Universitatea Politehnica Bucuresti

 Facultatea de Automatică și Calculatoare, Departamentul de Informatică și Inginerie

**Inferență scenelor artificiale bazată pe execuția eficientă în paralel a grafurilor tensorale aciclice direcționate**

***Rezumat***

**Andrei Ionut DAMIAN (doctorand)**

**Nicolae Tapus (conducator de doctorat)**

**2021**

Contents

[1 Abstract 3](#_Toc89958242)

[2 Introducere 5](#_Toc89958243)

[2.1 Analiza aplicativitații cercetarii si dezvoltarii propuse in viața reală 5](#_Toc89958244)

[2.2 Vederea de ansamblu a propunerii de arhitectura si implementare 9](#_Toc89958245)

[2.3 Experimente vs. aplicativitate reala: provocări și așteptări 12](#_Toc89958246)

[3 Rezumatul tezei 14](#_Toc89958247)

[3.1.1 1.2.1 Problema propusă 14](#_Toc89958248)

[3.1.2 1.2.2 Rezumatul capitolului lucrărilor aferente 15](#_Toc89958249)

[3.1.3 1.2.3 Rezumatul capitolului Arhitectură 17](#_Toc89958250)

[3.1.4 1.2.4 Rezultate notabile ale cercetării 18](#_Toc89958251)

[3.1.5 1.2.5 Experimente [CMM24] 19](#_Toc89958252)

[3.1.6 1.2.6 Rezumatul principalelor contribuții personale 21](#_Toc89958253)

[3.1.7 1.2.7 Lucru în continuare 22](#_Toc89958254)

# Abstract

Construirea interfețelor grafice necesare interacțiunii utilizatorilor cu aplicatiile *(Graphical User Interface - GUI)*este un subdomeniu important al domeniului dezvoltării software, indiferent de mediul de implementare, scopul final al sistemelor, orizontatul sau verticalul țintă.

Inca de la inceputurile istoriei *GUI-urilor* au fost dezvoltate diverse tehnici pentru proiectarea vizuala, arhitecturala și implementarea propriu-zisă a interfețelor utilizator adresandu-se platforme diferite, incercandu-se echilibrarea intre viteza de implementare (prin *dezvoltarea* *rapidă a aplicației*- *RAD*), calitatea graficii și uzabilitate. Pe măsură ce sistemele au evoluat de-a lungul anilor, cu generație după generație, a apărut o nouă gamă de nevoi legate de *GUI*: nevoia de a regândi, re-proiecta și reconstrui interfețele grafice cu utilizatorul, precum și nevoia de automatia anumite comportamente. De mentionat ca prin „comportament automatizat” ne referim la sarcina de secvențiere și automatizare fara interactiune umana a unei serii de comenzi care sunt de obicei date de un operator uman al aplicatiei.

Nevoile menționate mai sus, respectiv de reproiectare si re-structurare, au fost abordate de ceva timp cu tehnici clasice precum: reutilizarea codului (dacă este disponibil) și reproiectarea interfețelor grafice (*GUI).*Pentru cazul de automatizare a comportamentului, au fost si sunt propuse de metode ce abordeaza simularea de “acțiuni *GUI*” bazate pe euristici programabile („scriptare”) precum „clic pe coordonatele X,Y pentru activare buton” sau “mutare cursor la pozitia H,W si introducere nume utilizator”. 

Astfel, instrumentele software de dezvoltare rapidă a aplicațiilor (*Rapid Application Development -* *RAD*) au evoluat pentru a oferi cicluri de dezvoltare a *GUI* cat mai rapide și mai productive și, legat de subiectul automatizării, a apărut o nouă gamă de sisteme software: sisteme software de *automatizare robotizata a proceselor* (sau RPA) care permit utilizatorii să construiască scenarii de automatizare în care serii de comenzi și operațiuni *GUI* sunt in fapt scriptate și executate pe baza euristicilor.

Cu toate acestea, în toate aceste cazuri, câteva probleme inerente nu au fost rezolvate, cum ar fi: (i) reproiectarea aplicațiilor software pentru care codul sursă initial nu mai este disponibil; (ii) iterație rapidă de la machetele produse de un designer grafic la interfețele funcționale – *de la mokup la GUI functional*; (iii) înțelegerea vizuala semantică  si automata, de catre un agent virtual intelligent,  a componentelor interfeței grafice cu utilizatorul, sau mai concret înțelegerea funcționalității *GUI-*ului si a elementelor sale vizuale;  (iv) înțelegerea semantică rapidă a formularelor tipărite/pre-tiparite sau grafice (pe ecran) și conversia informațiilor, fără limitarea la metodele clasice proprii *recunoașterii caracterelor*(OCR).

Toate aceste probleme enumerate anterior au primit si primesc o atenție activă atât din partea mediului academic, cât și din partea industriei fiind încă subiecte importante de cercetare.

Pe parcursul lucrarii vom aborda majoritatea acestor probleme si vom argumenta in vederea sprijinirii ipotezelor propuse atat prin prezentarea arhitecturala formala precum si prin enumerarea unora din rezultatele cercetarilor ce au fost applicate cu success in industrie si sunt folosite commercial la scara larga.

# Introducere

## Analiza aplicativitații cercetarii si dezvoltarii propuse in viața reală

Principala motivație din spatele muncii noastre a fost înrădăcinată în **nevoia**reală de **a converti aplicațiile vechi din medii vechi de execuție**– cum ar fi sistemele bazate pe baze de date desktop perimate sau bazate pe arhitecturi clasice client-server – **în sisteme bazate pe infrastructuri de tip Cloud**. Acest obiectiv particular are rădăcini multiple în ​​nevoia de gestionare eficienta a resurselor si este tratat in industria software prin abordarea clasica a gestionarii ciclurilor de dezvoltare software pentru migrarea aplicatiilor. In acest context este necesar sa enumeram rapid cateva din elementele cheie ce ingreuneaza si de multe ori blocheaza procesul de migrare. Din perspectiva motivației managementului resurselor, putem enumera atât costurile de contractare și de gestionare a echipei bazate pe dezvoltatori si experti in dev-ops, costuri necesare dezvoltării software pentru migrare, cât și costurile de oportunitate legate de timpul implicat în acest proces particular *(de mai jos* *descrie abordarea clasică a migrarii și costurile directe asociate)*. Aspectele tehnice aditionale ce trebuie luate in consideratie la analiza abordarii clasice sunt în mare parte legate de riscurile asociate cu potențiale module cu erori, probleme de configurare și implementare, problemele de acceptanta si adoptie și așa mai departe.

|  |
| --- |
| *Figura 1 - Abordarea clasică a migrației sistemului - echipa de dezvoltare software rulează un întreg ciclu de dezvoltare pentru a se implementa în noul mediu țintă (poate) bazat pe Cloud Computing* |

