Kauno technologijos universitetas

Informatikos fakultetas

Pokalbio robotų tyrimas dialogo modelio sudarymui

Baigiamasis magistro studijų projektas

Justas Gudaitis

Projekto autorius

Prof. Dr. Rytis Maskeliūnas

Vadovas

Kaunas, 2025

Kauno technologijos universitetas

Informatikos fakultetas

Pokalbio robotų tyrimas dialogo modelio sudarymui

Baigiamasis magistro studijų projektas

Dirbtinio intelekto informatika (6211BX007)

Justas Gudaitis (parašas)
Projekto autorius (data)

Prof. Dr. Rytis Maskeliūnas (parašas)
Vadovas (data)

Prof. Dr. Robertas Damaševičius (parašas)
Recenzentas (data)

Kaunas, 2025

Kauno technologijos universitetas

Informatikos fakultetas

Justas Gudaitis

Pokalbio robotų tyrimas dialogo modelio sudarymui

Akademinio sąžiningumo deklaracija

Patvirtinu, kad:

1. baigiamąjį projektą parengiau savarankiškai ir sąžiningai, nepažeisdama(s) kitų asmenų autoriaus ar kitų teisių, laikydamasi(s) Lietuvos Respublikos autorių teisių ir gretutinių teisių įstatymo nuostatų, Kauno technologijos universiteto (toliau – Universitetas) intelektinės nuosavybės valdymo ir perdavimo nuostatų bei Universiteto akademinės etikos kodekse nustatytų etikos reikalavimų;

2. baigiamajame projekte visi pateikti duomenys ir tyrimų rezultatai yra teisingi ir gauti teisėtai, nei viena šio projekto dalis nėra plagijuota nuo jokių spausdintinių ar elektroninių šaltinių, visos baigiamojo projekto tekste pateiktos citatos ir nuorodos yra nurodytos literatūros sąraše;

3. įstatymų nenumatytų piniginių sumų už baigiamąjį projektą ar jo dalis niekam nesu mokėjęs (-usi);

4. suprantu, kad išaiškėjus nesąžiningumo ar kitų asmenų teisių pažeidimo faktui, man bus taikomos akademinės nuobaudos pagal Universitete galiojančią tvarką ir būsiu pašalinta(s) iš Universiteto, o baigiamasis projektas gali būti pateiktas Akademinės etikos ir procedūrų kontrolieriaus tarnybai nagrinėjant galimą akademinės etikos pažeidimą.

Justas Gudaitis

Patvirtinta elektroniniu būdu

Gudaitis, Justas. Pokalbio robotų tyrimas dialogo modelio. Magistro studijų baigiamasis projektas / vadovas prof. dr. Rytis Maskeliūnas; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatikos mokslai, Informatika (B01).

Reikšminiai žodžiai: didieji kalbos modeliai, sintetiniai duomenys, dialogų sistema, modelių lyginimas, natūralios kalbos apdorojimas.

Kaunas, 2025. 61 p.

Santrauka

Sparčiai tobulėjant dirbtiniam intelektui natūralios kalbos apdorojime, pokalbių robotai tampa sudėtingesni, tačiau jų kūrimas mažoms kalboms, tokioms kaip lietuvių, yra iššūkis dėl ribotų duomenų ir mokymo kaštų. Šiame tyrime analizuojami ir tobulinami lietuviški dialogo modeliai, pasitelkiant didžiuosius kalbos modelius (LLM) ir parametrų efektyvumo derinimo (PEFT) metodus. Darbe siūloma strategija apima sintetinių duomenų generavimą, jų derinimą su lietuviškais rinkiniais, modelių adaptavimą LoRA metodu, išsamų duomenų filtravimą ir tokenizacijos vertinimą. *LLaMA 3.2 ir Gemma 3* modeliai (1B, 4B parametrų) treniruoti ir vertinti standartizuotais lietuviškais testais (*MMLU-LT, ARC-LT, TruthfulQA-LT*). Išanalizuota, kad PEFT metodai (ypač LoRA) efektyviai adaptuoja LLM lietuvių kalbai su ženkliai mažesniais resursais nei pilnas pertreniravimas. Sintetinių duomenų generavimas ir derinimas su filtruotais viešaisiais rinkiniais reikšmingai pagerino modelių konteksto supratimą ir atsakymų kokybę. Lyginant modelius, *Gemma 3 4B* (LoRA) pasiekė aukštus rezultatus *TruthfulQA MC2* (54.0%) ir *ARC-LT* (50.8%) testuose, o *LLaMA 3.2 3B* pirmavo *MMLU* (38.8%) ir *BLEU* (0.323) rodikliuose. Šiame darbe sukurti PEFT modeliai pasiekė konkurencingus ar net pranoko rezultatus lyginant su didesniais, pilnai pertreniruotais *LLaMA* modeliais. Tyrimo išvados rodo, kad strateginis PEFT metodų ir kokybiškų, įvairiapusių duomenų rinkinių taikymas leidžia sėkmingai kurti bei tobulinti lietuviškus dialogo modelius su ribotais ištekliais, prisidedant prie lietuvių kalbos technologijų plėtros.

Gudaitis Justas. Research of Chatbots for Dialogue Model Creation. Master's Final Degree Project / supervisor prof. dr. Rytis Maskeliūnas; Faculty of Informatics, Kaunas University of Technology.

Study field and area (study field group): Computer science, Informatics (B01).

Keywords: large language models, synthetic data, dialogue system, model comparison, natural language processing.

Kaunas, 2025, 61 p.

Summary

With the rapid advancement of artificial intelligence in natural language processing, chatbots are becoming more sophisticated. However, developing them for low-resource languages like Lithuanian presents challenges due to limited data and high training costs. This study analyzes and improves Lithuanian dialogue models by leveraging large language models (LLMs) and parameter-efficient fine-tuning (PEFT) methods. The proposed strategy includes synthetic data generation, integration with Lithuanian datasets, model adaptation using the LoRA method, comprehensive data filtering, and tokenization evaluation. The LLaMA 3.2 and Gemma 3 models (1B and 4B parameters) were trained and evaluated using standardized Lithuanian benchmarks (MMLU-LT, ARC-LT, TruthfulQA-LT). The analysis shows that PEFT methods, especially LoRA, effectively adapt LLMs to the Lithuanian language using significantly fewer resources than full retraining. Generating synthetic data and combining it with filtered public datasets substantially improved the models' contextual understanding and response quality. Among the models, Gemma 3 4B (LoRA) achieved high results on the TruthfulQA MC2 (54.0%) and ARC-LT (50.8%) tests, while LLaMA 3.2 3B led in MMLU (38.8%) and BLEU (0.323) scores. The PEFT models developed in this study achieved competitive or even superior results compared to larger, fully retrained LLaMA models. The findings indicate that the strategic use of PEFT methods and high-quality, diverse datasets enables the successful development and enhancement of Lithuanian dialogue models with limited resources, contributing to the advancement of Lithuanian language technologies.

