



UNIVERSITATEA DIN  
BUCUREȘTI

FACULTATEA DE  
MATEMATICĂ ȘI  
INFORMATICĂ



SPECIALIZAREA INFORMATICĂ

Lucrare de licență

**CORECTAREA AUTOMATĂ CU  
AJUTORUL LLM-URILOR : STUDIU  
DE CAZ - BACALAUREAT  
INFORMATICĂ**

Absolvent

Firca Liviu Nicolae

Coordonator științific

Marius Adrian Dumitran

București, Iunie 2025

## Rezumat

Acest studiu se întreabă dacă examenul de bacalaureat în informatică poate să fie corectat automat folosind Large Language Models (LLM). Motivul acestei interogări este pentru că bacalaureatul este un examen foarte important, și LLM-urile ar putea fi mai obiective și consistente decât corectorii umani. De asemenea, foarte mult timp este investit în corectarea acestor lucrări. În plus, studiul a explorat dacă LLM-urile pot genera feedback automat pentru profesori, pe baza corectării LLM-urilor, pentru a sprijini îmbunătățirea predării.

Pentru a determina dacă bacalaureatul poate fi corectat folosind LLM-uri, am măsurat eroarea notelor atribuite de LLM folosind Mean Absolute Error (MAE), și pe baza acestei evaluări a performanței am concluzionat că este posibil să corectezi lucrările de bacalaureat la informatică folosind LLM-uri. Pentru a evalua calitatea feedbackului, am verificat dacă acesta identifică toate greșelile comune ale studenților și dacă este corect din punct de vedere al conținutului.

## Abstract

This study investigates whether the computer science baccalaureate exam can be graded automatically using Large Language Models (LLMs). The motivation for this is that the baccalaureate is a very important exam, and LLMs could potentially be more objective and consistent than human graders. Additionally, a significant amount of time is invested in grading these papers. Furthermore, this study explores if LLMs can correctly generate feedback for teachers, on the basis of the grading provided by LLMs, in order to enhance their teaching.

To determine whether the baccalaureate can be graded using LLMs, the error of the grades assigned by the LLM was measured using Mean Absolute Error (MAE), and based on this performance evaluation, it shows that LLMs can be effectively used to grade computer science responses on the baccalaureate exam. In order to evaluate the quality of the feedback, we checked if it contained all the common errors made by the students and if the feedback actually contained any mistakes that the students didn't make.

# Cuprins

<b>1</b>	<b>Introducere</b>	<b>4</b>
<b>2</b>	<b>Preliminarii</b>	<b>6</b>
<b>3</b>	<b>Colectarea si prelucrarea datelor</b>	<b>8</b>
3.1	Colectarea datelor . . . . .	8
3.2	Prelucrarea datelor . . . . .	9
<b>4</b>	<b>Corectarea automata si feedback automat</b>	<b>10</b>
4.1	Corectarea automata . . . . .	10
4.2	Feedback automat . . . . .	12
	<b>Bibliografie</b>	<b>15</b>

# Capitolul 1

## Introducere

Bacalaureatul reprezintă unul dintre cele mai importante examene din sistemul de învățământ, oferind accesul la studiile universitare. Admiterea în anumite instituții de învățământ superior, precum și acordarea burselor în primul an de studiu, se pot baza pe rezultatele obținute la examenul de Bacalaureat, demonstrând astfel importanța acestora. Din acest motiv, obiectivitatea și acuratețea în corectare sunt necesare. În plus de acest lucru, profesorii investesc foarte mult timp pentru a corecta lucrările de bac. Aproximativ 100.000 de elevi dau bacul în fiecare an, și corectarea unei lucrări în general este lentă.