Pe langa aceasta problema reala a migrarii catre medii si infrastructuri Cloud, in urma analizei nevoilor din viața reală existente actualmente in piata, au fost identificate si alte cazuri importante conectate în domeniile automatizării și prototipării rapide a aplicatiilor si interfetelor grafice cum ar fi folosirea agenților inteligenți pentru automatizarea interacțiunii operator-*GUI* prin procese robotizate (*Robotic Process Automation*). Pentru a fi și mai concreti vom analiza putin mai detaliat un caz real aferent acestei directiei de automatizare a procesului de interactiune om-interfață prin inlocuirea agentilor de automatizare bazati pe scriptarea euristica a comenzilor cu agenți inteligenți. Vom analiza in continuare comportamentul unui agent de automatizare euristic ce trebuie sa actioneze o apasare de mouse pe un buton al interfeței cu utilizatorul. Putem rezuma sarcina în limbaj natural după cum urmează, pentru simplitatea explicației, deși de obicei este descrisă în limbaje de script: *„MUTAȚI mouse-ul în această zonă dreptunghiulară absolută a ecranului și efectuați operația CLICK”*ce se traduce in a comanda mouse-ului să se deplaseze într-o anumită locație și sa genereze un eveniment de tip *clic* pe interfața cu utilizatorul fără a necesita nici o intervenție umană*.*Urmatorul pas este sa va imaginați că, în timp ce locația absolută rămâne aceeași, butonul pe care trebuie să facem *clic* automat se mută cu forma sa datorită anumitor circumstanțe – spre exemplu mutarea *ferestrei* in care este respectivul buton sau rescalarea *ferestrei* aplicatiei. Pornind de la acest exemplu concret, obiectivul nostru este acela de a utiliza agenți inteligenți care să **recunoască vizual** în exemplul de mai sus dacă *formularul* *GUI* ***a fost mutat***în alt loc pe ecranul interfeței sau că aspectul vizual al zonelor de introducere de date a fost modificat. Pentru claritate in figurile urmatoare este ilustrat acest caz.

|  |
| --- |
| *Figura 2 – agentul RPA simplu preprogramat pe baza unor reguli pentru o anumită sarcină merge la anumite coordonate și completează informațiile necesare. Sarcina este indeplinita cu succes* |

|  |
| --- |
| *Figura 3 – In cazul urmatoru agentul RPA bazat pe euristici scriptate nu poate finaliza sarcina deoarece mediul și-a schimbat parametrii și euristica nu se mai aplică (echivalent: fereastra s-a mutat de la coordonatele preconfigurate și evenimentul „clic” implicit generează eroare)* |

În finalul conturarii contextului motivatiei cercetarii si dezvoltarii ce sta la baza lucrarii prezentăm o listă de cazuri de utilizare realiste bazate pe analiza nevoilor utilizatorilor legate de diferite scenarii, analiza realizata in cadrul etapelor initiale ale cercetarii doctorale:

1. Îmbogățirea și îmbunătățirea proceselor aferente automatizării interfetelor GUI, procese furnizate în prezent de sistemele RPA prin metode euristice. După cunoștințele noastre in domeniul RPA majoritatea implementarilor si sistemelor din piata se bazeaza pe metodele de scriptare euristica. Pentru acest caz particular ne propunem avansarea cercetarii si tehnologiei in directia utilizarii de agenti inteligenti cu capabilitate de intelegere semantica a GUI-urilor. Doar recent (2020) companiile din acest domeniu au promovat noi cercetari si abordări bazate pe *Computer Vision* si invatare automata profunda pentru sarcina de inferenta a componentelor interfețelor grafice. Acest domeniu se află încă în faza de cercetare și experimentare la momentul redactării acestui raport.
2. Migrarea, traducerea și întreținerea automată si la scară largă a sistemelor software vechi, cum ar fi sisteme complexe de contabilitate client-server/ER. sisteme bancare sau alte sisteme software dezvoltate si *mostente* in urma cu 15-30 de ani. Aplicațiile potențiale sunt multiple, pornind de la necesitatea de a traduce un sistem invechit dezvoltat pentru un anumit mediu de tip desktop intr-o noua infrastructura Cloud (un exemplu de bază ar fi dezvoltarea unei noi versiuni a unui sistem de contabilitate moștenit/invechit care trebuie sa migreaze dintr-un sistem bazat pe terminale text in interfete GUI online Cloud) până la necesitatea de a înțelege rapid și la fața locului interacțiunea și comportamentul utilizatorului în cadrul unui sistem complex.
3. Prototiparea rapidă în limbaje de programare a elementelor de experiența a utilizatorului și in particular a programarea interfeței cu utilizatorul a fost în mod constant o cerință importantă în dezvoltarea *front-end*. Deși această sarcină specială este strâns legată de cea anterioară de traducere și migrare, este totuși un scenariu diferit mai ales prin prisma situatiei in care „intrarea” sistemului nu este un ecran al unei aplicatii si un mokup de aplicatie neexistenta (realizat cu unelete de grafica pe calculator sau chiar desenat manual). O altă cerință potențială și constrângere a acestui caz de utilizare este cerința pentru dezvoltarea fără cunostinte de programare – *code-less GUI development* – adică dezvoltarea unei interfețe grafice cu utilizatorul fără cunoștințe prealabile de limbaje de programare.
4. Nu în ultimul rând, unul dintre cazurile de utilizare din viața reală se referă la înțelegerea avansată și automatizata a rapoartelor, formularelor și tabelelor electronice. În diverse domenii atât orizontale, cum ar fi contabilitatea sau logistica, cât și verticale, cum ar fi sectorul financiar, există o nevoie din ce în ce mai mare de operațiuni mai eficace și mai eficiente prin automatizarea inteligentă a proceselor. Organizațiile folosesc diverse sisteme pentru a produce date în diferite formate și etape ale unui anumit proces, toate acestea sub forma de rapoarte sau formulare citibile de catre actori umani ai procesului. Exemple reale de cazuri de utilizare sunt companiile care generează facturi fizice sau electronice sau documente de raportare a tranzacțiilor și apoi le livrează partenerilor lor, care, la rândul lor, au opțiunea de a extrage manual sau automat informații structurate fără a avea la dispozitie un sistem de analiza sau inter-schimbare a datelor structurate. În toate aceste scenarii, există o nevoie din ce în ce mai mare de post-procesare avansată a datelor dincolo de recunoasterea de baza a caracterelor (OCR). Pentru o explicație mai clară vom analiza rapid două cazuri particulare:
5. colectarea și prelucrarea automată a datelor pe baza formularelor web online. Acest prim caz este un proces de analiză, inferență și traducere a interfeței cu utilizatorul care, în acest scenariu, va genera o ieșire de tip *cod scriptat*care poate varia de la simplu JSON la HTML5 mai complex ( [https://html. spec.whatwg.org/multipage/](https://translate.google.com/translate?hl=en&prev=_t&sl=en&tl=ro&u=https://html.spec.whatwg.org/multipage/) ) și până la PHP ( [https://www.php.net/docs.php](https://translate.google.com/translate?hl=en&prev=_t&sl=en&tl=ro&u=https://www.php.net/docs.php) ) sau alte limbaje de script pentru aplicații web.
6. digitizarea și preprocesarea datelor structurate aferente tabelelor și formularelor pre-tipărite sau scanate, cum ar fi rapoartele financiare lunare sau alte date similare nestructurate (imagine). În acest scenariu particular, obiectivul este de a genera o reprezentare digitală multiplatformă, cum ar fi valorile separate prin virgulă (CSV) ale unui document tipărit, sau cum ar fi o foaie de calcul tipărită care nu este disponibilă în formă digitală editabilă.