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Santrumpų ir terminų sąrašas

Santrumpos:

BERT – (angl. *Bidirectional Encoder Representations from Transformers*) Dvipusio kodavimo reprezentacijos iš transformatorių

CARP – (angl. *Clue And Reasoning Prompting*) Užuominų ir samprotavimo pagrindu grįstas užklausų formulavimas

CoT – (angl. *Chain-of-Thought*) Minties grandinė

FIM – (angl. *Fill-In-the-Middle)* Vidurio užpildymas

GPU - (angl. *Graphics Processing Unit*) Grafinis procesorius

GPT – (angl. *Generative Pre-trained Transformer*) Generatyvinis iš anksto apmokytas transformatorius

GQA – (angl. *Grouped Query Attention*) Grupinių užklausų dėmesys

LM – (angl. *Language Model*) Kalbos modelis

LLM – (angl. *Large Language Model)* Didelis kalbos modelis

LoRA – (angl. *Low-Rank Adaptation*) Mažo rango adaptacija

LOMO – (angl. Low-Memory Optimization) Mažos atminties optimizacija

NLP – (angl. *Natural Language Processing*) Natūralios kalbos apdorojimas

PEFT – (angl. *Parameter-Efficient Fine-Tuning*) Parametrus tausojantis tikslusis derinimas

PLM – (angl. *Pre-trained Language Model*) Išankstinai apmokytas kalbos modelis

QLoRA – (angl. *Quantized Low-Rank Adaptation*) Kvantuota mažo rango adaptacija

RLHF – (angl. *Reinforcement learning from human feedback*) Sustiprinamasis mokymasis iš žmonių grįžtamojo ryšio

SFT – (angl. *Supervised fine-tuning*) Prižiūrimas tikslusis derinimas

SP – (angl. *Self-Polish*) Savęs tobulinimas

TPU – (angl. *Tensor Processing Unit*) Tensoriaus procesorius

VRAM - (angl. *Video Random Access Memory*) Vaizdo operatyvioji atmintis

pp – (angl. *percentage point*) Procentinis taškas

Įvadas

Dirbtinio intelekto pažanga natūralios kalbos apdorojimo srityje iš esmės pakeitė žmonių bendravimą su mašinomis. Pokalbių robotai, kadaise buvę paprastos taisyklėmis pagrįstos sistemos, dabar virsta sudėtingais dialogo modeliais, gebančiais suprasti, interpretuoti ir reaguoti į žmogaus kalbą panašiai kaip žmogus. Šiame tyrime daugiausia dėmesio skiriama pokalbių robotų dialogo modelių kūrimui ir tobulinimui, pasitelkiant naujausius mašininio mokymosi metodus.

Dialogo modelio kūrimas yra labai svarbus gerinant žmogaus ir kompiuterio sąveiką, užtikrinant natūralesnį ir veiksmingesnį bendravimą. Tai turi didelę reikšmę ne tik klientų aptarnavimui ir pramogoms, bet ir švietimo tikslams bei prieinamumui neįgaliesiems. Tyrimo tikslas - ištirti, kaip pokalbių robotus galima išmokyti suprasti kontekstą, valdyti pokalbio eigą ir generuoti atsakymus, kurie būtų ne tik tinkami, bet ir pritaikyti konkrečiai kalbai.

Svarbiausias šių tyrimų aspektas - natūralios kalbos supratimo (angl. *Natural Language Understanding,* NLU) ir natūralios kalbos generavimo (angl. *Natural Language Generation,* NLG) metodų naudojimas. Šios technologijos leidžia pokalbių robotams tiksliau analizuoti naudotojo įvestį ir generuoti nuoseklesnius ir kontekstui tinkamus atsakymus. Be to, tyrime nagrinėjamas nurodytos kalbos integravimas į pokalbių robotus, leidžiantis jiems atpažinti ir reaguoti į nurodytą kalbą, taip dar labiau pagerinant sąveikos kokybę.

Tikslas ir uždaviniai

Pagrindinis šio tyrimo tikslas – sukurti išplėstinį pokalbių roboto modelį, skirtą efektyviam dialogo supratimui ir generavimui nurodytuose kontekstuose.

Uždaviniai:

1. Įvairių dialogo modelių analizė ir palyginimas, siekiant nustatyti jų veiksmingumą nurodytuose kontekstuose.
2. Pažangių natūralios kalbos supratimo ir generavimo metodų įgyvendinimas siekiant pagerinti pokalbių roboto supratimo ir atsakymų generavimo galimybes.
3. Papildomų duomenų integravimo į pokalbių robotus tyrimas, siekiant geresnių rezultatų atliekant konkrečias kalbos užduotis.
4. Sukurto pokalbių roboto modelio rezultatų konkrečių užduočių scenarijuose vertinimas ir palyginimas su kitais.

1. Šiuolaikinė kalbos modelių technologija

Dirbtinio intelekto pažanga natūralios kalbos apdorojimo *(*angl. *Natural Language Processing*, NLP*)* srityje iš esmės pakeitė žmonių bendravimą su mašinomis. Viena iš svarbiausių šios srities naujovių yra 2017 m. pasirodęs straipsnis *Attention is All You Need*[1], kuris pateikė transformerio architektūrą – naują metodą, leidžiantį didelį kalbos modelį (angl. *Large Language Model*, LLM) efektyviau tvarkyti ir generuoti tekstą. Ši architektūra leido modeliams giliau suvokti kalbą. Jie vienu metu mokėsi bei įsimindavo ilgesnius tekstus, remdamiesi kontekstine informacija[1].

