A[glob_idx_a];
        A_shared[threadIdx.y][threadIdx.x + BLOCK_SIZE] =
    A[glob_idx_a + A_BLOCK_STRIDE];
        A_shared[threadIdx.y + BLOCK_SIZE][threadIdx.x] =
    A[glob_idx_a + BLOCK_SIZE];
        A_shared[threadIdx.y + BLOCK_SIZE][threadIdx.x + BLOCK_SIZE] =
        A[glob_idx_a + BLOCK_SIZE + A_BLOCK_STRIDE];

          syncthreads();

        // Write transposed submatrix to global memory
        A_t[glob_idx_a_t] = A_shared[threadIdx.x][threadIdx.y];
        A_t[glob_idx_a_t + A_T_BLOCK_STRIDE] =
    A_shared[threadIdx.x + BLOCK_SIZE][threadIdx.y];
        A_t[glob_idx_a_t + BLOCK_SIZE] =
    A_shared[threadIdx.x][threadIdx.y + BLOCK_SIZE];
        A_t[glob_idx_a_t + A_T_BLOCK_STRIDE + BLOCK_SIZE] =
    A_shared[threadIdx.x + BLOCK_SIZE][threadIdx.y + BLOCK_SIZE];
    }
      """% {"block_size": block_size})

    func = mod.get_function("transpose")
    func.prepare("PPii")

    from pytools import Record
    class TransposeKernelInfo(Record): pass

    return TransposeKernelInfo(func=func,
            block=(block_size, block_size, 1),
            block_size=block_size,
            granularity=2*block_size)




def _transpose(tgt, src):
    krnl = _get_transpose_kernel()

    w, h = src.shape
    assert tgt.shape == (h, w)
    assert w % krnl.granularity == 0
    assert h % krnl.granularity == 0

    krnl.func.prepared_call(
                    (w // krnl.granularity, h // krnl.granularity), krnl.block,
                    tgt.gpudata, src.gpudata, w, h)




def transpose(src):
    w, h = src.shape

    result = gpuarray.empty((h, w), dtype=src.dtype)
     transpose(result, src)
    return result
```


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

```
146 def check transpose():
147     from pycuda.curandom import rand
148 
149     for i in numpy.arange(10, 13, 0.125):
150         size = int(((2**i) // 32) * 32)
151         print(size)
152 
153         source = rand((size, size), dtype=numpy.float32)
154 
155         result = transpose(source)
156 
157         err = source.get().T - result.get()
158         err norm = la.norm(err)
159 
160         source.gpudata.free()
161         result.gpudata.free()
162 
163         assert err_norm == 0, (size, err_norm)
164 
165 
166 
167 
168 def run_benchmark():
169     from pycuda.curandom import rand
170 
171     powers = numpy.arange(10, 13, 2**(-6))
172     sizes = [int(size) for size in numpy.unique(2**powers // 16 * 16)]
173     bandwidths = []
174     times = []
175 
176     for size in sizes:
177 
178         source = rand((size, size), dtype=numpy.float32)
179         target = gpuarray.empty((size, size), dtype=source.dtype)
180 
181         start = pycuda.driver.Event()
182         stop = pycuda.driver.Event()
183 
184         warmup = 2
185 
186         for i in range(warmup):
187              transpose(target, source)
188 
189         count = 10
190 
191         cuda.Context.synchronize()
192         start.record()
193 
194         for i in range(count):
195             _transpose(target, source)
196 
197         stop.record()
198         stop.synchronize()
199 
200         elapsed seconds = stop.time since(start)*1e-3
201         mem_bw = source.nbytes / elapsed_seconds * 2 * count
202 
203         bandwidths.append(mem bw)
204         times.append(elapsed seconds)
205 
206     slow_sizes = [s for s, bw in zip(sizes, bandwidths) if bw < 40e9]
207     print("Sizes for which bandwidth was low:", slow_sizes)
208     print("Ditto, mod 64:", [s % 64 for s in slow sizes])
209     from matplotlib.pyplot import semilogx, loglog, show, savefig, clf, xlabel, ylabel
210     xlabel('matrix size')
211     ylabel('bandwidth')
212     semilogx(sizes, bandwidths)
213     savefig("transpose-bw.png")
214     clf()
215     xlabel('matrix size')
216     ylabel('time')
```


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

```
217     loglog(sizes, times)
218     savefig("transpose-times.png")
219
220
221
222
223 #check transpose()
224 run_benchmark()
```

4.1.2 Framework-ul state-of-the-art XGBoost

In vederea aplicarii celor mai recente cercetari in domeniile vizate a fost selectat printre modelele state-of-the-art si framework-ul XGBoost dezvoltat in perioada 2015-2016, framework ce a fost popularizat in perioada desfasurata dupa depunerea aplicatiei initiale a proiectului Cloudifier si actualmente este considerat state-of-the-art in domeniul modelelor de tip ansamblu de algoritmi de inteligenta artificala destinati predictiei.

In continuare prezentam testele realizate pe aceasta tehnologie pe parcursul desfasurarii activitatilor 1.1 – Analiza State-of-the-Art

```
"""

Created on: 10/27/2016

Last Modified: 12/23/2016


@author: Andrei Ionut DAMIAN

"""



# Tune learning_rate

from numpy import loadtxt

from xgboost import XGBClassifier

from sklearn.grid_search import GridSearchCV

from sklearn.cross_validation import StratifiedKFold
```


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

```
from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble import VotingClassifier

from sklearn.linear_model import LogisticRegression

from sklearn.svm import SVC

from sklearn.neighbors import KNeighborsClassifier

from sklearn.model_selection import GridSearchCV

from xgboost import XGBClassifier

from xgboost import XGBRegressor

import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score

from sklearn.metrics import mean_absolute_error

from sklearn.metrics import make_scorer

from sklearn.preprocessing import StandardScaler

import sys

import re


import time


def copy_from_combined(train_ds, test_ds, combined_ds, columns, index_pos):

    for column in columns:

        train_ds.set_value(train_ds.index, column, combined_ds[:index_pos][column].values)

        test_ds.set_value(test_ds.index, column, combined_ds[index_pos:][column].values)



if __name__ == "__main__":


    Verbose = False
```


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

```
init_train_data =  pd.read_csv('train.csv')

init_test_data =  pd.read_csv('test.csv')




train_data = init_train_data.copy()

test_data = init_test_data.copy()


sex_values={'male':1,'female':0}


train_data['Sex']=train_data['Sex'].replace(sex_values)


test_data['Sex']=test_data['Sex'].replace(sex_values)


train_data.set_value(train_data.Embarked.isnull(), "Embarked", "C")

test_data.set_value(test_data.Fare.isnull(), 'Fare', 8.05) # the 60 year old guy


train_data.set_value(train_data.Cabin.isnull(), 'Cabin', 'U0')

test_data.set_value(test_data.Cabin.isnull(), 'Cabin', 'U0')


###

### feature engineering

### all features will begin with EF_ prefix

###

# create nr of names feature

train_names = train_data.Name.map(lambda x: len(re.split(' ', x)))

train_data.set_value(train_data.index, 'EF_Names', train_names)

del train_names
```


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

```
test_names = test_data.Name.map(lambda x: len(re.split(' ', x)))

test_data.set_value(test_data.index, 'EF_Names', test_names)

del test_names



#create title feature (later will be factorized/O.H.E.-ed)

prog = re.compile(', (.*?)\.')


title = train_data.Name.map(lambda x: prog.findall(x)[0])

title[title=='Mme'] = 'Mrs'

title[title.isin(['Ms','Mlle'])] = 'Miss'

title[title.isin(['Don', 'Jonkheer'])] = 'Sir'

title[title.isin(['Dona', 'Lady', 'the Countess'])] = 'Lady'

title[title.isin(['Capt', 'Col', 'Major', 'Dr', 'Officer', 'Rev'])] = 'Officer'