## Vederea de ansamblu a propunerii de arhitectura si implementare

In cadrul domeniului Inteligenței Artificiale s-au realizat salturi de inovare considerabile în ultimii ani, în principal datorită îmbunătățirilor continue in subdomeniul invatarii profunde automate și, de asemenea, datorită adoptării pe scară largă în cadrul comunității de cercetare și dezvoltare AI a calculului paralel bazat pe GPU și, nu in ultimul rand, datorita proliferarii bibliotecilor publice de seturi de date pentur o multitudine de domenii tinta ca medicina, vehicule autonome, securitate si siguranta, intelegerea limbajelor naturale, s.a.m.d. În multitudinea de domenii tinta de cercetare existente și potențiale pe care le adreseaza IA trebuie să menționăm un domeniul important si anume cel al dezvoltării sistemelor software și al automatizării întreținerii sistemelor software. Acest domeniu este considerat, de majoritatea cercetatorilor si practicienilor, un domeniu de cercetare din categoria „Sfântul Graal” datorita incapacitatii chiar si a celor mai complexe modele de a rationa si intelege procesele de analiza, proiectare si imlementare pe care le parcurge un actor uman si in particular un programator software.

Viziunea noastră, prezentată în continuare în cadrul acestei lucrari, a fost să cercetăm și să dezvoltăm sisteme cu adevărat inteligente capabile să analizeze fluxurile video din experiența utilizatorului – deduse din ecrane-martor ale interactiunii operatorului uman cu interfetele grafice GUI – fluxuri provenite din diverse surse și, în final, să inferam analize reale și utilizabile, inclusiv detalii reale la nivel de cod sursa ale respectivelor interfețe observate, cum ar fi exemplul simplu descris în figura de mai jos.

|  |
| --- |
| *Figura 4 - De la o aplicație desktop cu un singur formular la un script de design vizual (cum ar fi HTML, JSON etc.) si până la codul sursă real într-un limbaj de programare țintă* |

Elementul cheie al cercetarii propuse este acela de a obtine un model neural bazat pe invatare automata profunda sau o suita de astfel de modele incapsulate intr-o banda de asamblare cu scopul de a realiza inferența și analiza unei scenei artificiale reprezentate de o imagine a interfeței cu utilizatorul. Pe o perioadă de peste 3 ani am cercetat și dezvoltat diverse experimente care sunt menționate în cadrul tezei, cu accent deosebit pe cercetările și experimentele din ultima perioada. Un alt obiectiv al perioadei de cercetare din cadrul studiilor doctorale a fost analiza și compararea cercetărilor și dezvoltarii experimentale cu alte cercetări similare și alte inițiative existente în aceasta zona.

In cadrul lucrarilor stiintifice publicate pe durata studiilor doctorale, ca de altfel si in cadrul lucrarii, argumentam ca tehnicile si cercetarile specifice stadiului actual al cercetarii in zona modelelor neurale de imagistic sunt focusate, construite si antrenate pe baza seturilor de imagini naturale. Datorită acestui fapt, modelele DAG (convolution directioned acyclic graph) isi vor dezvolta în timpul procesului de antrenament *detectori* de caracteristici ale imaginilor naturale, cu excepția modulelor neurale initiale din fluxul grafului orientat care vor învăța caracteristicile vizuale primitive de bază (linii, puncte, colori, etc). Ca urmare, una dintre zonele de interes ale cercetării și experimentării din cadrul studiilor doctorale a fost cercetarea si dezvoltarea modelelor DAG specializate pentru recunoașterea imaginilor sintetice și antrenarea acelor modele pe propriile noastre seturi de date experimentale.

In final modelele neurale propuse si antrenate „end-to-end” sunt în astfel capabile să deducă detalii funcționale ale *experienței utilizatorului* – primitive de GUI – din imaginile sintetice de intrare propuse. Aceste informatii pot fi ulterior procesate pentru a fi traduse automat în codul sursă operațional intr-o platforma țintă, cum ar fi HTML/JavaScript.

În cadrul lucrarii prezente este abordat din anumite perspective stadiul actual al cercetarii si tehnicii în două domenii diferite, dar conectate: cel al modelelor de învățare profundă automata pentru imagistica computerizată și zona procesarii si calcului numeric paralel bazat pe GPU cu scopul de instruire eficienta a DAG-urilor (Directed Acyclical neural Graph) si al operaționalizarii in productie. În continuare, vom prezenta arhitectura întregului nostru experiment end-to-end, inclusiv lucrările si abordarile timpurii bazate pe arhitecturi de modele superficiale și ulterior cele din urmă clase de arhitecturi de modele bazate pe invatare automata profunda. O sub-secțiune specială este dedicată cercetării și dezvoltării setului nostru de date de imagini artificiale care este publicat în regim Open Source pentru a beneficia de potentiala contributie a comunitatii internaționale de cercetare.

În cele din urmă, lucrarea se incheie cu o revizuire a celor mai importante contribuții propuse și a concluziilor tezei, concluzii ce includ potențiale aplicații multiple din viața reală – de la implementarea de sine stătătoare a modelelor noastre până la încorporarea arhitecturilor CloudifierNet in pipeline-uri RPA externe. O observație importantă este că, în ciclurile noastre de cercetare și experimentare, am proiectat și implementat două versiuni diferite ale CloudifierNet și anume (v1 denumit *CloudifierNetV1*, v2 denumit *CloudifierNetV2*) și pentru fiecare versiune am implementat mai multe sub-versiuni *ajustand*dimensiunea graficului și cerințele sale de calcul si resurse GPU.