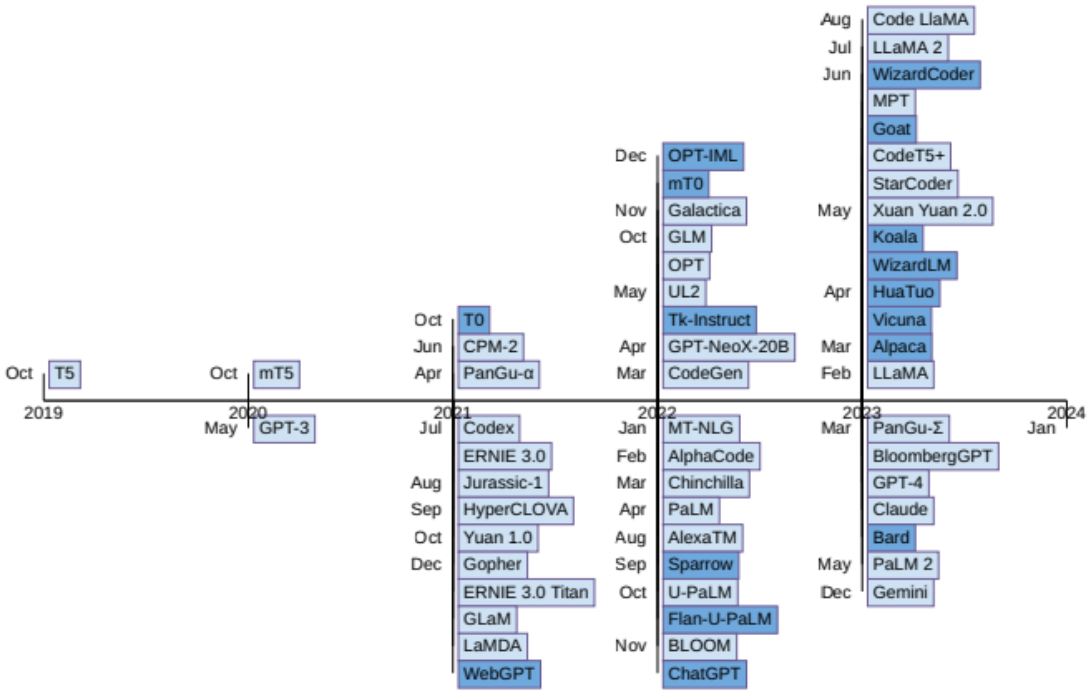
Šiandienos LLM-ai, tokie kaip GPT *(*angl. *Generative Pre-trained Transformer)* ir BERT *(*angl. *Bidirectional Encoder Representations from Transformers)*, pasižymi ne tik geresniu supratimu apie kalbos struktūrą, bet ir gebėjimu prisitaikyti prie įvairių kalbų ir dialektų. Tai svarbu, nes skirtingos kalbos reikalauja skirtingų apdorojimo metodų, ypač tokiose srityse kaip semantinė analizė ar kalbos generavimas[2].

Naujausios LLM technologijos yra pritaikomos ne tik anglų kalbai, bet ir kitoms kalboms. Tai leidžia sukurti daugiakalbius modelius, gebančius dirbti su daugybe skirtingų kalbų[3]. Tokių modelių kūrimas yra ypač svarbus globalizacijos amžiuje, kai bendravimas vyksta daugiau nei viena kalba.

Pokalbių robotai, naudodami šias pažangias LLM technologijas, gali efektyviai bendrauti įvairiomis kalbomis, suprantant ne tik tekstą, bet ir kontekstą, kuriame jis pateikiamas. Tai leidžia jiems tapti dar labiau interaktyviais ir pritaikomais vartotojų poreikiams, neatsižvelgiant į jų kalbinį foną.

1.1. Didžiųjų kalbos modelių raida

Istorinė pažanga NLP srityje vyko nuo statistikos iki neuroninio kalbos modeliavimo, o vėliau nuo PLM (angl. *Pre-trained Language Model*) iki LLM. Tradiciniu LM mokoma konkrečių užduočių modelių prižiūrimoje aplinkoje, o PLM mokoma savaime prižiūrimoje aplinkoje iš didelio teksto korpuso, siekiant išmokti bendrųjų paaiškinimų[4]. Tiksliai sureguliuotas PLM pranoksta tradicinį LM sprendžiant tolesnius uždavinius. Našumas gerėja, kai PLM plečiama naudojant daugiau parametrų ir mokymo duomenų. perėjimas į LLM tampa baigtu[5].



2.1 pav. Didelių kalbos modelių leidinių chronologinis vaizdas

Literatūroje pasiūlyta daug LLM[6]. 1 pav. parodytas didėjantis išleistų LLM skaičius ir žymūs siūlomi LLM pavadinimai. Modeliai per dešimtmetį evoliucionavo nuo statistinių metodų iki šiuolaikinių 100 mlrd. parametrų transformerių. Keli iš tokių modelių yra parodyti:

2.1 lentelė. Laiko juosta su pagrindinių LLM leidimų chronologija

Laikotarpis	Pagrindiniai įvykiai
2013 – 2016	*Word2Vec, GloVe* – pasirodo įterptys (angl. *embeddings*)
2017	Straipsnis *Attention Is All You Need* – pristatomas transformeriai
2018–2019	*BERT, GPT-2, XLNet* – PLM paradigma
2020	*GPT-3* (175 B) – *few shot* funkcionalumas
2022	*BLOOM-176B* – pirmas didelis visiškai atviras daugiakalbis modelis
2023	*LLaMA 2* (7 B–70 B) – atviri svoriai, PEFT bumas
2023	*Mistral* 7 B – našumas > *LLaMA 2* 13 B
2024	*LLaMA 3* šeima (8 B-405 B)
2024	Gemma 2 (1 B–27 B) – 256 k BPE žodynas
2025	Gemma 3 (1 B–27 B), 128 k kontekstas, 140 + kalbų
2025	EuroLLM-9B – orientuotas į 24 ES kalbas

Laiko juosta atspindi spartų LLM evoliucionavimą – nuo pradinių *embedding* metodų iki šiuolaikinių multimodalinių modelių su ilgo konteksto palaikymu ir kalbiniu universalumu. Toks chronologinis pagrindas leidžia toliau išsamiau analizuoti atskirus modelius, jų architektūrinius ypatumus ir pritaikomumą konkrečioms užduotims, ypač kalboms su ribotu išteklių kiekiu.

NLP modelių kūrimas mažiau paplitusioms kalboms yra iššūkis dėl jų mažesnio populiarumo ir riboto turimo turinio, palyginti su plačiai naudojamomis kalbomis (anglų arba kinų). „Iš maždaug 6 000 šiandien naudojamų kalbų vien dešimt populiariausių jų sudaro didžiulį 76,3 proc. viso interneto turinio kiekį“[7]. Tokia padėtis ne tik skatina kurti NLP modelius šioms mažiau naudojamoms kalboms, bet taip pat reikalauja strategijų, kaip perduoti žinias iš turtingo turinio kalbos modelio ir spręsti šią spragą.

Ištyrus, kaip modeliai veikia atliekant konkrečias užduotis, galima nustatyti jų stipriąsias ir silpnąsias puses, kad būtų galima priimti labiau pagrįstus sprendimus dėl suderinimo su specifine užduotimi.