_ = train_data.set_value(train_data.index, 'EF_Title', title)

del title


title = test_data.Name.map(lambda x: prog.findall(x)[0])

title[title=='Mme'] = 'Mrs'

title[title.isin(['Ms','Mlle'])] = 'Miss'

title[title.isin(['Don', 'Jonkheer'])] = 'Sir'

title[title.isin(['Dona', 'Lady', 'the Countess'])] = 'Lady'

title[title.isin(['Capt', 'Col', 'Major', 'Dr', 'Officer', 'Rev'])] = 'Officer'

_ = test_data.set_value(test_data.index, 'EF_Title', title)

del title



##

## from now on we will combine both TRAIN and TEST in order to computer

## accurate averages/predictions for certain variables and engineered
```


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

```
    ## predictors that we will construct
    ##


    combined = pd.concat([train_data, test_data], ignore_index=True)
    ## Generate "Deck" feature - factorization of Cabin 1st letter
    deck = combined[~combined.Cabin.isnull()].Cabin.map( lambda x : re.compile("([a-zA-
Z]+)").search(x).group())
    deck = pd.factorize(deck)[0]
    combined.set_value(combined.index, 'EF_Deck', deck)
    del deck


    ## Extract "Room" feature - the number contained in the cabin
    ## info. Standardize the result
    checker = re.compile("([0-9]+)")
    def roomNum(x):
        nums = checker.search(x)
        if nums:
            return int(nums.group())+1
        else:
            return 1
    rooms = combined.Cabin.map(roomNum)
    _ = combined.set_value(combined.index, 'EF_Room', rooms)
    combined['EF_Room'] = (combined.EF_Room-combined.EF_Room.min())/combined.EF_Room.max()
    del checker, roomNum


    ## now compute Group size
    combined['EF_Group_size'] = combined.Parch + combined.SibSp + 1
    combined['EF_Group_type'] = pd.Series('M', index=combined.index)
    combined.set_value(combined.EF_Group_size >4, 'EF_Group_type', 'L')
    combined.set_value(combined.EF_Group_size==1, 'EF_Group_type', 'S')
```


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

```
    ##


    # save work in train/test
    saved_cols = ['EF_Deck', 'EF_Room', 'EF_Group_type', 'EF_Group_size',
                  'EF_Names', 'EF_Title']
    copy_from_combined(train_data, test_data, combined, saved_cols, 891 )
    # done saving


    # transform 'Embarked', 'Sex', 'EF_Title', 'EF_Group_type' in OH features
    # normalize Fare
    scaler = StandardScaler()
    combined['EF_Fare']       =       pd.Series(scaler.fit_transform(combined.Fare.reshape(-
1,1)).reshape(-1),
                                      index=combined.index)
    ## now lets drop some of standard predictors that have been
    ## re-engineered


    combined.drop(labels=['PassengerId', 'Name', 'Cabin', 'Survived', 'Ticket', 'Fare'],
                  axis=1, inplace=True)
    ## now lets encode as planned
    combined = pd.get_dummies(combined,
                              columns=['Embarked', 'Sex', 'EF_Title', 'EF_Group_type'])


    # now FINALLY predict AGE
    X_train_age = combined[~combined.Age.isnull()].drop('Age', axis=1)
    y_train_age = combined[~combined.Age.isnull()].Age
    xgb_regressor = XGBRegressor(max_depth=4)
    regr_scoring = make_scorer(mean_absolute_error, greater_is_better=False)
    regr_parameters = {'reg_alpha':np.linspace(0.1,1.0,5),
                       'reg_lambda': np.linspace(1.0,3.0,5)}
```


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

```
    tmp_rgr = GridSearchCV(xgb_regressor,
                           param_grid = regr_parameters,
                           scoring = regr_scoring,
                           n_jobs=6)

    tmp_rgr.fit(X_train_age, y_train_age)
    reg_xgb = tmp_rgr.best_estimator_

    age_preds = reg_xgb.predict(combined[combined.Age.isnull()].drop('Age', axis=1))
    combined.set_value(combined.Age.isnull(), 'Age', age_preds)
    combined['EF_NorAge']      =      pd.Series(scaler.fit_transform(combined.Age.reshape(-
1,1)).reshape(-1),
                                    index=combined.index)
    combined['EF_NorNames']   =   pd.Series(scaler.fit_transform(combined.EF_Names.reshape(-
1,1)).reshape(-1),
                                      index=combined.index)
    combined['EF_Group_size']                                                          =
pd.Series(scaler.fit_transform(combined.EF_Group_size.reshape(-1,1)).reshape(-1),
                                      index=combined.index)

    # now save processed engineered features
    saved_cols = ['EF_NorAge', 'EF_NorNames', 'EF_Group_size', 'EF_Fare']
    copy_from_combined(train_data, test_data, combined, saved_cols, 891 )
    # done

    # clean features that have been transformed

    train_data.drop(labels=['PassengerId', 'Name', 'EF_Names', 'Cabin', 'Ticket', 'Age',
'Fare'],
                    axis=1, inplace=True)
```


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

```
    test_data.drop(labels=                    ['Name', 'EF_Names', 'Cabin', 'Ticket', 'Age',
'Fare'],
                   axis=1, inplace=True)


    train_data  =  pd.get_dummies(train_data,  columns=['Embarked',  'Pclass',  'EF_Title',
'EF_Group_type'])
    test_data  =  pd.get_dummies(test_data,  columns=['Embarked',  'Pclass',  'EF_Title',
'EF_Group_type'])
    test_data['EF_Title_Sir']  =  pd.Series(0,  index=test_data.index)  #  missing  in  test
dataset




    X = train_data.drop(['Survived'], axis=1)
    Y = train_data.Survived




    # grid search
    model = XGBClassifier()
    learning_rate = [0.01, 0.1, 0.2, 0.3]
    reg_lambda = [ 2.0, 3.0, 4.0, 5.0]
    reg_alpha =[2.0, 3.0, 4.0, 5.0]
    n_estimators = [100, 400]
    max_depth = [4, 6, 8]
    param_grid = dict(max_depth = max_depth,
                      n_estimators = n_estimators,
                      learning_rate = learning_rate,
                      reg_alpha = reg_alpha,
                      reg_lambda = reg_lambda)
```

```
kfold = StratifiedKFold(Y, n_folds=10, shuffle=True)
scoring = make_scorer(accuracy_score, greater_is_better=True)
grid_search = GridSearchCV(model,
                           param_grid,
                           scoring= scoring,
                           n_jobs=-1,
                           cv=kfold)
result = grid_search.fit(X, Y)
final_clf = result.best_estimator_
# summarize results
means, stdevs = [], []
for params, mean_score, scores in result.grid_scores_:
    stdev = scores.std()
    means.append(mean_score)
    stdevs.append(stdev)
    print("%f (%f) with: %r" % (mean_score, stdev, params))

print("Best: %f using %s" % (result.best_score_, result.best_params_))

SAVE = True
if SAVE:
    PassengerId = test_data.PassengerId
    test_data.drop("PassengerId", axis = 1, inplace = True)
    test_data = test_data[X.columns]
    y_f_preds = final_clf.predict(test_data)
    save_data = pd.DataFrame(columns = ['PassengerId', 'Survived'])
    save_data.PassengerId = PassengerId
    save_data.Survived = pd.Series(y_f_preds, index=save_data.index)

    file_name = str(time.time())+'_pxgb_predictions.csv'
```


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

```
save_data.to_csv(file_name, index=False)
```

4.1.3 Modele proprii testate in Eigen

O alta abordare in vederea determinarii celei mai eficiente si fiabile solutii pentru implementarea componentelor de Machine Learning din cadrul Cloudifier.NET este utilizarea bibliotecilor de nivel scazut scrise in C++, biblioteci ce formeaza actualmente state-of-the-art in aceasta zona si au fost analizate in cadrul etapei/subactivitatii 1.1 “Analiza state-of-the-art” a proiectului. In particular au fost analizate bibliotecile EIGNEN (http://eigen.tuxfamily.org/index.php?title=Main_Page sau https://en.wikipedia.org/wiki/Eigen_(C%2B%2B_library))

In continuare prezentam testele realizate pe aceasta tehnologie pe parcursul desfasurarii activitatilor 1.1 – Analiza State-of-the-Art. Aceste teste reprezinta implementarea la nivelul cel mai jos al nucleului de calcul a algoritmului de Machine Learning bazat regresia logistice softmax.