## Experimente vs. aplicativitate reala: provocări și așteptări

Inca de la inceput trebuie precizat ca rezultatele noastre experimentale finale au fost operaționalizate într-un prototip de sistem funcțional care a fost re-utilizat in cadrul unor aplicatii cu utilitate directa industriala. Totodata, in limita resurselor disponibile, o echipă de cercetători și ingineri de date lucrează la îmbunătățirea performanței benzii de asamblare de modele și a mediului de execuție rapidă.

Una din cele mai importante probleme pe care am incercat sa le tratam cu atentie afost aceea a reajustarii continue a asteptarilor. În timpul cercetarii si experimentelor din perioada studiilor doctorale ne-am confruntat cu mai multe probleme care separă în mod clar rezultatele muncii noastre experimentale de aplicarea la nivel real si la scara larga. Analiza acestor provocări individuale este importantă atât din perspectiva stabilirii așteptărilor corecte de la modulele experimentale realizate, stabilirea așteptărilor versus ambițiie de cercetare, cât și a previzualizării lucrărilor ulterioare care pot fi făcute pentru a îmbunătăți rezultatele actuale:

1. ***Domeniul limitat de aplicabilitate al modelelor si modulelor cercetare si dezvoltate, domeniu determinat de generarea/producția setului de date******sintetice*** *de*: Una din activitatile propuse de cercetare și experimentare a fost aceea de generare a Setul de date artificiale (AD) – și anume imaginile primitivelor/comenzilor proprii interfeței cu utilizatorul GUI precum și capturile complete de ecran ale interfeței cu utilizatorul. Acestea nu pot, sub nicio formă, să surprindă toate variațiile potențiale ale interfețelor utilizator ale oricărui standard sau abordare de UX (experiență de utilizare) disponibil anterior sau în prezent. În teza noastră, precum și în lucrările publicate, punem accent pe selecția efectivă a mai multor standarde de interfață cu utilizatorul, cum ar fi aplicațiile vechi Microsoft Windows bazate pe MFC, Delphi sau alte medii de dezvoltare similare. Cu toate acestea, un set de date universal și, prin urmare, un pipeline de modele potențial aplicabil universal depășește domeniul de aplicare al activității noastre.
2. ***Aplicarea nefezabila a rezultatelor proiectului experimental la sisteme și aplicații ce prezinta abordări netradiționale ale UX/GUI (experienței utilizatorului) cu interfețe de utilizator care nu respectă regulile vizuale și funcționale clasice****:*Ca o problemă complementară celei prezentate anterior o avem și pe cea privind imposibilitatea suitei noastre de modele neuronale de a „înțelege” interfețele utilizator care nu urmează abordări comune în ceea ce privește fluxul experienței utilizatorului.  Un astfel de exemplu sunt interfețele în care, să zicem, „butoanele” nu urmează niciun model vizual pe care îl au de obicei butoanele interfeței grafice cu utilizatorul in cadrul seturilor de date folosite la antrenare.
3. ***Necesitatea potențială de a limita mediul de dezvoltare țintă la aplicații bazate pe web care nu necesită funcționalități complexe de partea clientului***: În sfârșit, în urma numeroaselor experimente și analize de cazuri reale de utilizare, am ajuns la concluzia că din multitudinea de medii țintă potențiale (cum ar fi mobil-Android, mobil-iOS, web, MS-windows, Unity etc.) ne vom concentra pe aplicațiile web cu abordarea clasică model-view-controller limitând utilizarea bibliotecilor complexe de pe partea client (cum ar fi jQuery) la chestiuni simplu și ușor.

# Rezumatul tezei

În această secțiune vom prezenta pe scurt principalele aspecte descrise în fiecare dintre secțiunile principale ale tezei cu referiri la capitolele individuale respective.

## Descriere ipotezelor si a motivatiei

Motivația din spatele tezei și activităților sale de cercetare și dezvoltare – prezentate cu cazurile de utilizare din industrie în ***Capitolul 1.1***- a fost aceea de a construi o abordare eficientă și viabilă pentru analiza și recunoașterea cu o granularitate fină a imaginilor specifice interfețelor cu utilizatorul (GUI) fără utilizarea de euristici descrise de script-uri sau logica programabilă.

Identificarea punctul de plecare al ancorarii in nevoia reala si motivația din spatele lucrarii si a ipotezelor propuse poate fi urmărită în diverse cazuri de utilizare din industrie cat și in experimente academice – de la nevoia de automatizare a proceselor robotizate cu autoadaptare sau nevoia de a converti aplicațiile moștenite (legacy) în noi infrastructuri de tip Cloud Computing fără insa a avea disponibilitatea codului sursă inițial. Alte cazuri reale de utilizare în industrie care merită menționate sunt recunoașterea formularelor electronice completate sau a șabloanelor de formulare electronice sau extragerea sau introducerea datelor sau prototiparea rapidă a interfețelor moderne de utilizator fără a fi nevoie de instrumente de proiectare software, cum ar fi instrumentele de dezvoltare rapidă a aplicațiilor.

Pentru a avea o vedere mai concretă și o mai bună înțelegere, vom lua, analiza si împărți câteva cazuri de utilizare în definiția problemei, ipoteza soluției propuse și aplicația reală la nivelul industriei:

1. Recunoașterea vizuală automată de computer a interfețelor GUI fără acces la regulile predefinite sau euristici programabile
   * **Problemă**: Având în vedere capturile de ecran ale aplicațiilor vechi, interfețele utilizatorului generează o colecție de artefacte, locații și scopuri/functiuni de interfață.
   * **Ipoteza**: Folosind modele neurale detectăm și localizăm fiecare dintre obiectele interfeței GUI
   * **Aplicație**: Automatizarea proceselor robotizate beneficiază de acest caz de utilizare, renunțând la necesitatea regulilor scriptate și a euristicilor, reducând astfel timpul de implementare și configurare și crescând drastic viabilitatea generală și invariabilitatea dimensiunii/locației/aspectului elementelor GUI.
2. Prototiparea rapidă a interfețelor utilizator GUI funcționale, bazată pe interfețe vizuale vechi sau simple machete
   * **Problemă:**Plecand de la ecrane si/sau machete de interfață cu utilizatorul, capturi de ecran ale aplicațiilor moștenite sau simple schite facute manual se generează codul de script complet al interfeței cu utilizatorul
   * **Ipoteza:**Folosind modele neurale adanci se analizează imaginile de intrare și generează direct secvențe de cod text (script)
   * **Aplicație**: prototipare rapidă de-la-desen-la-cod a interfețelor utilizator chiar si de către non-programatori – deci fără cunoștințe anterioare ale limbajului de programare tinta– sau designeri vizuali și generarea de cod sursă pentru o platformă Cloud;

## Rezumarea stadiului actual al cercetarii

Analiza stadiului actual al tehnicii și modul în care aceasta se leagă de cercetarea prezentata in lucrare este descrisa în ***Capitolul 2***al tezei. Mai multe domenii si subiecte de cercetare active in comunicatea stiintifica, subiecte strâns legate de zonele de cercetare propuse, sunt prezentate si analizate in detaliu: abordări bazate pe GPU pentru optimizarea calculului numeric și arhitecturi avansate pentru construirea de grafuri aciclice direcționate pentru imagistica computerizata și prelucrarea limbajelor naturale.