O tehnologie foarte promițătoare care ar putea a priori rezolva această problemă este Large Language Models (prescurtat LLM). Ele sunt o funcție matematică cu scopul de a procesa limbajul (în majoritate textul) pentru a obține obiectivul dorit (o reprezentare vectorială a propoziției, un răspuns adecvat la o întrebare etc.). Numele lor vine din faptul că numărul de parametri pe care îl are un LLM și cantitatea textului la care are acces la antrenare sunt factorii cei mai importanți în performanța sa. LLM-urile au fost folosite înainte de acest studiu pentru a nota răspunsurile în cadrul Massive Open Online Courses (MOOC) — cursuri online disponibile publicului, accesibile unui număr mare de participanți prin internet [4]. Rezultatele au fost promițătoare, mai ales atunci când a existat un barem explicit pentru notare. Ele s-au arătat a fi mai consistente (variația notelor atribuite era mai mică) decât corectorii umani [4].

Pentru că nu există date publice cu lucrări de bac, cât și notele lor primite, ne reducem abordarea doar la bacalaureatul de informatică pentru a ușura colectarea datelor, dar și studiul în general. În plus de acest lucru, baremul acestui examen este destul de obiectiv. Studiul de față își propune să investigheze două direcții de cercetare: Este posibil de a corecta lucrările de bacalaureat în informatică folosind un LLM? și Este posibil de a da feedback unui profesor folosind un LLM în acest context? Prima oară, o să explicăm noțiunile și resursele folosite pentru studiu în preliminarii. După aceea, o să explorăm modul în care datele au fost create, dar și prelucrate, în capitolul 3. În capitolul 4 va fi explicată corectarea automată a lucrărilor produse de studenți, dar și feedbackul generat pentru profesor (dacă elevii ar face parte din aceeași clasă, ce informații i-ar putea fi

oferite profesorului pentru a îmbunătăți predarea sa?). În cele din urmă, o să expunem rezultatele studiului în concluzie.

# Capitolul 2

## Preliminarii

O să explicăm tehnologiile folosite, dar și anumite noțiuni utilizate în acest studiu.

Cel mai învechit LLM folosit este GPT 4 [5]. Modelul a fost publicat în martie 2023, și faptul că este cel mai vechi va fi important ulterior.

Al doilea LLM „învechit” este DeepSeek-V3, care a fost publicat în decembrie 2024 [2]. A introdus diferite mecanisme care i-au permis să fie mai ieftin decât competiția sa, dar tot păstrând o performanță similară.

Al treilea LLM studiat este DeepSeek-R1, care a fost publicat în ianuarie 2025 [1]. În contrast cu GPT 4, DeepSeek-V3 și GPT 4.1, acesta este un thinking model, adică LLM-ul este forțat să „gândească” înainte să dea un răspuns utilizatorului. Partea de gândire constă în încercarea a mai multor abordări, dar și un spirit mai critic, care este mai greu de convins. Tipic vorbind, LLM-ul încearcă să separe problema în mai multe etape de rezolvare în partea de gândire. Noutatea adusă de acest LLM este că e mult mai ieftin decât competiția lui, dar și mai performant (decât LLM-urile de gândire care au fost publicate înaintea lui).

Gemini 2.5 este al doilea cel mai nou LLM utilizat în acest studiu [3]. Modelul a fost publicat în martie 2025. Nu sunt foarte multe lucruri cunoscute despre el, în afară că și el este un thinking model, și că a ridicat performanța state-of-the-art la mai multe capacități, inclusiv matematică [3]. Există două variante: cea pro și cea flash, evident, varianta pro este cea mai scumpă.

Cel mai recent LLM studiat este GPT 4.1 [6]. Modelul a fost publicat în aprilie 2025 și are o performanță mult mai bună decât predecesorul său, GPT 4o.

Acum că am introdus tehnologiile folosite, trebuie și să precizăm noțiunile folosite:

- Inferența: în acest context, se referă la procesul prin care modelul generează un răspuns pornind de la o interogare dată de la utilizator. Un lucru important de notat este că s-a observat că mai multe resurse alocate inferenței (mai multe cuvinte generate, răspuns mai ”gândit” și detaliat) rezultă în performanță mai bună. De exemplu, s-a arătat că dacă instruiști un LLM să se gândească pas cu pas și nu să

răspundă pur și simplu, performanța crește [7].