```
//
// EigenEngine - ML engine based on Eigen library
//
// Created: 11/01/2016
// Last modified: 12/29/2016
//
// @Author: Andrei Ionut DAMIAN
// @Contributor: Octavian BULIE

#pragma once

#include "stdafx.h"
#include "stdio.h"
#include <string>
#include <iostream>
#include <fstream>
#include <vector>
```

```
#include <set>

#include <Eigen/Dense>
#include <Eigen/Core>
#include <Eigen/SVD>

#include <sys/stat.h>

#include <chrono>

#include <algorithm>
#include <random>

using namespace std;
using namespace std::chrono;


using Eigen::MatrixXd;
using Eigen::VectorXd;

using namespace Eigen;
using namespace std;


struct TrainCrossSplits
{
        MatrixXd X_train;
        MatrixXd X_cross;
        vector <string> Labels;
        VectorXd y_train;
        VectorXd y_cross;
};



class GenericEngine
{
private:
```


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

```
        long LoadedDataNrFields;
        long LoadedDataNrRows;
        long TrainTestSplitPos;

        std::default_random_engine random_engine;

protected:
        milliseconds start_time;
        milliseconds end_time;

        bool bBiasAdded; // variable that stores bias information for pre-loaded data

        string CLF_NAME;
        long NR_FEATS;
        long NR_CLASSES;

public:

        bool VERBOSE_ENGINE;

        MatrixXd *X_loaded;
        VectorXd *y_loaded;

        MatrixXd *X_train;
        VectorXd *y_train;

        MatrixXd *X_cross;
        VectorXd *y_cross;

        MatrixXd *LoadedData;
        vector <string> LoadedDataHeader;
        vector <string> LabelsVector;


        GenericEngine()
        {
                CLF_NAME = "Generic Engine";
```


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

```
        // obtain a time-based seed:
        unsigned seed = std::chrono::system_clock::now().time_since_epoch().count();
        random_engine = default_random_engine(seed);

        VERBOSE_ENGINE = true;

        bBiasAdded = false;

        X_train = NULL;
        y_train = NULL;
        X_loaded = NULL;
        y_loaded = NULL;
        X_cross = NULL;
        y_cross = NULL;
        LoadedData = NULL;
}

bool file_exists(const std::string& name);


void debug_info(string str_message)
{
        if (VERBOSE_ENGINE)
                printf("\n[DEBUG] %s", str_message.c_str());
}

void debug_info(string msg, MatrixXd mat)
{
        std::stringstream ss;
        ss << mat;
        string str_matrix = ss.str();
        string msgp = "\n[DEBUG] " + msg + "\n";
        printf(msgp.c_str());
        std::cout << str_matrix << std::endl;
}


void debug_info(MatrixXd mat)
```


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

```
{
	std::stringstream ss;
	ss << mat;
	string str_matrix = ss.str();
	printf("\n[DEBUG] Matrix:\n");
	std::cout << str_matrix << std::endl;
}

void debug_info(VectorXd vec, bool bHorizontal)
{
	std::stringstream ss;
	if (bHorizontal)
		ss << vec.transpose();
	else
		ss << vec;
	string str_vector = ss.str();
	printf("\n[DEBUG] Vector:\n");
	std::cout << str_vector << std::endl;
}

void debug_info(string msg, VectorXd vec, bool bHorizontal)
{
	std::stringstream ss;
	if (bHorizontal)
		ss << vec.transpose();
	else
		ss << vec;
	string str_vector = ss.str();
	string msgp = "\n[DEBUG] " + msg + "\n";
	printf(msgp.c_str());
	std::cout << str_vector << std::endl;
}
void debug_info()
{

	printf("\n[DEBUG] [PRESS ENTER]");
	int c = getc(stdin);
```


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

```
        }


        MatrixXd ShuffleMatrixRows(MatrixXd DataMatrix);


        int FindLabelId(vector <string> labels, string value);


        void BeginTimer();

        long EndTimer();


        vector <string> ToLabels(VectorXd y);


        TrainCrossSplits LoadCSV(const string& inputfile, const bool bShuffle = false, const
bool bAddBias = false);


};


class GenericLinearEngine : public GenericEngine

{

protected:


        long nr_batches; // how many batches have been processed (epochs, online trainings,
etc)

        VectorXd *SingleClassTheta;

        MatrixXd *Theta;

        VectorXd *J_values;


        void add_cost(double J);



private:


        void init();


public:
```


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

```
GenericLinearEngine()
{
        CLF_NAME = "VIRTUAL Generic Linear Engine";
        init();

}
~GenericLinearEngine()
{
        debug_info("Deleting object [" + CLF_NAME + "]");

        if (LoadedData != NULL)
                delete LoadedData;
        if (X_loaded != NULL)
                delete X_loaded;
        if (y_loaded != NULL)
                delete y_loaded;
        if (X_train != NULL)
                delete X_train;
        if (y_train != NULL)
                delete y_train;
        if (X_cross != NULL)
                delete X_cross;
        if (y_cross != NULL)
                delete y_cross;
        if (Theta != NULL)
                delete Theta;
        if (SingleClassTheta != NULL)
                delete SingleClassTheta;
        if (J_values != NULL)
                delete J_values;
}

VectorXd PredictSingleClass(MatrixXd X);

virtual MatrixXd Predict(MatrixXd X);

vector <string> PredictLabels(MatrixXd X);
```


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

```
        vector <string> PredictLabelsUsingYHat(MatrixXd y_hat);

        string GetName();
        MatrixXd& GetTheta();

        float NRMSE(VectorXd y_hat, VectorXd y);
        float RMSE(VectorXd y_hat, VectorXd y);
        float CrossEvaluationSingleClass(bool bClass);
        float TrainEvaluationSingleClass(bool bClass);

        float CrossEvaluation(bool bClass);
        float TrainEvaluation(bool bClass);
};

class NormalRegressor : public GenericLinearEngine
{
protected:

        int t;

public:
        NormalRegressor()
        {
                NR_FEATS = 0;
                NR_CLASSES = 0;
                CLF_NAME = "Batch Normal Regressor";
        }

        void Train(MatrixXd X, MatrixXd y);
        void Train();

};

class OnlineClassifier :  public GenericLinearEngine
{
protected:
```


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

```
        // temp variables

        MatrixXd LastYHat;
        MatrixXd LastGrad;
        MatrixXd LastXObs;
        MatrixXd LastYOHM;
        MatrixXd LastYERR;

        double LearningRate;

        MatrixXd softmax(MatrixXd z);
        double cross_entropy(MatrixXd yOHM, MatrixXd y_hat);


public:
        OnlineClassifier(int nr_features, int nr_classes, vector <string> &labels, double
alpha_learning_rate)
        {
                CLF_NAME = "Online Linear Classifier";
                NR_FEATS = nr_features;
                NR_CLASSES = nr_classes;
                LabelsVector = labels;
                LearningRate = alpha_learning_rate;