Adoptarea la scara larga a unitatilor de calcul (PC, laptop, etc) bazate pe GPU-uri cu capabilitati de calcul considerabil, recenta evoluție rapidă a acestor placi hardware și noile adăugări in special in familia Nvidia si nu numai (cum ar fi cele mai recente TPU - Tensor Processing Units) au permis atât comunității academice, cât și celei comerciale să proceseze cantități mari de date folosind paralelizare la un nivel de eficienta superior algoritmi numerici. Operațiile de convoluție discrete facil vectorizabile si paralelizabile, calculul paralel al mai multori fluxuri de date in grafurile tensoriale, arhitecturile de tip *Transformer* sunt doar câteva exemple în care calculul numeric paralel bazat pe GPU/TPU a contribuit major la avansul cercetarii si aplicativitatii acesteia. 

În munca noastră, câteva zone speciale și componente individuale au primit o atenție specială în ceea ce privește optimizarea calculului numeric paralel: noua abordare propusă pentru sub-grafuri cu mecanisme de tip porți și grafurile neurale pentru recunoașterea scenelor artificiale.  Deși construcția *kernel*-urilor GPU este un subiect incapsulat de bibliotecile de calcul și optimizare tensoriala*,*cum ar fi *Tensorflow* sau *PyTorch*, am menționat tehnicile și abordările atat in continutul tezei cat si in lucrarile publicate la inceputul studiilor doctorale.

Arhitecturile de învățare profundă rezultate din cele mai recente cercetări de ultimă oră sunt a doua zonă principală de lucru conexe prezentată în ***Capitolul 2.1***, precum și în ***Capitolul 2.2***. În primul rând, am inventariat abordările clasice cunoscute în Computer Vision concentrându-ne pe abordări mai recente bazate pe grafice aciclice orientate profund. Deși principalele arhitecturi de viziune profundă prezentate se referă în mod direct la diferite probleme cunoscute și cazuri de utilizare, principalul obiectiv al experimentelor noastre și, prin urmare, rezultatele prezentate sunt concentrate pe inferența scenei interfeței grafice cu utilizatorul (localizarea și detectarea obiectului), precum și pe segmentarea la nivel de pixeli. [[CMM20]](https://translate.googleusercontent.com/translate_f" \l "_cmnt20)

Un accent secundar a fost îndreptat în zona modelelor generative pentru generarea secvenței codului sursă. Acest lucru a fost realizat atât împreună cu cercetarea pe subiectul abordărilor de segmentare a scenei și a subtitrării imaginilor, cât și cu analiza tendințelor moderne în procesarea neuronală a limbajului natural.

Nu în ultimul rând, în cadrul acestui domeniu de cercetare, un subiect important a fost acela de a găsi arhitecturi optime și eficiente pentru graficele aciclice direcționate tensorial. Mai precis, un aspect important al lucrării a fost acela de a găsi abordări inovatoare pentru auto-ajustarea arhitecturilor de graf care să folosească calculul numeric paralel în loc de abordări clasice, cum ar fi căutarea exhaustivă sau aleatorie a spațiului de opțiuni pentru arhitectura grafului (căutare în grilă). În acest domeniu special, lucrările și experimentele anterioare privind mecanismele de auto-învățare au primit o atenție specială în ***Capitolul 2.5***și pe scurt în ***Capitolul 2.2.2***.

Pentru a rezuma principalele domenii de activitate conexe avem următoarele:

1. **Evaluare**foarte eficientă a **graficului tensor, bazată pe**calculul paralel numeric **GPU**
2. Arhitecturi de viziune profundă pentru **localizarea și segmentarea artefactelor**
3. Metode generative de traducere **imagine-în-cod-secvență**
4. **Auto-învățare a hiperparametrului grafic**pe baza reglajului real al parametrilor graficului folosind obiective de optimizare

### 1.2.3 Rezumatul capitolului Arhitectură

Întregul proces de cercetare și experimentare s-a bazat pe o abordare ciclică și avansare pas cu pas pornind de la abordări simple către soluții mai profunde și mai complexe pentru obiectivele propuse. Acest proces a condus la două tipuri de rezultate, așa cum este descris în ***Capitolul 3.1***: rezultatele și livrabilele specifice domeniului, precum și rezultatele care pot fi aplicate în mai multe domenii și care au fost deja aplicate la mai multe cazuri de utilizare industrială din viața reală. Procesul de proiectare detaliat și abordarea arhitecturală cu toate detaliile lor sunt descrise pe deplin ***în Capitolul 3***al tezei. Pașii inițiali ai procesului de cercetare și experimentare s-au bazat în întregime pe abordări de bază „superficiale” (adică abordări de învățare automată care nu au straturi de grafice ascunse) [[CMM21]](https://translate.googleusercontent.com/translate_f" \l "_cmnt21) de utilizare a modelelor de regresie în mediul de calcul numeric paralel pentru a evalua opțiunea de a avea ansambluri de modele simple. Practic, abordarea inițială „de bază”, bazată pe modele simple de învățare automată, [[CMM22]](https://translate.googleusercontent.com/translate_f" \l "_cmnt22) a fost aceea de a folosi calcule paralele cu produse punctuale ale spațiului de intrare – reprezentarea imaginii RGB a interfeței cu utilizatorul – cu matrice de greutate reprezentând o potențială interfață cu utilizatorul primitive, toate acestea folosind mecanisme 2D cu ferestre glisante. Mai concret, am folosit capacitățile de calcul numeric paralel al GPU pentru a calcula mai multe ipoteze 1-vs-all pentru toate regiunile potențiale ale imaginii interfeței cu utilizatorul țintă.