1.2. Pagrindinių LLM apžvalga

1.2.1. GPT-3

Kadangi tyrimo metu GPT-4 klasės modelis nėra laisvai prieinamas tobulinimui nuspręsta aptarti senesnę LLM versiją GPT-3.

GPT serijos modeliai, tokie kaip GPT-3, CodeX, InstructGPT, ChatGPT sulaukė didelio dėmesio dėl savo išskirtinių natūralios kalbos apdorojimo galimybių. Tačiau tiksliai sureguliuoti *(*angl. *Fine-tune)* GPT-3 konkrečioms užduotims skirtose imtyse gali būti sudėtinga dėl didelio modelio dydžio[8].

Naudojant GPT-3 asmeniniais tikslais, būtina atidžiai stebėti generuojamą atsakymą, kad būtų sušvelnintos galimos neigiamos pasekmės. Tai reiškia, kad reikia atidžiai prižiūrėti GPT-3 sukurtus teiginius, siekiant užtikrinti, kad jis atitiktų numatytą paskirtį ir nesukurtų nepageidaujamo turinio.

GPT-3 empiriniai rezultatai labai priklauso nuo konteksto pavyzdžių pasirinkimo. Pasirinkus veiksmingus konteksto pavyzdžius, kurie yra semantiškai panašūs į testinės užklausos pavyzdį, galima geriau išnaudoti GPT-3 teksto generavimo galimybes[9].

Norint tiksliai pritaikyti GPT-3 teksto generavimui tam tikra kalba ar tarme, galima naudoti GPT-3 kalbos modelio mokymosi metodą, kuris apima kelis etapus. GPT-3 veikia ir su nedideliu kiekiu tikslių duomenų ir iš prigimties pažįsta praktiškai visas sritis[10]. Pavyzdžiui, atliktas tyrimas, kuriame daugiausia dėmesio skirta GPT-3 tiksliniam derinimui generuojant sintetinius naujienų straipsnius danų kalba. Tyrimo metu nustatyta, kad modelio veikimą galima įvertinti pagal žmogaus ir mašinos aptikimo rodmenis ir kad tiksliai sureguliuotas BERT klasifikatorius pasiekė aukštą užduoties tikslumą[11]. Kitame tyrime sukurta dirbtiniu intelektu pagrįsta teksto generatoriaus programa, naudojanti GPT-3 kalbos modelį, kad būtų galima automatiškai generuoti tekstą ir padėti įveikti teksto turinio kūrimo kliūtis. Programa gali prisijungti prie GPT-3 variklio, kurio užklausos metodas buvo pakeistas, ir generuoti naudotojo poreikiams pritaikytą tekstą pagal žiniatinklio sąsajos sistemoje įvestus raktažodžius[12].

Gali būti sudėtinga įvertinti GPT-3 veikimą ne anglų kalba, nes kitoms kalboms nėra nustatytų vertinimo rodiklių ar lyginamųjų standartų[13].

- Architektūra – 96 sluoksnių, 175 B parametrų autoregresinis transformeris[14].
- Stipriosios pusės – *few-shot* gebėjimai, platus enciklopedinis žinių fonas.
- Silpnosios pusės – didelė skaičiavimo kaina, uždaras kodas, ribotas pritaikymas mažoms kalboms.
- Derinimo įžvalga – kontekstinių pavyzdžių parinkimas smarkiai veikia našumą

1.2.2. LLaMA 2

"Llama 2" yra didelis kalbos modelis su 7B, 13B ir 70B parametrų variantais. Jis naudoja optimizuotą transformatorių architektūrą ir *supervsed fine-tuning (*SFT*)* bei stiprinimo mokymąsi su žmogaus atsiliepimais (angl. *Reinforcement learning from human feedback,* RLHF), siekiant suderinti su žmogaus pageidavimais dėl naudingumo ir saugumo.

„Llama 2“ yra reikšmingas „Llama 1“ patobulinimas[15]. Modeliai iš anksto apmokyti naudojant optimizuotą automatinio regresinio transformatoriaus architektūrą. Apmokymui naudotas naujas viešai prieinamų internetinių duomenų mišinys, kuris apdorotas atliekant išmanesnį duomenų valymą ir atnaujinimą. Svarbu paminėti, kad apmokymui nebuvo naudoti jokie „Meta“ vartotojų duomenys. Modeliai apmokyti ant 2 trilijonų žetonų, o tai yra 40 % daugiau nei „Llama 1“. Be to, konteksto ilgis buvo padvigubintas iki 4096 žetonų (4k). Didesniems modeliams, konkrečiai 34B ir 70B, siekiant pagerinti išvedimo mastelį, pritaikytas grupinis užklausų dėmesys GQA (angl. *Grouped Query Attention*)[16].

Vertinant „Llama 2“, ypač pokalbiams optimizuotą „Llama 2-Chat“ versiją, pažymima, kad ji gerokai pranoksta esamus atvirojo kodo pokalbių modelius. Palyginus su uždarojo kodo modeliais, „Llama 2-Chat“ modeliai atrodo esantys panašūs į kai kuriuos iš jų. Žmonių vertinimai rodo, kad didžiausias „Llama 2-Chat“ modelis (70B) yra konkurencingas su „ChatGPT“. Konkrečiai, šis modelis pasiekė 36% laimėjimo rodiklį ir 31,5% lygiųjų rodiklį, palyginti su „ChatGPT“, remiantis užklausų rinkiniu. „Llama 2-Chat 70B“ modelis taip pat gerokai pranoksta „PaLM-bison“ pokalbių modelį. Modelio tobulinimas apėmė SFT su aukštos kokybės duomenimis ir RLHF[15].

- 7 B/13 B/70 B variantai; treniruota su 2 T žetonų korpusu.
- Įdiegti GQA + 4 k žetonų kontekstas; PEFT (LoRA, QLoRA) palaikymas.
- Bazinėje versijoje < 1 % LT turinio[15]

Dėl turinio kiekio bei lengvo treniravimo modelių pritaikymo yra privalus papildomas derinimas.

1.2.3. BLOOM

BLOOM turi 176 milijardus parametrų, t. y. vienu milijardu daugiau nei GPT-3. Jis turi 70 sluoksnių, 112 dėmesio galvučių *(*angl. *attention heads)* viename sluoksnyje, paslėptą 14336 dimensiją ir 2048 ženklų sekos ilgį. Jis naudoja ALiBi pozicijos įterpinius ir GeLU aktyvavimo funkciją.