                Theta = new MatrixXd(NR_FEATS+1, NR_CLASSES); // add 1 row for biases

                Theta->fill(0);

        }

        void SimulateOnlineTrain();

        void OnlineTrain(MatrixXd xi, VectorXd yi);

        double CostFunction();

        MatrixXd Predict(MatrixXd X);
```


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

```
};



//
// BEGIN Generic Engine Class definitions - basic ancestor helper class
//

inline bool GenericEngine::file_exists(const std::string & name)
{
	if (FILE *file = fopen(name.c_str(), "r")) {
		fclose(file);
		return true;
	}
	else {
		return false;
	}
}
inline MatrixXd GenericEngine::ShuffleMatrixRows(MatrixXd DataMatrix)
{
	long size = DataMatrix.rows();
	PermutationMatrix<Dynamic, Dynamic> perm(size);
	perm.setIdentity();


	std::shuffle(perm.indices().data(),
				perm.indices().data() + perm.indices().size(),
				this->random_engine);

	MatrixXd A_perm = perm * DataMatrix; // permute rows
	return(A_perm);
}

inline int GenericEngine::FindLabelId(vector<string> labels, string value)
{
```


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

```
        int pos = find(labels.begin(), labels.end(), value) - labels.begin();

        if (pos >= labels.size()) {
                //old_name_ not found
                pos = -1;
        }

        return(pos);
}
void GenericEngine::BeginTimer()
{
        milliseconds ms = duration_cast< milliseconds >(
                system_clock::now().time_since_epoch()
                );
        start_time = ms;
}

inline long GenericEngine::EndTimer()
{
        milliseconds ms = duration_cast< milliseconds >(
                system_clock::now().time_since_epoch()
                );
        end_time = ms;
        return (end_time - start_time).count();
}

inline vector<string> GenericEngine::ToLabels(VectorXd y)
{
        vector <string> labels;
        for (long i = 0;i < y.size();i++)
        {
                string s = LabelsVector[y(i)];
                labels.push_back(s);
        }
        return(labels);
}
```


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

```
inline TrainCrossSplits GenericEngine::LoadCSV(const string & inputfile, const bool bShuffle,
const bool bAddBias)

{
        int nr_rows = 0;
        int nr_cols = 0;
        string fname = inputfile;
        TrainCrossSplits rec_results;

        if (!file_exists(inputfile))
                throw std::invalid_argument("Received invalid file in LoadCSV: " + fname);

        ifstream infile(fname, std::ifstream::in);



        if (!infile.good())
                throw std::invalid_argument("Received invalid file in LoadCSV: " + fname);

        debug_info("Loading " + fname + " dataset...");
        vector< vector<string> > result;
        while (!infile.eof())
        {
                //go through every line
                string line;

                getline(infile, line);

                vector <string> record;
                nr_cols = 0;

                std::size_t prev = 0, pos;
                while ((pos = line.find_first_of(",;", prev)) != std::string::npos)
                {
                        if (pos > prev)
                        {
                                record.push_back(line.substr(prev, pos - prev));
                                nr_cols++;
                        }
```


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

```
                prev = pos + 1;
        }
        if (prev < line.length())
        {
                record.push_back(line.substr(prev, std::string::npos));
                nr_cols++;
        }


        if (nr_cols > 0)
        {
                result.push_back(record);
                nr_rows++;
        }
    }


    //
    // now load whole data, X and y matrices
    // assume last column of loaded data is the results / labels
    //

    LoadedDataNrFields = result[0].size();
    LoadedDataNrRows = nr_rows - 1; // rows minus field names row

    debug_info("Loaded " + std::to_string(LoadedDataNrRows) + " X " +
std::to_string(LoadedDataNrFields) + " dataset");

    LoadedData = new MatrixXd(LoadedDataNrRows, LoadedDataNrFields);
    y_loaded = new VectorXd(LoadedDataNrRows);
    X_loaded = new MatrixXd(LoadedDataNrRows, LoadedDataNrFields - 1);

    std::set <string> LabelsSet;


    long i, j;
    for (j = 0;j < LoadedDataNrFields;j++)
        LoadedDataHeader.push_back((string)result[0][j]);
```


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

```
//
// assume dataset is curated and ONLY last column contains text labels
//

vector <string> loaded_labels;

for (i = 0;i < LoadedDataNrRows;i++)
        for (j = 0;j < LoadedDataNrFields;j++)
        {
                double fcell = 0;
                string scell = result[i + 1][j];
                try
                {
                        if (j != ((LoadedDataNrFields - 1)))
                                fcell = ::atof(scell.c_str());
                }
                catch (...)
                {

                }
                (*LoadedData)(i, j) = fcell;
                if (j == (LoadedDataNrFields - 1))
                {
                        LabelsSet.insert(scell);
                        loaded_labels.push_back(scell);
                }

        }

LabelsVector.assign(LabelsSet.begin(), LabelsSet.end());

for (int label_idx = 0;label_idx < loaded_labels.size();label_idx++)
{
        string c_label = loaded_labels[label_idx];
        int iLabel = FindLabelId(LabelsVector, c_label);
        (*LoadedData)(label_idx, LoadedDataNrFields - 1) = iLabel;
```


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

```
	}


	if (bShuffle)
	{
		MatrixXd ttt = ShuffleMatrixRows(*LoadedData);
		*LoadedData = ttt;

	}

	float test_size = 0.2;
	int test_rows = LoadedDataNrRows * test_size;
	int train_rows = LoadedDataNrRows - test_rows;
	TrainTestSplitPos = train_rows;

	*X_loaded = LoadedData->leftCols(LoadedDataNrFields - 1);
	*y_loaded = LoadedData->rightCols(1);

	NR_FEATS = X_loaded->cols();
	NR_CLASSES = LabelsVector.size();


	if (bAddBias)
	{
		// now add bias
		VectorXd bias(LoadedDataNrRows);
		bias.fill(1);
		MatrixXd *TempX = new MatrixXd(LoadedDataNrRows, LoadedDataNrFields - 1 + 1);
// bias size
		*TempX << bias, *X_loaded;
		bBiasAdded = true;
		delete X_loaded;
		X_loaded = TempX;
		// done adding bias
	}

	X_train = new MatrixXd(X_loaded->topRows(train_rows));
	X_cross = new MatrixXd(X_loaded->bottomRows(test_rows));
```


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

```
        y_train = new VectorXd(y_loaded->head(train_rows));
        y_cross = new VectorXd(y_loaded->tail(test_rows));

        rec_results.X_cross = *X_cross;
        rec_results.X_train = *X_train;
        rec_results.y_cross = *y_cross;
        rec_results.y_train = *y_train;
        rec_results.Labels = LabelsVector;

        return(rec_results);

}

//
// END Generig Engine Class definitions
//

//
// BEGIN Generic Linear Engine (Virtual class)
//

inline float GenericLinearEngine::NRMSE(VectorXd y_hat, VectorXd y)
{
        float maxmin = y.maxCoeff()-y.minCoeff();
        return(RMSE(y_hat, y) / maxmin);
}

inline float GenericLinearEngine::RMSE(VectorXd y_hat, VectorXd y)
{
        long nr_obs = y.size();
        VectorXd errors = (y-y_hat);
        if (VERBOSE_ENGINE)
        {
                debug_info("Errors (last 3):");
                debug_info(errors.tail(3));
        }
        double sqNorm = errors.squaredNorm();
```


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