- **Antrenarea:** în acest context, se referă la procesul de a lua un model neinițializat, care nu știe nici măcar să vorbească, și să îl îmbunătățești pe baza anumitor date (ar putea fi text de pe internet, cărți, și toate documentele scrise în text). Această îmbunătățire se face schimbând parametrii modelului.
- **Fine-tuning:** se referă la procesul de a lua un model deja antrenat și de a-i schimba parametrii a doua oară, pentru a-l face mai bun pentru o anumită sarcină. Lucrul acesta ar merge bine, de exemplu, dacă luăm un LLM care a fost antrenat să facă „orice” și îl specializăm să răspundă la emailuri. Ideea de fine-tuning este interesantă pentru acest studiu, pentru că dacă am face un fine-tuning pe LLM-urile studiate, ar fi foarte probabil ca performanța să crească puțin.

O premisă **foarte importantă** a acestui studiu este că dacă distanța medie între nota adevărată și nota asignată (mean absolute error, sau MAE) de LLM este sub 5 puncte din 100, atunci acuratețea este judecată ca fiind destul de bună pentru a corecta notele de bac. Am luat ca metrică de performanță MAE pentru că a fost folosită înainte pentru a măsura performanța LLM-urilor la notarea lucrărilor [4]. Este și mai intuitiv să luăm MAE în loc de MSE (mean squared error) în acest caz, pentru că și erorile mici sunt importante. De exemplu, o diferență mică între așteptările studentului și nota primită ar putea face studentul să depună o contestație. Motivul pentru care pragul pentru reușită este setat la o eroare de 5 puncte din 100 este pentru că:

1. Chiar dacă baremul este destul de obiectiv, există totuși o mică parte de interpretare subiectivă
2. Până în 2016, o notă trebuia să se schimbe cu cel puțin 0.5 puncte din 10 ca să fie admisă contestația

# Capitolul 3

## Colectarea si prelucrarea datelor

### 3.1 Colectarea datelor

Datele au fost generate de studenti in aplicatia teams a facultatii. Modul in care au lucrat a fost urmatorul: li s-a asignat un subiect de bac la informatica din 2020 pana in 2024 si au trebuit sa isi aleaga un numar aleatoriu dintre 5 si 10. Dupa aceea au fost instruiti sa faca greseli astfel incat nota lor sa corespunda cu numarul ales. Pentru a facilita corectarea lucrarilor, studentii au si trebuit sa indice greselile facute, dar si nota pe care si-au asignat-o. Am primit in total 42 de lucrari, dintre care au fost filtrate 9 lucrari pentru ca aveau date de proasta calitate. Problemele pe care le-am intalnit aici erau multiple

- Niste studenti pur si simplu nu au ales subiectul care trebuie (sesiunea gresita, profilul stiintele naturii in loc de Matematica-Informatica)
- Niste studenti au trimis in formatul gresit (fisier .docx a interferat cu sitaxa colului, fisiere excel in loc de .txt)
- Niste studenti pur si simplu nu erau programatori puternici, si nu s-au verificat destul , deci cand a fost efectiv corectata lucrarea au avut mai multe greseli decat au raportat. Acest lucru este problematic pentru ca incetineste mult corectarea lucrarilor. Totusi studentii erau din anul intai si aceste greseli au fost cel mai probabil neintentionate

Pentru a evita aceste probleme puteam face 2 lucrui:

- Sa creem un formular pentru submitere
- Sa mobilizam studenti cu mai multa experienta in programare