Pagal našumą BLOOM pranoksta kitus panašaus dydžio modelius SuperGLUE, natūralios kalbos supratimo teste, ypač daugiakalbėse užduotyse. BLOOM taip pat pasiekia konkurencingų rezultatų mašininio vertimo srityje, ypač kalbų su mažais ištekliais, palyginti su prižiūrimais modeliais, tokiais kaip M2M ir XLM-R. BLOOM pasiekia geresnius daugiakalbio apibendrinimo rezultatus nei OPT, kitas didelės apimties kalbų modelis, ir rodo, kad našumas didėja su parametrų skaičiumi. Tačiau BLOOM yra silpnesnis už Codex modelius, kurie yra smulkiai sureguliuoti kodui, atliekant kodo generavimo etaloną HumanEval[17].

- 176 B (ir 7 B, 3 B) daugiakalbis modelis, treniruotas ROOTS 46 kalbų korpusu ir 13 programavimo kalbų korpusu.
- 0-shot vertime pralenkia OPT, bet nusileidžia kodui skirtiems modeliams HumanEval teste.

1.2.4. mGPT

Modelis mGPT 1.3 B ir 13 B parametrų modelis yra daugiakalbė GPT-3 versija, apimanti 61 kalbą iš 25 kalbų šeimų, įskaitant mažiau atstovaujamas ir mažai išteklių turinčias kalbas. Straipsnyje teigiama, kad mGPT turi šiuos pranašumus lyginant su GPT-3[18]:

- mGPT pasiekia panašią arba geresnę veiklą tarp-kalbinio natūralios kalbos supratimo užduotyse ir vertinimuose 33 kalbomis.
- mGPT pasižymi didesne lingvistine įvairove ir gali generuoti nuoseklias ir kontekstui tinkamas kalbas mažiau paplitusias kalbas.
- mGPT turi mažesnes skaičiavimo, energetikos ir anglies dvideginio išmetimo sąnaudas nei GPT-3, nes jis naudoja mažiau parametrų ir mokymosi žingsnių[18].

1.2.5. Mistral 7 B (ir Mixtral 8×7 B)

Mistral ir Mixtral yra vienas iš geriausių jaunesnės kartos modelių pavyzdžių, kurie parodo, kad tvarkingas duomenų rinkinys gali įveikti modelius, kurie treniruoti su daug neprižiūrėtų duomenų. Mixtral taip pat puikiai išnaudoja savo galimybes kaip MoE (angl. *Mixture of experts*) modelis. 8x7B Mixtral modelis yra aštuonių modelių kombinacija, kurių kiekvienas treniruotas su skirtingais duomenimis ir kalbomis, tačiau sujungti į vieną bendrą modelį ir turi bendrą kartu treniruotą komunikacijos sluoksnį.

- Naudoja SWA + GQA; 7 B modelis pralenkia LLaMA 2 13 B įvairiose užduotyse.
- Bazinis korpusas daugiausia EN, bet atviroji Apache 2.0 licencija palanki LT adapteriams[19]

MoE yra reikalaujantis daug resursų, tačiau daug žadantis Mistral serijos modelis.

1.2.6. LLaMA3

„Meta“ sukurti „LLaMA 3“ modeliai žymi reikšmingą pažangą daugiaakalbių LLM srityje. Nuo 3.0 versijos šie modeliai buvo nuosekliai tobulinami, siekiant pagerinti jų gebėjimą apdoroti įvairias kalbas, įskaitant mažiau paplitusias, tokias kaip lietuvių. Tai pasiekta išplečiant mokymo duomenų apimtį ir įvairovę, taip pat taikant pažangius mokymo metodus, tokius kaip žinių distiliacija ir RLHF. Papildomai 3.2 versijos modeliai buvo optimizuoti daugiaakalbiams dialogams ir pasižymi geresniu našumu nei daugelis kitų atvirojo kodo ir uždarų pokalbių modelių

1.3. lentelė. LLaMA 3 iki 3.3 versijos modelio evolicija

Versija	Esminiai patobulinimai
3.0	Daugiakalbė *herd* architektūra, 8 B, 70 B, 405 B[20]
3.1	128 k žetonų kontekstas, 8 oficialios kalbos[20]
3.2	Tekstas + vaizdas, išplėstas kalbų sąrašas > 20[21]
3.3	70 B modelis < 30 GB VRAM

1.2.7. Gemma 2 ir Gemma3

„Google“ sukurti Gemma modeliai pasižymi labai geru kalbų supratimu, ypač vertimo užduotyse. Viena iš priežasčių – labai išplėstas žodynas. Pavyzdžiui, „Gemma 2“ modelis naudoja net 256 tūkst. skirtingų vienetų (žetonų), todėl sudėtingesni žodžiai, tokie kaip lietuviški, yra išskaidomi į mažiau dalių. Tai reiškia, kad modelis juos gali geriau suprasti ir išversti. Be to, „Gemma 2“ modeliai buvo mokomi naudojant pažangius metodus, tokius kaip žinių distiliacija – kai mažesnis modelis mokosi iš daug didesnio modelio, taip išlaikant kokybę, bet sumažinant reikiamus resursus[22].

Dar pažangesnis yra „Gemma 3“ modelis. Jis ne tik palaiko virš 140 kalbų, bet ir geba dirbti su tekstu ir vaizdais vienu metu (t. y. yra multimodalus). Jo konteksto langas siekia 128 tūkstančius žetonų, todėl modelis gali „perskaityti“ labai ilgas užduotis ar tekstus. Mokymo procese naudoti papildomi metodai, tokie kaip žmogaus grįžtamojo ryšio stiprinimas (RLHF), padeda modeliams geriau suprasti sudėtingesnius klausimus ir pateikti aiškesnius atsakymus[23]. Tokios savybės ypač naudingos, kai modelis taikomas mažesnėms kalboms, kaip lietuvių, nes leidžia išgauti geresnę kokybę net ir turint mažiau duomenų.

- Gemma 2 – 2-27 B, 256 k žodynas, spartesnis lietuvių kalbos suvokimas.
- Gemma 3 – multimodalumas, 128 k kontekstas, 140 + kalbų, pranašesnis LT vertime nei Gemma 2.

1.2.8. EuroLM

EuroLM yra dar vienas iš itin retų, bet vertų dėmesio modelių kurie turi specialų duomenų rinkinį modeliui treniruoti lietuvių kalbos. Tai yra atviro kodo modelis.