```
        return(sqrt(sqNorm / nr_obs));
}

inline void GenericLinearEngine::add_cost(double J)
{
        if (nr_batches == 0)
        {
                // first use :)
                J_values = new VectorXd(1);
                (*J_values)(nr_batches) = J;
        }
        else
        {
                J_values->conservativeResize(nr_batches + 1);
                (*J_values)(nr_batches) = J;
        }
        nr_batches++;
}

inline void GenericLinearEngine::init()
{

        debug_info("Generating object [" + CLF_NAME + "]");

        nr_batches = 0;
        Theta = NULL;
        SingleClassTheta = NULL;
        J_values = NULL;


}

double myexp(double val)
{
        return(exp(val));
}

MatrixXd& GenericLinearEngine::GetTheta()
```


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

```
{
        return *Theta;
}
string GenericLinearEngine::GetName()
{
        return(CLF_NAME);
}
```

```
double myround(double f)
{
        return(round(f));
}

inline VectorXd GenericLinearEngine::PredictSingleClass(MatrixXd X)
{
        VectorXd *pred = new VectorXd(X.rows());

        *pred = X * (*SingleClassTheta);

        return(*pred);
}

inline MatrixXd GenericLinearEngine::Predict(MatrixXd X)
{
        MatrixXd preds = X * (*Theta);
        return(preds);
}

inline vector<string> GenericLinearEngine::PredictLabels(MatrixXd X)
{
        MatrixXd y_hat = Predict(X);
        vector <string> PredictedLabels;
```


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

```
	for (long i = 0;i < y_hat.rows();i++)
	{
		int y_hat_idx;
		y_hat.row(i).maxCoeff(&y_hat_idx);
		PredictedLabels.push_back(LabelsVector[y_hat_idx]);
	}
	return(PredictedLabels);
}

inline vector<string> GenericLinearEngine::PredictLabelsUsingYHat(MatrixXd y_hat)
{
	vector <string> PredictedLabels;
	for (long i = 0;i < y_hat.rows();i++)
	{
		int y_hat_idx;
		y_hat.row(i).maxCoeff(&y_hat_idx);
		PredictedLabels.push_back(LabelsVector[y_hat_idx]);
	}
	return(PredictedLabels);
}


inline float GenericLinearEngine::TrainEvaluationSingleClass(bool bClass)
{
	double dResult = 0.0f;
	VectorXd y = *y_train;
	MatrixXd X = *X_train;
	long nr_train = y.size();
	if (SingleClassTheta == NULL && Theta == NULL)
		return (dResult);

	VectorXd y_hat = PredictSingleClass(X);
	long nr_obs = y_hat.size();

	if (VERBOSE_ENGINE)
	{
		debug_info("Train Y_Hat vs. Y_train (last 3)");
		MatrixXd result(nr_train, 2);
```


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

```
		result << y_hat, y;
		debug_info(result.bottomRows(3));
	}


	if (bClass)
	{
		VectorXd y_hat_Rounded = y_hat.unaryExpr(ptr_fun(myround));
		long positives = 0;
		for (long i = 0;i < nr_obs;i++)
		{
			if (y_hat_Rounded(i) == (y)(i))
				positives++;
		}
		dResult = (double)positives / nr_obs;
	}
	else
	{
		dResult = NRMSE(y_hat, y);
	}

	return (dResult);
}

inline float GenericLinearEngine::CrossEvaluation(bool bClass)
{
	double dResult = 0.0f;
	VectorXd y = *y_cross;
	MatrixXd X;
	if (!bBiasAdded)
		X = *X_cross;
	else
		X = X_cross->rightCols(NR_FEATS);

	long nr_cross = y.size();
	if (Theta == NULL)
		return (dResult);
```


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

```
	MatrixXd y_hat = Predict(X);

	long nr_obs = X.rows();

	if (VERBOSE_ENGINE)
	{
		MatrixXd result(nr_cross, y_hat.cols() + 1);
		result << y_hat, y;
		debug_info("Cross Y_Hat vs. Y_cross (last 5):",result.bottomRows(5));
	}


	if (bClass)
	{
		vector <string> preds = PredictLabelsUsingYHat(y_hat);
		long positives = 0;
		for (long i = 0;i < nr_obs;i++)
		{
			string predicted = preds[i];
			string label = LabelsVector[(int)y(i)];
			if (predicted == label)
				positives++;
		}
		dResult = (double)positives / nr_obs;
	}
	else
	{
		dResult = -1;
	}

	return (dResult);

}

inline float GenericLinearEngine::TrainEvaluation(bool bClass)
{
	double dResult = 0.0f;
	VectorXd y = *y_train;
```


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

```
MatrixXd X;
if (!bBiasAdded)
	X = *X_train;
else
	X = X_train->rightCols(NR_FEATS);

long nr_cross = y.size();
if (Theta == NULL)
	return (dResult);

MatrixXd y_hat = Predict(X);

long nr_obs = X.rows();

if (VERBOSE_ENGINE)
{
	MatrixXd result(nr_cross, y_hat.cols() + 1);
	result << y_hat, y;
	debug_info("Train Y_Hat vs. Y_train (last 5):", result.bottomRows(5));
}


if (bClass)
{
	vector <string> preds = PredictLabelsUsingYHat(y_hat);
	long positives = 0;
	for (long i = 0;i < nr_obs;i++)
	{
		string predicted = preds[i];
		string label = LabelsVector[(int)y(i)];
		if (predicted == label)
			positives++;
	}
	dResult = (double)positives / nr_obs;
}
else
{
	dResult = -1;
```


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

```
	}

	return (dResult);

}

inline float GenericLinearEngine::CrossEvaluationSingleClass(bool bClass)
{
	double dResult = 0.0f;
	VectorXd y = *y_cross;
	MatrixXd X = *X_cross;
	long nr_cross = y.size();
	if (SingleClassTheta == NULL && Theta == NULL)
		return (dResult);

	VectorXd y_hat = PredictSingleClass(X);
	long nr_obs = y_hat.size();

	if (VERBOSE_ENGINE)
	{
		debug_info("Cross Y_Hat vs. Y_cross (last 3)");
		MatrixXd result(nr_cross, 2);
		result << y_hat, y;
		debug_info(result.bottomRows(3));
	}


	if (bClass)
	{
		VectorXd y_hat_Rounded = y_hat.unaryExpr(ptr_fun(myround));
		long positives = 0;
		for (long i = 0;i < nr_obs;i++)
		{
			if (y_hat_Rounded(i) == y(i))
				positives++;
		}
		dResult = (double) positives / nr_obs;
	}
```


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

```
        else
        {
                dResult = NRMSE(y_hat, y);
        }

        return (dResult);
}
//
// END Generic Linear Engine virtual class
//



//
// BEGIN Normal Regressor class definitions
//
void NormalRegressor::Train(MatrixXd X, MatrixXd y)
{
        X_train = new MatrixXd(X);
        y_train = new VectorXd(y);
        Train();
}


template<typename _Matrix_Type_>
_Matrix_Type_ pseudoInverse(const _Matrix_Type_ &a, double epsilon = std::numeric_limits<double>::epsilon())
{
        Eigen::JacobiSVD< _Matrix_Type_ > svd(a, Eigen::ComputeThinU | Eigen::ComputeThinV);
        double tolerance = epsilon * std::max(a.cols(), a.rows()) *svd.singularValues().array().abs()(0);
        return svd.matrixV() * (svd.singularValues().array().abs() > tolerance).select(svd.singularValues().array().inverse(), 0).matrix().asDiagonal() * svd.matrixU().adjoint();
}


void NormalRegressor::Train()
{
        debug_info("Training: " + CLF_NAME);
        MatrixXd X = *X_train;
```


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

```
VectorXd y = *y_train;
MatrixXd xTx = X.transpose() * X;
MatrixXd xT = X.transpose();