Urmând această abordare destul de naivă a inferenței scenei de captură de ecran a interfeței cu utilizatorul, am început să dezvoltăm abordări mai complexe bazate pe grafice aciclice direcționate cu module de convoluție discrete și separabile – această abordare arhitecturală fiind prezentată pe deplin în ***Capitolul 3.6***. În acest proces de rafinare a arhitecturilor, obiectivul nostru a fost acela de a construi o clasă bine optimizată și echilibrată de grafice aciclice orientate profund în ceea ce privește performanța vs costuri și consumul de energie. Astfel, am cercetat potențialul beneficiu al construirii de subgrafe auto-învățate bazate pe mecanisme de învățare care sunt explicate și formalizate în ***Capitolul 3.3***. Această abordare inovatoare ar permite graficului aciclic direcționat să-și adapteze propria funcționare și structura fluxului de date, pe baza obiectivului general de optimizare. Câteva aspecte importante privind arhitectura grafică aciclică direcționată bazată pe module convoluționale sunt prezentate în continuare în ***Capitolul 3,***cum ar fi analiza opțiunilor de învățare prin transfer pe care am explorat-o în ***Capitolul 3.4***, comparația dintre antrenamentul cu date artificiale versus imaginile generate manual de oameni în ***Capitolul 3.5.***și detalii despre procesul de optimizare în ***Capitolul 3.7***.

Pentru a sintetiza principiile principale utilizate în proiectarea arhitecturii noastre experimentale, putem rezuma următoarele:

1. Un set de **obiective și principii**de **bază**a fost stabilit de la bun început, cum ar fi utilizarea unor infrastructuri eficiente de calcul numeric paralel pe piața de masă bazate pe GPU
2. Experimentele au fost **începute de la**abordări de **nivel scăzut, complexitate scăzută**pentru a stabili linii de bază clare
3. **Complexitatea a fost adăugat crescut treptat**și mai multe provocări au fost abordate iterativ
4. În cele din urmă, arhitectura grafică de auto-învățare bazată pe mecanisme de auto-gating a permis generarea familiei noastre DAG

### 1.2.4 Rezultate notabile ale cercetării

Există trei domenii diferite în care munca noastră a produs rezultate validate, cu mențiunea importantă că unele dintre ele sunt utilizate în prezent în implementări de grad de producție ale diferitelor cazuri de utilizare a industriilor. Probabil cel mai important și relevant pentru industrie este propunerea unei abordări inovatoare de reglare a hiperparametrilor graficelor aciclice direcționate bazată pe auto-învățare în loc de căutarea clasică exhaustivă sau aleatorie în grilă în spații hiperparametrice, în esență mecanismul **Multi***self-***Gating**. Această inovație specială a descris în detaliu ***Capitolul 3.3***, utilizat în prezent într-o serie de sisteme de grad de producție în domeniul analizei predictive – prognoza cererii și predicția evenimentelor – demonstrând astfel aplicarea inter-industrială. [[CMM23]](https://translate.googleusercontent.com/translate_f" \l "_cmnt23) Agnosticismul de domeniu al acestei inovații propuse arată foarte promițător și sunt planificate lucrări suplimentare în acest domeniu în perioada următoare. Mai mult, în acest domeniu de cercetare, un accent deosebit a fost îndreptat către calculul ecologic – adică **eficiența energetică**– precum și **explicabilitatea experimentului**– unul dintre cele mai fierbinți subiecte din peisajul învățării automate de astăzi.

Al doilea domeniu general de cercetare în care rezultatele noastre au fost validate fie prin cercetări și experimente publicate și/sau prin dezvoltare experimentală industrială este cea a analizei imaginii artificiale-scenă (de exemplu: captură de ecran a interfeței cu utilizatorul) și generarea de cod sursă (script). În acest domeniu special, am dezvoltat - al treilea rezultat principal - un nou set de date care este disponibil public pentru experimentare, precum și o metodologie pentru extinderea potențială a acestui set de date propus. Pe lângă setul de date, munca noastră s-a concentrat pe găsirea de arhitecturi eficiente de grafice aciclice direcționate care să atingă cu succes atât obiectivele de segmentare a imaginilor interfeței cu utilizatorul, cât și să genereze cod sursă (sau script) de bază pentru proiectarea interfeței cu utilizatorul.

Ca o sinteză a celor mai importante rezultate ale cercetării, avem următoarele trei puncte principale:

1. Propunere pentru noua **unitate de**subgraf **multi-gate**care permite auto-învățarea structurii topologiei grafului (reconfigurarea nodurilor și arcelor)
2. **Arhitectură**eficientă a modulelor convoluționale discrete **pentru deducerea scenei artificiale**
3. ***Set de***date de scenă artificială disponibil public – un ***ImageNet*pentru interfețele utilizator**

### 1.2.5 Experimente [[CMM24]](https://translate.googleusercontent.com/translate_f" \l "_cmnt24)

În ceea ce privește experimentarea, așa cum va fi prezentat în mod clar pe parcursul ***capitolului 3***și mai precis în ***capitolul 4***, am avut două tipuri diferite de experimente: cele care au legătură directă cu obiectivul principal al tezei, precum și experimente, ambele pur academice. sau sisteme industriale actuale, legate de inovațiile propuse care sunt agnostice față de obiectivele noastre propuse. [[CMM25]](https://translate.googleusercontent.com/translate_f#_cmnt25) Pentru a menționa rapid principalele aplicații din industrie, trebuie să menționăm că avem atât cazuri de utilizare legate de viziunea profundă în domeniul medical, cât și cazuri de utilizare a analizei predictive.

Pentru prima categorie de experimente - cele legate de inferențe pe ecranul interfețelor utilizator - munca noastră, prezentată în rezultatele legate de ***Capitolul 3.4***, s-a concentrat pe evaluarea corectă a performanței arhitecturilor de graf aciclice direcționate propuse asupra datelor colectate, precum și pe creșterea iterativă. calitatea și cantitatea *setului de*date *CloudifierNet*. Pe lângă măsurarea performanței în detectarea și localizarea artefactelor interfeței cu utilizatorul, o zonă secundară de experimente a vizat scalarea și eficiența calculului. Mai concret, am experimentat cu diferite dimensiuni de grafic – de la o dimensiune minimă până la o extindere de 2x, 3x a nodurilor și arcelor de grafic - precum și diferite capacități de calcul, cum ar fi scalarea de la o infrastructură încorporată de 256 de nuclee numerice la peste 4000-8000 de elemente numerice. infrastructuri de bază întâlnite în mod obișnuit în GPU-urile moderne.

În această clasă specială de experimente și în special legate de setul de date *CloudifierNet*propus, am creat, de asemenea, o serie de aplicații experimentale cu diverse instrumente de dezvoltare cu scopul principal de a genera interfețe vizuale de utilizator, precum și cod sursă de proiectare a interfeței cu utilizatorul. Mai concret, am proiectat atât aplicații vizuale bazate pe *win32,*cât și aplicații vizuale portabile compatibile cu POSIX, fără niciun proces complex sau logică de afaceri, toate cu unicul scop de a genera scene artificiale pentru setul nostru de date - imagini (capturi de ecran) ale interfețelor potențiale de utilizator pentru diverse cazuri.