- 1.7 B ir 9 B modeliai, specialiai treniruoti 24 ES kalboms; 128 k žodynas optimizuotas Europos diakritikams.
- EuroLLM-9 B demonstruoja geriausią reitingą ES kalbų testuose[24]

1.2.9. Aya

„Aya“ atviras daugiakalbis instrukcijoms pritaikytas modelis, pristatytas 2023 m. pabaigoje. Jis yra papildytas mT5-XXL (13B) versija, papildomai išmokintas 101 kalba (tarp jų lietuvių) ir naudoja mC4 korpusą. Vertinant naujas užduotis, Aya (13B) gerokai aplenkė BLOOMZ (176B) ir mT0 (13B): vidutinis tikslumas siekė 73,9 %, palyginus su kitais modeliais, kurie surinko 65,5 % ir 72,9 %. Dokumentacijoje minima, kad modelis palaiko ir mažai išteklių turinčias kalbas, įskaitant lietuvių, latvių, estų bei slovakų. Tačiau, Aya neturi mažo parametrų dydžio modelių, jo parametrų skaičius siekia 13B[25].

1.3. Teksto apdorojimas

1.3.1. Tokenizatoriai

Pirmasis LLM žingsnis – žalią tekstą paversti žetonų seka. Plačiausiai taikomi metodai:

1.4 lentelė. Tokenizacijos variantai

Tokenizatorius	Vokab. dydis	Naudojančios šeimos	Pliusai LT kalbai	Minusai LT kalbai
Byte-BPE	≈ 50 k	GPT-2, mGPT	Pilnas *Unicode*, diakritikai visada užkoduojami	Dažni lietuviški žodžiai skaidomi į 3–6 baitus
SentencePiece-BPE	32 k	LLaMA 2/3, Mistral, TinyLLaMA	Greitas, paprasta treniruoti	LT žodžiai retai patenka į mokymo žodyną, atsiranda „žetonų infliacija“
SentencePiece-Unigram	250–256 k	BLOOM, Gemma 2/3, EuroLLM	Daug LT morfemų ir visos diakritikos 1–2 žetonais	Didesnė atminties apkrova *embedding* sluoksnyje

Tokenizatorius gali nulemti modelio taiklumą įvairiose užduotyse, ypač susijusiose su mažiau naudojamomis kalbomis. Rust ir kt.[26] kiekybiniai rezultatai aiškiai parodo, kad kalbai pritaikytas

tokenizatorius pagerina modelio našumą: monolingvinių modelių vidutinisvertinimas yra 70,8 % prieš 69,7 % mBERT (Δ=+1,1 pp), o F1 – 82,3 % palyginimui 81,2 % (Δ=+1,1 pp). Pavyzdžiui, suomių kalboje monolingviškai treniruotam modeliui yra 69,9 % prieš 66,6 % mBERT'e (Δ=+3,3 pp), o turkų kalboje sentimentų analizėje monolingviško modelio tikslumas – 88,8 % prieš 86,4 % (Δ=+2,4 pp) . Galų gale, net 79 % (38/48) modelių-užduočių kombinacijų su vienakalbiu tokenizatoriumi lenkia originalų mBERT tokenizatorių . Taigi kalbai specializuotas tokenizavimas yra svarbi detalė, kuri ne tik sumažina perteklinį segmentavimą, bet ir reikšmingai pagerina tolimesnį modelio veikimą.

Iš senesnių šaltinių galima matyti, kad tokenizatorius turi daro didelę įtaką LLM taiklumui, ypač su retesnėmis kalbomis.

1.3.2. Įterpimo procesas

Kiekvienas žetonas $E : \mathbb{Z} \rightarrow \mathbb{R}^d$ projekcijoje gauna d-matį vektorių, kuriame užkoduojama semantika ir (per pozicijos bei segmento vektorius) sintaksinė informacija[27]. Dabartinė praktika:

- Žetonų įterptys (angl. *Token embeddings*)– mokosi kartu su modeliu; per LoRA galima keisti tik dalį jų [28].
- Pozicijos įterptys (angl. *Position embeddings*) – absoliutūs (*GPT-2*), sinusiniai (*LLaMA*), arba RoPE rotaciniai (*Gemma 3*)
- Segmento įterptys (angl. *Segment / modality embeddings*) – būtini *V-LLM* (*LLaMA 3.2*) ir *encoder-decoder* (*mT5 → Aya*) architektūrose[29].

Lietuvių kalboje gausios priesagos ir galūnės lemia žymiai platesnį žodžio formų spektrą. Jei žodynas neišmoksta visų detalių, įterpties sluoksnis stokoja reprezentacijų ir reikalingas papildomas treniravimas[30] arba sintetiniai duomenys.

1.3.3. Samprotavimo stiprinimas

Vien tik papildomi duomenys neužtikrina loginio samprotavimo. Tyrimai siūlo kelias kitokias naudojimosi technikas:

- *Chain-of-Thought* (CoT) – priverčia modelį eksplikuoti tarpinius žingsnius[31].
- *Self-Polish* (SP) – iteratyviai generuoja ir taiso atsakymą, mažindamas haliucinacijų skaičių[32].
- *Clue-And-Reasoning Prompting* (CARP) – suformuluoja užuominą ir paaiškinimą, didindamas klasifikavimo tikslumą semantiškai sudėtingose frazėse[33].
- *Example-Based Translation Learning* – pavyzdžių lyginimas, sėkmingai pagerinęs LLM vertimą žemo išteklių kalbose[34].
- *Fill-in-the-Middle* (FIM) – paprasta duomenų transformacija, leidžianti autoregresiniams modeliams mokytis redagavimo bei kodo užpildymo[35].

Dideli kalbos modeliai apdoroja tekstą naudodami pažangius natūralios kalbos apdorojimo metodus. Jie yra apmokyti apdoroti didelius tekstinių duomenų kiekius, kad suprastų kalbą, nustatytų modelius ir sukurtų panašų į žmogaus tekstą. LLM, pavyzdžiui, GPT-3, NLP užduotims atlikti naudoja tikslaus derinimo paradigmą *(*angl. *fine-tuning paradigm)*, kai jie iš anksto apmokomi dideliame teksto masyve, o tada tikslinami konkrečiuose anotuotuose duomenų rinkiniuose įvairioms užduotims, pavyzdžiui, teksto klasifikavimui ir kalbos generavimui, atlikti[33]. Be to, buvo pasiūlyti tokie skatinimo metodai, kaip minčių grandinės *(*angl. *Chain-of-Thought, CoT)* ir savipoliravimas *(*angl. *Self-Polish, SP),* siekiant pagerinti LLM samprotavimo gebėjimus, palaipsniui tikslinant pateiktas problemas ir tobulinant jų sprendimo procesą[32]. Šiais metodais siekiama spręsti uždavinius, su

kuriais susiduria LLM, spręsdami su samprotavimu susijusias užduotis, pavyzdžiui, spręsdami matematikos žodinius uždavinius ir atsakinėdami į loginiu argumentavimu pagrįstus klausimus. Įvairiose užduotyse įrodytas LLM efektyvumas apdorojant tekstą, tačiau jie vis dar susiduria su iššūkiais, susijusiais su samprotavimo užduotimis, kurias tyrėjai aktyviai stengiasi spręsti.