## 3.2 Prelucrarea datelor

Datele folosite au fost nu numai lucrarile studentilor, dar si subiectele si barelmele fiecarui an. Acestea au trebuit sa fie rescrise pentru a avea o structura mai adaptata pentru LLM-uri. In plus de acestea, au fost rescrise intr-un limbaj mai simplu si usor de inteles. Am filtrat o a doua oara lucrarile studentilor pana a avea 23 de lucrari in total. Acest lucru a fost facut din 2 motive : a limita costurile lucrarii (fiecare query costa relativ mult) si a permite testarea a mai multor LLM-uri, dar si pentru a avea date de calitate. Lucrarile au fost corectate de autor, cu atentie sporita la barem dar si la potentialele erori facute de elev (tot continutul lucrarilor a trebuit recorectat, pentru a garanta o nota cat mai apropiata de realitate)

# Capitolul 4

## Corectarea automata si feedback automat

### 4.1 Corectarea automata

Una dintre premisele lucrării este ca dacă LLM-ul are o distanță mai mică de 0.5 puncte din 10, atunci ar putea teoretic înlocui un corector uman. Ne întrebăm în primul rând dacă cea mai directă abordare (se da toată lucrarea, tot baremul, și tot subiectul și este instruit să noteze lucrarea pe baza baremului) produce rezultate bune. Pentru acest lucru o să folosim GPT 4.1 din 2 motive: Nu este un thinking model (vrem să avem abordarea cât mai simplă), și este un model recent pentru studiu. Eroarea folosită este mean absolute error (MAE), dar se ia în calcul și cel mai prost rezultat CMPR, când distanța între nota reală și cea asignată a fost cea mai mare

Modelul	MAE	CMPR
GPT 4.1	4.78	18

Tabela 4.1: Prima abordare, punctaj acordat din 100 puncte

Cum putea vedea modelul se încadrează tehnic în limita superioară impusă (5 puncte din 100). Totuși, este foarte aproape de această limită, și CMPR este foarte mare.

În plus analizând justificarea acordată a notelor puteam observa mai multe lucruri în neregulă:

- Uneori nu corectează pe baza baremului, de exemplu dacă vede un program scris ”foarte lenes”, el îi va da 0 puncte chiar dacă baremul generos de la bac i-ar acorda 5 puncte
- Uneori LLM-ul a rescris codul cu intenția de a-l puncta (lucru care se întâmplă dacă textul dat este foarte lung), dar a omis o parte a codului care a judecat-o irelevantă. Dar după ce a trecut prin cerințele baremului, a depunctat studentul pentru că

lipseste partea a codului care nu a vrut sa o copieze. Cred ca lucrul acesta are 2 cauze simultane : un LLM nu se poate "gandii" la tot textul in acelasi timp, deci nu isi da seama de eroarea comisa. In plus, planificarea in avans a fost clar proasta.

- Uneori nu si-a dat seama ca raspunsul corect se afla in barem, deci a incercat sa gaseasca singur solutia, dar nu a fost corect. Deci corectarea a fost si ea gresita

Multe dintre neintelegerile sale au fost din cauza contextului prea mare, care l-a facut sa ignore baremul sau sa corecteze gresit. Din cauza asta avem o a doua abordare: ne dam seama ca fiecare intrebare de la bac este independenta, deci se imparte fiecare lucrare in 7 "lucrari", si baremul si subiectul sunt si ele impartite cum trebuie. Dupa ce LLM-ul corecteaza tot, sunt recombinate toate cele 7 diviziuni in aceasi lucrare, si numai pe baza lucrarii intregi se masoara performanta. In plus de asta folosim mai multe modele, inclusiv si thinking models. Din curiozitate, am inclus si GPT 4, pentru a vedea cat de important este factorul ca modelele sunt noi. Folosind aceleasi metrici:

Modelul	MAE	CMPR
GPT 4.1	3.63	12
GPT 4	11.06	25.5
Gemini 2.5 flash	3.67	11
Gemini 2.5 pro	3.09	11
Deepseek v3	5.8	14
Deepseek r1	4.78	14