Pe lângă toate experimentele legate de inferența scenei artificiale menționate anterior, am urmărit o conductă de experimentare secundară pentru inovațiile agnostice de proiect propuse, prezentate inițial în ***Capitolul 3.3***și evaluate în ***Capitolul 4.4***. Un accent deosebit în acest domeniu a fost acela de a experimenta cu diferite arhitecturi de graf aciclice direcționate cu sau fără utilizarea unității noastre *Multi Gated*propuse . Obiectivul principal al acestor experimente a fost acela de a testa îmbunătățirea performanței pe sarcini simple de clasificare (cum ar fi clasificarea imaginilor MNIST cu un singur canal ale numerelor scrise de mână) pe care le primește un grafic aciclic direcționat clasic după ce a fost augmentat de *unitatea*noastră *Multi Gated.*Obiectivul secundar al experimentelor din acest domeniu a fost acela al auto-explicabilității: extragerea automată a informațiilor de gate și efectuarea – dacă este posibil – de operațiuni de tăiere a porții, scăzând astfel numărul de înmulțiri de matrice necesare procesării întregului grafic de calcul. Această abordare de tăiere propusă ar permite obținerea de subgrafe care nu sunt auto-gate din modulele în care porțile sunt complet închise sau deschise.

### 1.2.6 Rezumatul principalelor contribuții personale

Direct legat atât de rezultatele experimentării, cât și de arhitecturile și conductele de grafice aciclice direcționate subiacente propuse, în ***Capitolul 5***avem următoarea listă de contribuții principale la cercetare și inovare, precum și aplicații practice implementate în scenarii din viața reală:

1. ***Multi Gated Unit***: Propunerea unei noi abordări inovatoare pentru găsirea structurii grafice optime (hiperparametri ai graficului de auto-învățare) direct prin reducerea la minimum a funcției obiective a sarcinii. Această contribuție specială este [independentă de](https://translate.googleusercontent.com/translate_f#_cmnt26) domeniu și a fost testată atât în sarcinile legate de *viziunea profundă*[[CMM26]](https://translate.googleusercontent.com/translate_f" \l "_cmnt26) , cât **și implementată cu succes în sistemele de analiză predictivă la nivel de producție**. Cele două rezultate principale ale acestei contribuții de cercetare pot fi rezumate după cum urmează:
   1. Eliminarea necesității abordărilor de căutare în rețea care ar necesita executarea a milioane de experimente pe infrastructura de calcul paralelă și reducând astfel drastic **amprenta de carbon**a procedurii de căutare a arhitecturii grafice
   2. Capacitatea de autoexplicare a structurii interioare a modelului pe baza activărilor porții din fiecare unitate Multi Gated pentru întregul grafic. O altă vedere a vederii acestei descoperiri este aceea de a avea o metodă pentru definirea unei structuri unice (hiper-parametri) pentru fiecare nod individual al unui graf aciclic orientat profund.

1. ***Setul de date CloudifierNet***: Publicarea Open Source a unui nou set de date care permite cercetarea și experimentarea în domeniul recunoașterii automate a conținutului interfeței cu utilizatorul. Setul de date poate fi utilizat într-o gamă largă de experimente, cum ar fi: antrenarea agenților pentru a recunoaște elementele interfeței cu utilizatorul în scopul de a simula/genera mesaje de aplicație, de a genera cod sursă de funcționalitate de bază pentru proiectarea aspectului sau chiar de a construi abordări generative (cum ar fi *rețele adverse generative*) care ar putea automatiza procesul de proiectare pe lângă crearea codului sursă.

1. ***Arhitecturi CloudifierNet***: Proiectele de grafice ***aciclice***direcționate propuse se bazează pe îmbunătățiri incrementale pe deasupra mai multor arhitecturi de ultimă generație, utilizând și modulele multi-gate.

1. ***Aplicații din viața reală pentru domeniul medical***: [[CMM27]](https://translate.googleusercontent.com/translate_f" \l "_cmnt27) Folosind arhitectura CloudifierNet propusă, în ultimii 2 ani, am reușit să dezvoltăm și să propunem sisteme de grad de producție în domeniul oncologiei precum:
   1. Sistem complet reglat automat bazat pe arhitectura CloudifierNet propusă pentru dermatologi și deducerea leziunilor dermatologice severe
   2. Aplicație capabilă să efectueze analize de colposcopie (sau *cervigramă*) pentru ginecologie oncologică pentru a detecta potențiale leziuni de col uterin și gravitatea acestora

### 1.2.7 Lucru în continuare

În ***capitolul 6***al tezei sunt propuse o serie de îmbunătățiri, lucrări ulterioare aflate în prezent în faza de cercetare și experimentare pentru două domenii diferite. Una dintre domeniile de lucru propuse este specifică sarcinii de procesare a scenei de imagine artificială, în timp ce a doua direcție generală de îmbunătățire este legată de auto-ajustarea hiper-parametrilor.

În domeniul recunoașterii artificiale a scenei, avem mai multe direcții clare de lucru ulterioar planificat, precum și direcții potențiale de explorare, toate descrise în ***Capitolul 6.2***, ***Capitolul 6.3***și în cele din urmă ***Capitolul 6.4***. Principalul motor al activității planificate în această direcție generală este strâns legat atât de cazurile de utilizare din viața reală pe care le *vizam*, cât și de obiectivul academic de îmbunătățire a *setului de*date *CloudifierNet*.

În domeniul auto-ajustării hiper-parametrice, obiectivele noastre, descrise în ***Capitolul 6.8,***sunt împărțite între îmbunătățirea în continuare a câștigurilor de eficiență energetică, precum și promovarea capacității de autoexplicare și auto-tăieri ale unității *Multi Gated*propuse . Credem cu tărie că o *unitate Multi Gate*mai fiabilă și aplicabilă universal va încuraja atât echipele academice, cât și echipele de cercetare din industrie să o folosească, mai degrabă decât să aplice explorarea spațială clasică de căutare în grilă hiper-parametrică. Indiferent de domeniul țintă în care se aplică autoajustarea hiperparametrilor, reducerea amprentei de carbon a căutării optime a arhitecturii grafice va fi redusă dramatic atunci când se generează o singură arhitectură auto-învățată, mai degrabă decât la efectuarea operațiunilor de căutare în spațiul hiperparametrilor. Deși un singur proces clasic de optimizare a graficului profund ar putea avea un cost de energie mult mai mic decât cel al unuia auto-ajustat – datorită multitudinii de calcule adăugate de mecanisme de deschidere în unitatea Multi Gated – utilizarea celui mai târziu implică o ***singură***iterație completă de optimizare a graficului .