Siekiant spręsti LLM gebėjimo samprotauti trūkumo problemą, kai sprendžiami sudėtingi lingvistiniai reiškiniai teksto klasifikavimo užduotyse, pradėti taikyti tokie metodai kaip CARP[33]. Kitas metodas apima pavyzdžių lyginimą, kai LLM mokomi mokytis vertimo, ir šis metodas davė daug žadančių rezultatų gerinant LLM vertimo gebėjimus[34]. Be to, tyrimai parodė, kad autoregresiniai kalbos modeliai gali išmokti užpildyti tekstą pritaikius duomenų rinkiniui paprastą transformaciją, todėl buvo pasiūlyta modelius pagal nutylėjimą mokyti užpildyti vidurį *FIM*[35].

1.4. Dėmesio mechanizmai

Dėmesio mechanizmai didelės apimties kalbos modeliuose atlieka svarbų vaidmenį suprantant ir kuriant į žmogų panašų tekstą. LLM sėkmė labai priklauso nuo transformatoriaus modelio, kuris naudoja savąjį dėmesį, kad pasvertų skirtingų įvesties sekos dalių svarbą. Šis mechanizmas leidžia modeliui sutelkti dėmesį į svarbią informaciją ir neatsižvelgti į nereikšmingus ar perteklinius elementus, fiksuojant tolimąsias priklausomybes ir kontekstinius ryšius tekste [36].

Tačiau tradiciniai dėmesio mechanizmai transformatoriuose susiduria su mastelio didinimo problemomis dėl kvadratinio vykdymo laiko ir atminties sudėtingumo, ypač kai susiduriama su ilgomis sekomis. Šiems iššūkiams spręsti buvo sukurti įvairūs metodai, skirti aproksimuoti dėmesio mechanizmus. Tarp jų yra aproksimacijos naudojant retas matricas, mažo rango matricas arba jų derinį. Šiais metodais siekiama greičiau aproksimuoti įvairius dėmesio komponentus, nereikalaujant tikslių skaičiavimų. Nepaisant šių pasiekimų, tikslios dėmesio matricos aproksimacijos įrašuose per subkvadratinį laiką vis dar yra sudėtingas uždavinys. Naujausi pasiekimai, tokie kaip *KDEFormer*, padarė pažangą šioje srityje, pateikdami įrodomus aproksimacijas subkvadratiniu laiku, darant prielaidą, kad dėmesio matricos įrašai yra riboti. Šis modelis palaiko priežastinį maskavimą (angl. *casual masking*), kuris labai svarbus šiuolaikinėms transformatorių architektūroms, ir demonstruoja reikšmingą spartos padidėjimą praktiniuose taikymuose[37].

Tolesnė dėmesio mechanizmų raida akivaizdi įvedus retencijos tinklą *RetNet*[38], kuris yra galimas Transformatoriaus modelio įpėdinis didelių kalbos modelių atveju. *RetNet* perskirsto parametrus, kad būtų galima sąžiningai palyginti, ir demonstruoja panašius, o gal net geresnius rezultatus nei transformatoriai, ypač didėjant modelio dydžiui. Šis tinklas siūlo daug žadančią kryptį sprendžiant didelių kalbos modelių mastelio keitimo ir efektyvumo problemas[39].

Taip pat kylanti naujovė MDM, difuzijos būdu veikiantys modeliai gebantys geriau atlikti matematinius veiksmus ir generuoti originalesnius atsakymus[40].

Be to, dėmesio mechanizmų taksonomiją galima klasifikuoti pagal požymių vektorius, modelio užklausas arba bendruosius mechanizmus, nesusijusius nei su vienu, nei su kitu. Šią taksonomiją sudaro įvairios kategorijos ir subkategorijos, pavyzdžiui, su požymiais susiję, su užklausomis susiję ir bendrieji dėmesio mechanizmai. Kiekvienas tipas atlieka tam tikrus vaidmenis, pavyzdžiui, tvarko požymių daugybę, požymių lygius arba požymių reprezentacijas. Pavyzdžiui, bendro dėmesio mechanizmai leidžia modeliams kartu atkreipti dėmesį į kelių tipų įvestis, pavyzdžiui, vaizdus ir

tekstą, taip padidinant modelio gebėjimą integruoti ir interpretuoti sudėtingas duomenų įvestis, dar kitaip jau minėtus, multimodalinius duomenis[41].

1.5. Transformatorių architektūros

Vienas iš pagrindinių šių architektūrų komponentų yra dėmesio mechanizmas, neatsiejamas nuo transformatoriaus modelio. Šis mechanizmas apskaičiuoja kiekvieno įvesties sekos simbolio svertines reprezentacijas, remdamasis jo svarba kitiems simboliams. Paprastoje transformatoriaus architektūroje, kurią sudaro koduotojas ir dekoderis su keliais identiškais blokais, naudojami keli dėmesio (MHA) sluoksniai. Šie sluoksniai yra svarbiausi tvarkant įvesties žetonų sekas. Kalbos modeliuose, pavyzdžiui, BERT ir GPT, naudojami šios architektūros variantai, kuriuose daugiausia dėmesio skiriama dvikrypčiam (BERT) arba vienkrypčiam (GPT) informacijos apdorojimui[29].

Transformer-XL yra dar vienas svarbus patobulinimas, nes, palyginti su paprastu transformatoriaus modeliu, užtikrinamas didesnis efektyvumas ir galimybė treniruoti modelius su didesniais konteksto langais. Šioje architektūroje įdiegta pasikartojimo schema, leidžianti pakartotinai naudoti ankstesnių segmentų atvaizdus, gerokai pagreitinti vertinimą ir apdoroti ilgesnį kontekstą[42].

Nuolatinė transformatorių architektūrų raida akivaizdi moksliniuose tyrimuose, kuriuose nagrinėjamos įvairios architektūros modifikacijos ir paieškos strategijos. Tai apima vienodą atsitiktinę atranką ir koordinačių architektūros paiešką, kuriomis siekiama optimizuoti transformatorių modelius konkretiems duomenų rinkiniams ir užduotims, pabrėžiant nuolatines pastangas tobulinti ir gerinti šiuos modelius įvairioms taikomosioms programoms[16].