VectorXd TempTheta1(X.cols());
VectorXd TempTheta2(X.cols());
long duration1;
long duration2;

if (VERBOSE_ENGINE)
{
        // 1st solving with pseudo-inverse
        high_resolution_clock::time_point t1 = high_resolution_clock::now();
        MatrixXd xTxInv = pseudoInverse(xTx);
        TempTheta1 = xTxInv * xT * y;
        high_resolution_clock::time_point t2 = high_resolution_clock::now();
        duration1 = duration_cast<microseconds>(t2 - t1).count();



        // now second method
        high_resolution_clock::time_point t3 = high_resolution_clock::now();
        TempTheta2 = xTx.ldlt().solve(xT * y);
        high_resolution_clock::time_point t4 = high_resolution_clock::now();
        duration2 = duration_cast<microseconds>(t4 - t3).count();

        //SingleClassTheta = new VectorXd(TempTheta1);
        SingleClassTheta = new VectorXd(TempTheta2);


}
else
{
        // now second method
        TempTheta2 = xTx.ldlt().solve(xT * y);
        SingleClassTheta = new VectorXd(TempTheta2);
}
```


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

```
	if (VERBOSE_ENGINE)
	{
		debug_info("X data features size = " + to_string(X_loaded->cols()));

		debug_info("Theta PInv = " + to_string(duration1) + " microsec");
		debug_info("Theta ldlt = " + to_string(duration2) + " microsec");

		debug_info("T1(pinv) T2(ldlt):");
		MatrixXd comp(TempTheta1.size(), 2);
		comp << TempTheta1, TempTheta2;
		debug_info(comp);
		if (*SingleClassTheta == TempTheta2)
			debug_info("Using Theta2");
		else
			debug_info("Using Theta1");

	}
}
//
// END Normal Regressor class definitions
//



inline void OnlineClassifier::SimulateOnlineTrain()
{
	if (Theta != NULL)
		delete Theta;
	Theta = new MatrixXd(NR_FEATS + 1, NR_CLASSES);
	Theta->fill(0); // reset Theta

	long TEST_DEBUG = 1000;

	BeginTimer();

	for (long i = 0;i < X_train->rows();i++)
	{
```


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

```
		MatrixXd obs = X_train->row(i);
		VectorXd yi(1);
		yi(0) = (*y_train)(i);


		if (VERBOSE_ENGINE)// && (i == TEST_DEBUG))
		{
			std::stringstream ss;
			for (size_t i = 0; i < yi.size(); ++i)
			{
				if (i != 0)
					ss << ",";
				ss << yi[i];
			}

			debug_info("Training "+to_string(i)+" th example with y = " +
ss.str(),obs);
		}

		MatrixXd xi;
		if (bBiasAdded)
			xi = obs.rightCols(NR_FEATS);
		else
			xi = obs;

		OnlineTrain(xi, yi);

		if (VERBOSE_ENGINE)// && (i == TEST_DEBUG))
		{
			//long time_cost = EndTimer();
			//debug_info("Total time = " + to_string(time_cost) + " ms");
			debug_info("y_OHM (1 row): ",LastYOHM.topRows(1));
			debug_info("y_hat (1 row): ",LastYHat.topRows(1));
			debug_info("error (1 row): ",LastYERR.topRows(1));
			debug_info("Gradient (2 rows): ",LastGrad.topRows(2));

			debug_info("J array las val: ",J_values->tail(1), true);
			debug_info("Theta (2 rows): ",Theta->topRows(2));
```


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

```
                        //debug_info();
                }
        }
}

//
// BEGIN Online Classifier definitions
//
// yi is index in VectorLabels
void OnlineClassifier::OnlineTrain(MatrixXd xi, VectorXd yi)
{
        long nr_rows = xi.rows();
        long nr_cols = xi.cols();

        VectorXd bias(nr_rows);
        bias.fill(1);
        MatrixXd TempX(nr_rows, nr_cols + 1);
        TempX << bias, xi;

        long m = nr_rows; // for convenience
        MatrixXd yOHM(nr_rows, NR_CLASSES);
        yOHM.fill(0);
        for (long i = 0;i < nr_rows;i++)
        {
                for (long j = 0;j < NR_CLASSES;j++)
                        // now assume LabelsVector is correctly constructed
                        // and yi[i] is index in that vector
                        if (yi(i) == j)
                                yOHM(i, j) = 1;
        }

        // now we have the one hot matrix lets start working !
        MatrixXd y_hat = Predict(xi);
        double J = (1.0 / m) * cross_entropy(yOHM, y_hat); // MUST add regularization
        add_cost(J);

        MatrixXd error = yOHM - y_hat;
```


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

```
        MatrixXd Grad = (-1.0 / m) * TempX.transpose() * error; // MUST add regularization


        *Theta = *Theta - (LearningRate * Grad);

        LastGrad = Grad;
        LastYOHM = yOHM;
        LastYHat = y_hat;
        LastYERR = error;
        LastXObs = xi;

}

inline double OnlineClassifier::CostFunction()
{

        return 0.0;
}
inline MatrixXd OnlineClassifier::Predict(MatrixXd X)
{
        long nr_rows = X.rows();
        long nr_cols = X.cols();

        VectorXd bias(nr_rows);
        bias.fill(1);
        MatrixXd TempX(nr_rows, nr_cols +1);
        TempX << bias, X;
        MatrixXd XTheta = TempX * (*Theta);

        MatrixXd SM = softmax(XTheta);

        return(SM);
}



inline MatrixXd OnlineClassifier::softmax(MatrixXd z)
{
```


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

```
        MatrixXd SM(z.rows(), Theta->cols());

        ArrayXXd arr(z);

        // first shift values
        arr = arr - z.maxCoeff();
        arr = arr.exp();

        //cout << z;
        //cout << arr;

        ArrayXd sums = arr.rowwise().sum();

        arr.colwise() /= sums;

        SM = arr.matrix();


        return(SM);

}

double myclip(double val)
{
        double eps = 1e-15;
        if (val < eps)
                return(eps);
        else
                if (val > (1 - eps))
                        return(1 - eps);
                else
                        return(val);
}
inline double OnlineClassifier::cross_entropy(MatrixXd yOHM, MatrixXd y_hat)
{
        //y_hat = y_hat.unaryExpr(ptr_fun(myclip));

        MatrixXd J_matrix = (yOHM.array() * y_hat.array().log()).matrix();
```


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