Pentru a rezuma în mod clar lucrările noastre ulterioare propuse, putem enumera următoarele obiective pe baza ***capitolului 6***propus al tezei:

1. Recunoaștere avansată a scenei artificiale cu interfața utilizatorului, inclusiv inferența din videoclipuri și inferența comportamentului procesului;
2. Iterații ulterioare de îmbunătățire a *setului de*date *CloudifierNet*și crearea unei comunități academice și industriale în jurul acestui set de date;
3. În sfârșit, îmbunătățirea unității *Multi Gated*atât din perspectiva aplicării cu succes în diverse cazuri de utilizare, cât și în avansarea capacității de autoexplicare și a reducerii de calcul folosind mecanisme de auto-taiere.

Pagină **1** de **25**

[[CMM1]](https://translate.googleusercontent.com/translate_f#_cmntref1) Ar putea fi util macar 1 exemplu apropo de ce înseamnă automate behavior

[[CMM2]](https://translate.googleusercontent.com/translate_f#_cmntref2) Pentru mine este neclar ce înseamnă acest lucru

[[CMM3]](https://translate.googleusercontent.com/translate_f#_cmntref3) pentru mine nu este atat de evidenta legatura intre GUI si RPA

[[CMM4]](https://translate.googleusercontent.com/translate_f#_cmntref4) "thus" pentru mine înseamnă "as a result or consequence of this "; am recitit de cateva ori si imi este neclar cum de aparitia RPA este rezultatul faptului ca nevoile de rethink redesign si automate behavior au fost adresate cu tehnici clasice. Cred ca "thus" se refera la fraza anterioara, adica au aparut noi tipuri de nevoi cu privire la GUI si odata cu aceste nevoi au aparut si RPA; poate ar fi utila o reformulare

[[CMM5]](https://translate.googleusercontent.com/translate_f#_cmntref5) din pacate nu inteleg acest punct (iii)

cred ca ar trebui "than" in loc de "then";

[[CMM6]](https://translate.googleusercontent.com/translate_f#_cmntref6) cred ca fraza este prea lunga si ceva mai greu de urmarit

[[CMM7]](https://translate.googleusercontent.com/translate_f#_cmntref7) din pacate nu imi este clar acest exemplu pentru a scoate in evidenta fapta ca vrem sa inlocuim "scripted" agents cu intelligent ones. Poate ar fi utila sau reformulare.

[[CMM8]](https://translate.googleusercontent.com/translate_f#_cmntref8) cred ca este un typo aici ca nu se intelege

[[CMM9]](https://translate.googleusercontent.com/translate_f#_cmntref9) Abia acum am inteles la ce se referă exemplul din paragraful anterior

[[CMM10]](https://translate.googleusercontent.com/translate_f#_cmntref10) poate ar fi utile 1-2 exemple de astfel de sisteme foarte cunoscute

[[CMM11]](https://translate.googleusercontent.com/translate_f#_cmntref11) imi este neclar asta

[[CMM12]](https://translate.googleusercontent.com/translate_f#_cmntref12) nu imi este clar ce inseamna asta

[[CMM13]](https://translate.googleusercontent.com/translate_f#_cmntref13) 1-2 exemple?

[[CMM14]](https://translate.googleusercontent.com/translate_f#_cmntref14) a fost des folosit deja acest cuvant, insa eu nu am inteles la ce se refera synthetic images, synthethic data set

[[CMM15]](https://translate.googleusercontent.com/translate_f#_cmntref15) poate ar fi de folos 1 exemplu. In mod general acolo unde se face referire la aspectele "common" sau "legacy", eu simt nevoia de 1 exemplu macar.

[[CMM16]](https://translate.googleusercontent.com/translate_f#_cmntref16) si aici simt nevoia unui exemplu vizual (o schita facuta de mana care se transforma in cod). Asta pare Science Fiction si capteaza puternic atentia. Daca se poate arata ceva ar fi foarte util

[[CMM17]](https://translate.googleusercontent.com/translate_f#_cmntref17) cred ca ar fi uitila o reformulare ca este greu de citit fraza asta. poate 2 propozitii sau ";" intre exemplu

[[CMM18]](https://translate.googleusercontent.com/translate_f#_cmntref18) cred ca trebuie o reformulare sau vreo virgula pe undeva intrucat nu se intelege; sau 2 propozitii.

[[CMM19]](https://translate.googleusercontent.com/translate_f#_cmntref19) si aici cred ca ar trebui o virgula fiindca se citeste cam greu

[[CMM20]](https://translate.googleusercontent.com/translate_f#_cmntref20) nu se intelege. cred ca ar fi utila o reformulare sau fraze mai scurte.

[[CMM21]](https://translate.googleusercontent.com/translate_f#_cmntref21) nu imi e clar ce inseamna ca sunt anumite cuvinte intre ghilimele; chiar nu stiu daca este ok dpdv academic; poate ar trebui explicat ce înseamnă shallow approches

[[CMM22]](https://translate.googleusercontent.com/translate_f#_cmntref22) la fel ghilimele; poate ar trebui explicat ce ar trebui sa se inteleaga prin bazelina approach

[[CMM23]](https://translate.googleusercontent.com/translate_f#_cmntref23) cred ca induce in eroare fapt ca propbabil cel mai important rezultat are o aplicabilitate in afara obiectivelor descrise la inceputul lucrarii. Poate ar fi totusi o idee sa scoti in fata rezultate care au cea mai mare legatura si greutate fata de obiectivele principale ale lucrarilor.

Pe mine ma luat prin surprindere acest fapt

[[CMM24]](https://translate.googleusercontent.com/translate_f#_cmntref24) As pune si in finalul acestui sub capitolul sumarizarea facuta ca la celelalte sub-capitole. O consider foart utila.

[[CMM25]](https://translate.googleusercontent.com/translate_f#_cmntref25) din formulare imi pare ca se da o greutate mai mica experimentelor din a 2-a categorie fiindca pare ca nu au legatura cu obiectivele lucrari. Asta mi-a sarit mie in ochi. Poate ar fi utila sau reformulare

[[CMM26]](https://translate.googleusercontent.com/translate_f#_cmntref26) poate ar fi utile 1-2 exemple

[[CMM27]](https://translate.googleusercontent.com/translate_f#_cmntref27) oare nu ar fi bine sa faci referire la aplicabilitatea aceasta si in alte capitole de mai sus?

Pare așa ca a aparut subit aici aplicabilitatea aceasta desi au mai fost sub-capitole mai sus care au trata ideea de real life use-cases si nu a fost mentionata aplicabilitatea medicala;

Cumva pare ori ca nu este importanta (pusa la coada) ori ca nu e suficient de minutios abordata.

Pentru oricine citeste asta cred ca este unmopment wow, insa si un pic de neclaritate fiindca nu sa mai zis nimic despre asta pana acum si pare ca nu are legatura cu obiectivele lucrari.