Tabela 4.2: A doua abordare, punctaj acordat din 100 puncte

Putem remarca imediat ca GPT 4 este un model foarte prost in comparare cu ceilalti, deci confirma ideea ca LLM-urile s-au ameliorat foarte mult in anii recenti si nu mai este cazul de a folosi unul care nu este recent. Un LLM de actualitate este mult mai performant si de multe ori mult mai ieftin de folosit. Un alt lucru care iese la iveala este ca resursele alocate inferentei este crucial: exista o diferenta semnificativa intre Gemini 2.5 flash si Gemini 2.5 pro de 0.5 puncte. In plus de asta exista o diferenta semnificativa intre Deepseek v3 si Deepseek r1 de 1 punct, iarasi sustinand ideea ca mai multe resurse alocate la inferente rezulta in predictii mai bune. Deci se pot distinge 2 factori cruciali in a determina performanta predictiilor:

- Resursele alocate inferentei
- Cat de recent este LLM-ul folosit (cu atat este mai recent, cu atat mai bine)

Putem remarca si faptul ca GPT 4.1 si Gemini 2.5 pro au avut erori foarte mici: 3.6 si 3. Acest lucru este destul de impresionant pentru ca exista o parte subiectiva in a interpreta baremul, si rezultatul cel mai bun (3) este cu mult sub limita superioara impusa. CMPR este totusi destul de mare pentru ambele modele. Doua greseli pe care am putut sa le identific este ca :

- O data un student a facut o greseala in logica codului, si Gemini 2.5 pro a spus ca algoritmul este gresit, cand era clar doar o mica greseala de logica
- O data un student nu a respectat cerinta, dar GPT 4.1 i-a acordat toate punctele, pentru ca nu si-a dat seama ca structurile **repetă ... până când ...** si **până când ... repetă ...** sunt foarte diferite in contextul cerintei : trebuia ca studentul sa inlocuiasca a doua structura in pseudocod cu prima, pentru a arata ca a inteles cum se pot echivala algoritmic ambele. Studentul nu a folosit structura care trebuie, dar a primit totusi toate punctele.

Ca solutie la aceste probleme, ar putea fi folosit un barem mai explicit, mai redundant, sau poate chiar si fine-tuning.

## 4.2 Feedback automat

Vrem sa ne interesam daca LLM-urile pot sa analizeze cum trebuie greselile studentilor, si sa propuna feedback relevant profesorului (ce parte din materie nu a fost bine inteleasa de catre studenti, si trebuie predata mai bine). Pentru asta, luam cele top 2 modele in performanta, si le testam. Datele primite ca input pentru LLM sunt corectarile lucrarilor obtinute, fara subiect sau lucrarea propriu zisa (a trebuit sa justifice nota LLM-ul, deci exista destul context doar in corectare). Aceasta parte este mult mai subiectiva decat cea precedenta, pentru ca este greu sa cuantifici calitatea unui feedback. Probele comune ale lucrarilor, identificate de mine sunt :

- sintaxa lui C++
- Algoritmica nu este inteleasa bine
- Intrebarile nu sunt citite bine, si de multe ori studentii s-au aruncat in a scrie o solutie fara sa se gandeasca
- I/O este implementat prost
- Prelucrarea stringurilor (`char[ ]`) nu este bine inteleasa
- Studentii pierd foarte multe puncte la intrebarile cu raspunsuri multiple

Problemele comune, identificate de GPT 4.1 sunt:

1. Algoritmica nu este inteleasa bine
2. ~~Studentii nu lucreaza bine cu matrici~~ (acest feedback este categoric fals)
3. Prelucrarea stringurilor (`char[ ]`) nu este bine inteleasa

4. I/O este implementat prost
5. Parametrii referinta (`int f(&n)`) nu sunt bine folositi
6. Initializarile sunt prost facute

Deci LLM-ul a identificat niste probleme foarte importante, unele care nu au fost identificate de mine. Totusi (2) este un feedback gresit. Cea ce lipseste in acest feedback este ca sintaxa nu a fost bine inteleasa, si ca studentii pierd prea multe puncte la intrebarile cu raspuns multiplu.