```
        double J = -(J_matrix.sum());
        return(J);
}
//
// END Online Classifier definitions
//
```


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

4.2 Tehnici bazate pe Retele Neurale Adanci

4.2.1 Utilizarea CNN in segmentarea semantica

Retelele convolutionale adanci constituie principalul vector stiintific si tehnologic care avanseaza stadiul actual al tehnologiei si cercetarii in domeniul recunoasterii computerizate.

Reţelele convolutionale sunt modele de Machine Learning cu capabilitati de vizualizare puternice, care produce ierarhii de caracteristici. Intrinsec, retelele convolutionale adanci ne pot prezenta informatia vizualala pe diverse nivele de analiza semantica cu ajutorul nivelelor "ascunse" (hidden layers) de neuroni artificiali. Conform lucrarii stiintifice "**Fully Convolutional Networks for Semantic Segmentation**" publicata de Johnatan Long, Trevor Darell et al. se arata ca retelele convolutionale dense si complete se pot proiecta si construi astfel incat sa genereze predictii si inferente semantice complete la nivel de pixel – acesta fiind pasul natural al modificarii predictiei brute de la nivelul intregii imagini care este analiza pana la nivelul punctului grafic unitar (pixel). Astfel se poate asocia fiecare pixel individual cu o anumita forma bidimensionala sau tridimensionala.

4.2.2 Utilizarea LSTM pentru sisteme expert de tip Bot

Retelele neurale de tip LSTM (Long Short Term Memory) simuleaza functionarea neuronilor biologici responsabili de stocarea informatiei pe termen scurt si lung cu ajutorul unei porti de scriere, stocare si rescriere.

In materialele studiate este demonstrat atat prin formalizare matematica la nivel de model abstract de Machine Learning cat si prin demonstratii experimentale ca retelele neurale adanci de tip LSTM pot fi antranate complet cu script-uri de tip call-center sau alte tipuri de conversatii realizate anterior intre actori umani. In urma antrenarii reteaua LSTM este capabila sa formuleze raspunsuri coerente si logice la intrebari/probleme descrise de un utilizator uman in

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

limbaj natural – utilizand fraza/intrebarea in limbaj natural a operatorului uman ca si punct de plecare a analizei si predictiei raspunsului ideal.

Concret vom utiliza retele de tip LSTM in vederea realizarii unui Bot care sa permita unui utilizator uman sa descrie modul de functionare al aplicatiei informatice dorite iar sistemul Cloudifier.NET va putea formaliza in limbaje de programare notiunile intelese, analizate si proiectate de componentele LSTM are Bot-ului.

4.2.3 Framework-ul state-of-the-art TensorFlow

Pentru implementarea tuturor algoritmilor de tip Deep Learning se va utiliza state-of-the-art actual in acest domeniu lansat de Google in anul 2016 – framework-ul TensorFlow impreuna cu framework-ul Keras care are scopul de a adauga facilitati de nivel inalt bibliotecilor TensorFlow. In continuare este prezentata sintaxa TensorFlow (descrisa in limbajul Python) urmata de sintaxa Keras.

```
import tensorflow as tf
import numpy as np

# Create 100 phony x, y data points in NumPy, y = x * 0.1 + 0.3
x_data = np.random.rand(100).astype(np.float32)
y_data = x_data * 0.1 + 0.3

# Try to find values for W and b that compute y_data = W * x_data + b
# (We know that W should be 0.1 and b 0.3, but TensorFlow will
# figure that out for us.)
W = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
b = tf.Variable(tf.zeros([1]))
y = W * x_data + b

# Minimize the mean squared errors.
loss = tf.reduce_mean(tf.square(y - y_data))
optimizer = tf.train.GradientDescentOptimizer(0.5)
train = optimizer.minimize(loss)
```


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

```
# Before starting, initialize the variables.  We will 'run' this first.
init = tf.global_variables_initializer()

# Launch the graph.
sess = tf.Session()
sess.run(init)

# Fit the line.
for step in range(201):
    sess.run(train)
    if step % 20 == 0:
        print(step, sess.run(W), sess.run(b))

# Learns best fit is W: [0.1], b: [0.3]
```

4.2.4 Framework-ul state-of-the-art Keras

```
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation
from keras.optimizers import SGD

model = Sequential()
# Dense(64) is a fully-connected layer with 64 hidden units.
# in the first layer, you must specify the expected input data shape:
# here, 20-dimensional vectors.
model.add(Dense(64, input_dim=20, init='uniform'))
model.add(Activation('tanh'))
model.add(Dropout(0.5))
model.add(Dense(64, init='uniform'))
model.add(Activation('tanh'))
model.add(Dropout(0.5))
model.add(Dense(10, init='uniform'))
model.add(Activation('softmax'))

sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy',
              optimizer=sgd,
              metrics=['accuracy'])

model.fit(X_train, y_train,
          nb_epoch=20,
          batch_size=16)
score = model.evaluate(X_test, y_test, batch_size=16)
```


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

5 Anexa – Rapoarte lunare

5.1 Raport stiintific lunar 1

Raport stiintific

de cercetare-dezvoltare in cadrul Cloudifier SRL

Nr. 62/31.10.2016

Nume proiect	Platforma de migrare automatizată în cloud a aplicațiilor și sistemelor informatice clasice cloudifier.net
Beneficiar	CLOUDIFIER SRL
Cod MySMIS	104349
Nr. iregistrare	P_38_543
Director Proiect	Andrei Ionut DAMIAN
Activitate conform planului de proiect	1. Activităţi de cercetare-dezvoltare (cercetare industrială şi/sau dezvoltare experimentală) - 1.1 State-of-the-art
Luna	Octombrie 2016
Echipa de cercetare-dezvoltare	Andrei Ionut DAMIAN Octavian BULIE
Descrierea activitatilor desfasurate activitatii	In decursul acestei luni a inceput procesul de analiza a stadiului curent al tehnologiei in domeniul sistemelor de tip Machine Learning cu accent pe zona de Deep Learning, domeniu de

cercetare in dezvoltare la nivel international ce a luat amploare deosebita in ultimii 5 ani.

Scopul analizei stadiului actual al tehnologiei este acela de a determina metodele cele mai moderne/actuale de realizare a predictiilor/inferentelor in imagistica – in particular in cazul proiectului CLOUDIFIER referindu-ne la analiza imaginilor captate in timp real in timpul functionarii aplicatiilor si implicit analiza automatizata cu ajutorul recunoasterii avansate de forme/imagini a aplicatiilor “legacy” in vederea transalatarii acestora automatizate.

In decursul lunii octombrie 2016 au fost analizate cele mai recente si avansate lucrari de cercetare fundamentala si industriala provenite de la cele mai prestigioase institute si universitati printre care enumeram:

- Caltech – California Institute for Technology
- MIT – Massachusetts Institute for Technology
- Stanford
- University of Toronto
- Harvard
- University of Washington

Principalele zone analizate au fost:

- Metodele de tip Deep Learning bazate pe Retele Neuronale Convolutionale – Deep Convolutional Neural Networks
- Metode de tip shallow learning pentru invatarea supervizata a structurilor si a elementelor de imagistica utilizand modele de invatare in timp real (online learning)

- Cele mai moderne abordari in Deep Learning – Tensor Flow
- Cele mai moderne abordari in shallow learning – Extreme Boosted Decision Trees / Random Forests - XGBoost

In decursul lunilor noiembrie si decembrie se va continua analiza conform graficului de implementare a proiectului cu accent pe urmatoarele:

- Determinarea unui algoritm ideal pentru identificarea primitivelor de interfata grafica (butoane, campuri, ferestre, texte statice, etc) si a pozitiei acestora in cadrul ecranelor interfetelor grafice
- Determinarea unui algoritm de tip Machine Learning pentru generarea AUTOMATA de interfe grafice si cod sursa aferente pe baza schitelor facute manual pe suport de hartie, tabla, etc
- Analiza TensorFlow
- Analiza XGBoost
- Analiza metode si propuneri pentru biblioteci interne
- Analiza si testarea experimentala a mediilor de procesare numerica masiv paralela cu ajutorul GPU (tehnologiile bazate pe nuclee de calul masiv paralel CUDA)

Perioada	Efort in ore-om	Descriere
3.10.2016-14.10.2016	160	Selectia si analiza preliminara a celor mai importante lucrcari din domeniul recunoasterii de imagini cu ajutorul retelelor adanci neurale convolutionale (Deep Convolutional Neural Networks). A fost inceputa analiza state-of-the-art pe ultimile

17.10.2016-21.10.2016	80	cercetari realizate de laboratoarele de cercetare ale Google in Inteligenta Artificiala – biblioteca TensorFlow Analiza XGBoost – actualmente cea mai puternica infrastructura si biblioteca de shallow learning bazata pe modele de tip ansamblu Inceperea efectuarii de teste experimentale pe modele arhitecturale simple bazate pe regresii logistice adaptate si optimizate online si retele neural cu conectare completa. Testele s-au realizat dupa cum urmeaza: • Python cu ajutorul: ○ Sci-Kit-Learn ○ Biblioteca dezvoltata intern in cadrul Cloudifier pentru regresii logistice avansate (OnlineClassifierEngine.py) ○ Biblioteca de retele neurale cu conectivitate completa realizata in cadrul Cloudifier • C++ cu ajutorul bibliotecii de calcul numeric optimizat Eigen
24.10.2016-31.10.2016	96	

5.2 Raport stiintific lunar 2

Raport stiintific

de cercetare-dezvoltare in cadrul Cloudifier SRL

Nr. 98/29.12.2016

Nume proiect	Platforma de migrare automatizată în cloud a aplicațiilor și sistemelor informatice clasice cloudifier.net
Beneficiar	CLOUDIFIER SRL
Cod MySMIS	104349
Nr. iregistrare	P_38_543
Director Proiect	Andrei Ionut DAMIAN
Activitate conform planului de proiect	1. Activităţi de cercetare-dezvoltare (cercetare industrială şi/sau dezvoltare experimentală) - 1.1 State-of-the-art
Luna	Noiembrie 2016
Echipa de cercetare-dezvoltare	Andrei Ionut DAMIAN Octavian BULIE
Descrierea activitatilor desfasurate activitatii	In decursul acestei luni a fost continuat procesul de analiza a stadiului curent al tehnologiei in domeniul sistemelor de tip Machine Learning cu accent pe zona de Deep Learning, domeniu de cercetare in dezvoltare la nivel international ce a luat amploare deosebita in ultimii 5 ani. Scopul analizei stadiului actual al tehnologiei este acela de a determina metodele cele mai moderne/actuale de realizare a

predictiilor/inferentelor in imagistica – in particular in cazul proiectului CLOUDIFIER referindu-ne la analiza imaginilor captate in timp real in timpul functionarii aplicatiilor si implicit analiza automatizata cu ajutorul recunoasterii avansate de forme/imagini a aplicatiilor “legacy” in vederea transalatarii acestora automatizate.

In decursul lunii noiembrie 2016 au fost analizate cele mai recente si avansate lucrari de cercetare fundamentala in vederea determinarii unui set de algoritmi de Machine Learning ideali pentru identificarea primitivelor de interfata grafica (butoane, campuri, ferestre, texte statice, etc) si a pozitiei acestora in cadrul ecranelor interfetelor grafice

- Determinarea unui algoritm de tip Machine Learning pentru generarea AUTOMATA de interfe grafice si cod sursa aferente pe baza schitelor facute manual pe suport de hartie, tabla, etc
- Analiza TensorFlow
- Analiza XGBoost
- Analiza metode si propuneri pentru biblioteci interne
- Analiza si testarea experimentala a mediilor de procesare numerica masiv paralela cu ajutorul GPU (tehnologiile bazate pe nuclee de calul masiv paralel CUDA)

In decursul lunii decembrie se va continua analiza inceputa in lunile octombrie si noiembrie conform graficului de implementare a proiectului.

Perioada	Efort in ore-om	Descriere
01.11.2016-11.11.2016	144	Analiza model arhitectural de Retea Neurala total conectata in Python pentru analiza elementelor de interfata grafica si testarea experimentala a acesteia pe seturi de data de imagistica. Analiza PyCUDA – infrastructura programabila pentru procesoarele de calcul numeric masiv paralel CUDA
14.11.2016-30.11.2016	208	Testarea in regim CPU si GPU a celui mai recent framework de Deep Learning realizat de laboaratoarele de cercetare ale Google – TensorFlow

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

5.3 Raport stiintific lunar 3

Raport stiintific lunar

de cercetare-dezvoltare in cadrul Cloudifier SRL

Nr. 112/9.12.2016

Nume proiect	Platforma de migrare automatizată în cloud a aplicațiilor și sistemelor informatice clasice cloudifier.net
Beneficiar	CLOUDIFIER SRL
Cod MySMIS	104349
Nr. iregistrare	P_38_543
Director Proiect	Andrei Ionut DAMIAN
Activitate conform planului de proiect	1. Activităţi de cercetare-dezvoltare (cercetare industrială şi/sau dezvoltare experimentală) - 1.1 State-of-the-art
Luna	decembrie 2016
Echipa de cercetare-dezvoltare	Andrei Ionut DAMIAN Octavian BULIE
Descrierea activitatilor desfasurate activitatii	In decursul acestei luni a fost continuat procesul de analiza a stadiului curent al tehnologiei in domeniul sistemelor de tip Machine Learning cu accent pe zona de Deep Learning si in particular a sistemelor de analiza si recunoastere bazata pe inteligenta artificiala a imaginilor.

In decursul acestei luni analiza stadiului curent al cercetarii fost fost axat in principal pe lucrarea stiintifica publicata recent de J. Long et al “Fully Convolutional Networks for Semantic Segmentation”, lucrare considerata actualmente state-of-the-art in ceea priveste metodele de recunoastere si segmentare a componentelor in cadrul imaginilor. Pentru referinta prezentam anexat un scurt rezumat in limba engleza a lucrarii de referinta.

Principalele puncte pe care le urmarim in cercetare sunt urmatoarele:

1. Determinarea metodelor optime bazate pe Deep Learning pentru recunoasterea si segmentarea (identificarea locatiei spatiale) a elementelor de interfata grafica pe care Cloudifier.NET va trebuie sa le translateze automatizat din aplicatiile legacy in aplicatiile din mediul cloud computing.
2. Aplicarea de metode simple bazate pe algoritmi de machine learning superficiali (regresie logistica, arbori de decizie, clasificare naiva bazata pe teorema lui Bayes, clusterizare cu analiza distantelor euclidiene) precum si metode de segmentare iterativa a imaginilor analizate cum ar fi metoda ferestrelor deplasate continuu (“ferestre alunecatoare” sau sliding-windows algorithm)

Perioada	Efort in ore-om	Descriere
01.12.2016-09.12.2016	96	Continuarea analizei metodelor de recunoastere a imaginilor prin CNN (Convolutional Deep Neural Networks)
12.12.2016-	224	

Perioada	Efort in ore-om	Descriere
31.12.2016		Analiza unui model arhitectural Alpha ce urmeaza a fi definitivat in cadrul activitatii 1.2 de cercetare. Sistemul/model arhitectural Alpha va consta in construirea unui model matematic predictiv care sa poate recunoaste elemente simple de interfata grafica de utilizator (meniu, buton, etc) si sa poata reda locatia si 1-2 alte atribute de baza ale acestora

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

Fully Convolutional Networks for Semantic Segmentation

Evan Shelhamer, Jonathan Long, Trevor Darrell

(Submitted on 20 May 2016)

Convolutional networks are powerful visual models that yield hierarchies of features. We show that convolutional networks by themselves, trained end-to-end, pixels-to-pixels, improve on the previous best result in semantic segmentation. Our key insight is to build "fully convolutional" networks that take input of arbitrary size and produce correspondingly-sized output with efficient inference and learning. We define and detail the space of fully convolutional networks, explain their application to spatially dense prediction tasks, and draw connections to prior models. We adapt contemporary classification networks (AlexNet, the VGG net, and GoogLeNet) into fully convolutional networks and transfer their learned representations by fine-tuning to the segmentation task. We then define a skip architecture that combines semantic information from a deep, coarse layer with appearance information from a shallow, fine layer to produce accurate and detailed segmentations. Our fully convolutional network achieves improved segmentation of PASCAL VOC (30% relative improvement to 67.2% mean IU on 2012), NYUDv2, SIFT Flow, and PASCAL-Context, while inference takes one tenth of a second for a typical image.

Comments: to appear in PAMI (accepted May, 2016); journal edition of arXiv:1411.4038
Subjects: Computer Vision and Pattern Recognition (cs.CV)
Cite as: arXiv:1605.06211 [cs.CV]

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