Cand vine vorba de gemini 2.5 pro, a identificat toate problemele pe care le-am vazut, mai putin faptul ca studentii au pierdut prea multe puncte la intrebarile cu raspuns multiplu. Nu am primit feedback gresit din partea lui gemini 2.5 pro.

Pentru a completa partea de feedback, le-am cerut LLM-urilor sa genereze 30 probleme, pentru ca studentii sa se poata antrena cu ele. Din cele 30 generate de GPT 4.1, 10 au fost irelevante, si dintre cele 30 generate de gemini, 7 au fost irelevante.

Putem trage concluzia ca :

- Se pare ca o performanta mai buna in MAE si CMPR la corectare se transpune intr-o performanta mai buna cand vine vorba de feedback
- Gemini 2.5 pro este cel mai bun model, si la corectare automata, dar si la feedback

Totusi, ambele LLM-uri au dat feedback relevant.

Pe baza rezultatelor din Capitolul 4, se poate trage concluzia că există o probabilitate ridicată ca lucrările de bacalaureat la informatică să poată fi corectate automat cu un LLM. De asemenea, este posibilă implementarea unui sistem automatizat de feedback pentru profesori.

Alte rezultate din acest studiu arată că factorii cei mai importanți pentru acuratețea corectării sunt:

- Cât de recent este LLM-ul
- Resursele alocate pentru inferență (putere de calcul  $\times$  timp, cât de detaliată și „gândită” este corectarea LLM-ului)

În plus, implementarea unei notări automate trebuie să împartă lucrarea în părți cât mai mici posibil: dacă textul dat LLM-ului este prea mare, atunci LLM-ul poate omite baremul sau poate comite alte greșeli.

O ipoteză care reiese din studiu este că, dacă un LLM are un MAE și un CMPR mai bun la notarea lucrărilor, atunci este mai bun pentru a transmite feedback. Pentru a dezvolta și mai departe rezultatele studiului și pentru a fi cât se poate de siguri de rezultatele sale, am putea relua evaluarea performanței ca în Capitolul 4, dar cu un set de date mai mare ( $>1000$  lucrări).

Studiul de față nu propune nicio schimbare socială, dar dacă LLM-urile ar prelua corectarea lucrărilor de bac, atunci am recomanda păstrarea corectării umane cel puțin pentru lucrările cu notă contestată.

# Bibliografie

- [1] DeepSeek-AI et al., „DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning”, în *arXiv preprint arXiv:2501.12948* (Ian. 2025), URL: <https://arxiv.org/abs/2501.12948>.
- [2] DeepSeek-AI et al., „DeepSeek-V3 Technical Report”, în *arXiv preprint arXiv:2412.19437* (Dec. 2024), URL: <https://arxiv.org/abs/2412.19437>.
- [3] Google DeepMind, *Gemini 2.5: Our most intelligent AI model*, Accesat : Iunie 2025, Mar. 2025, URL: <https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025>.
- [4] Shahriar Golchin, Nikhil Garuda, Christopher Impey și Matthew Wenger, „Grading Massive Open Online Courses Using Large Language Models”, în *Proceedings of the 31st International Conference on Computational Linguistics*, ed. de Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio și Steven Schockaert, Abu Dhabi, UAE: Association for Computational Linguistics, Ian. 2025, pp. 3899–3912, URL: <https://aclanthology.org/2025.coling-main.263/>.
- [5] OpenAI, *GPT-4 Technical Report*, arXiv:2303.08774, 2023, URL: <https://arxiv.org/abs/2303.08774>.
- [6] OpenAI, *Introducing GPT-4.1 in the API*, Accesat: Iunie 2025, Apr. 2025, URL: <https://openai.com/index/gpt-4-1/>.
- [7] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le și Denny Zhou, „Chain-of-thought prompting elicits reasoning in large language models”, în *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, New Orleans, LA, USA: Curran Associates Inc., 2022, ISBN: 9781713871088.