1.6. Išankstinio mokymo tikslai

Išankstinis didelių kalbos modelių mokymas yra sudėtingas ir daugialypis procesas, apimantis įvairius metodus ir tikslus. LLM, pavyzdžiui, GPT, BERT ir BART, paprastai iš anksto mokomi naudojant didžiulius informacijos rinkinius, pradedant žiniatinklio paieškomis ir baigiant konkrečiais duomenų rinkiniais, Vikipedija ar knygomis. Pavyzdžiui, GPT-3 buvo apmokytas naudojant daugiau nei 100 mlrd. ženklų, o BERT ir BART - apie 3,3 mlrd. ženklų. Todėl šių modelių dydžiai gali būti milžiniški[27].

Išankstinio mokymo procesą galima suskirstyti į skirtingas klases pagal modelio architektūrą ir tikslus. Autoregresiniai kalbos modeliai (pvz.: GPT) numato kitą sekos žodį ir pasižymi tuo, kad juose naudojama autoregresinio dekoderio dalis transformatoriaus architektūroje. Maskuotos kalbos modeliai (pvz.: BERT) prognozuoja užmaskuotą žodį sekoje, remdamiesi dvikrypčiu kontekstu, ir paprastai naudoja transformatoriaus architektūros kodavimo dalį. Kodavimo ir dekodavimo modeliai (pvz.: BART, T5) mokomi paversti įvesties seką išvesties seka, derinant autoregresijos ir užmaskuotos kalbos modelių aspektus[27].

Nuolatinis išankstinis mokymas - tai metodas, kai iš anksto apmokyti modeliai atnaujinami naudojant naujus duomenis, o ne mokomi iš naujo, nes tai laikoma veiksmingesniu ir ekonomiškesniu metodu. Tyrimai rodo, kad įšilimo (*angl. warm-up*) etapo trukmė nuolatinio išankstinio mokymo atveju neturi didelės įtakos tolesnių užduočių atlikimui. Tačiau didžiausias mokymosi greitis atlieka lemiamą vaidmenį siekiant subalansuoti naujų (*„downstream"*) ir pradinių (*„upstream"*) duomenų našumą[30].

Kitas išankstinio mokymo aspektas yra susijęs su supratimu, kaip skirtingi išankstinio mokymo tikslai daro įtaką LLM kalbinių savybių mokymuisi. Tokios užduotys kaip žodžių skaičiaus sakinyje, sakinio hierarchinės struktūros ar pagrindinio sakinio veiksmažodžio laiko numatymas, be kita ko, naudojamos tokių modelių kaip BERT lingvistiniams gebėjimams ištirti. Šių modelių rezultatai sprendžiant tokias užduotis gali skirtis priklausomai nuo taikomų išankstinio mokymo tikslų[43].

Taip pat labai svarbu stabilizuoti didelių kalbos modelių išankstinį mokymą, nes tai gali turėti įtakos modelio našumui. Įrodyta, kad tokie metodai kaip įterpimo sluoksnio normalizavimas ir mastelinis įterpimas *(angl. embedding layer normalization and scaled embedding)* padeda išvengti nuostolių ir gradiento šuolių, kurie dažniau pasitaiko didesniuose modeliuose. Šis stabilizavimas svarbus siekiant geresnių rezultatų, ypač didėjant modelio dydžiui[44].

1.7. Tikslus derinimas ir modelio pritaikymas

1.7.1. SFT

Klasikinis PEFT (angl. *Parameter-Efficient Fine-Tuning*) pavyzdys yra *supervised fine-tuning*, kai visa arba dalis modelio treniruojama tiesiogiai su žmogaus sudarytais, anotuotais pavyzdžiais. SFT suteikia stiprų signalą, bet vis dar gali būti brangus, jei atnaujinami visi sluoksniai.

1.7.2. LoRA

Mažo rango adaptacija (angl. *Low-Rank Adaptation, LoRA*) praplečia PEFT idėją įterpdama papildomas žemo rango *rank-r* projekcijas kiekviename daugiasluoksniame perceptrone ar *self-attention* projekcijoje. Tokiu būdu atnaujinami tik keli tūkstančiai naujų parametrų, o pagrindinio modelio svoriai lieka užšaldyti, taip sumažinant vaizdo atmintį (angl. *Video Random Access Memory,* VRAM) poreikį ir padidinant treniravimo greitį[28].

$$h = W_0 x + \Delta W x = W_0 x + BAx \tag{1}$$

- $x \in R^k$ – įėjimo vektorius į sluoksnį (pvz., ankstesnio sluoksnio paslėpta būsena).
- $h \in R^d$ – išvesties vektorius (pvz., kitos paslėptos būsenos).
- $W_0 \in R^{d \times k}$ – Užšaldyta, iš anksto apmokyta svorių matrica. Jos $d \times k$ elementai LoRA tobulinimo metu nesikeičia.
- ΔW = B A – išmokstamas mažo rango atnaujinimas matricai $\boldsymbol{W_0}$. Kadangi

$$\Delta W = BA, \quad B \in R^{d \times r}, \qquad A \in R^{r \times k}, \tag{2}$$

  ΔW rangas neviršija r (kai $r \ll \min(d, k)$).
- Parametrų skaičius
  - Pilnas fine-tune atnaujina $d \times k$ parametrus.
  - LoRA atnaujina tik B ir A, iš viso atnaujinama $d \cdot r + r \cdot k \ll d \cdot k$

1.7.3. QLoRA

QLoRA *(angl. Quantized Low-Rank Adaptation)* žengia dar toliau: bazinį modelį kvantuoja iki 4-bitų, o LoRA projekcijos lieka 16-bitų tikslumu. Derinant kvantavimą su žemo rango adapteriais pasiekiamas beveik tas pats tikslumas kaip ir LoRA, tačiau treniruoti galima viename A100 40 GB GPU ar net ribotuose TPU (angl. *Tensor Processing Unit*) [45,46].

1.7.4. LOMO ir kitos atminties optimizacijos

Kai GPU (angl. *Graphics Processing Unit)* atmintis tampa pagrindine problema, padeda *LOw-Memory Optimization* (LOMO) technika, viename žingsnyje sujungianti gradiento skaičiavimą ir parametrų atnaujinimą; taip nebereikia laikyti gradientų tenzorių ir smarkiai mažėja VRAM apkrova[22][47].

1.7.5. Derinimo metodų sparta

Kompaktiškai priderintus modelius galima efektyviai aptarnauti ir treniruoti naudojant įvairius servisus, kurie, remdamiesi algoritmais, tokiais kaip *FlashAttention*[9], optimizuoja ilgų sekų apdorojimą ir GPU atminties paskirstymą[48].

Per trumpą laiką atsirado labai daug skirtingų metodų ir metodologijų kaip pagerinti ar patobulint pokalbio modelius, tačiau ilgiausią laiką išliko keli populiariausi sprendimai parodyti **1.4 lentelėje**.

1.4 lentelė. Modelio derinimo metodų palyginimas

Metodas	Atnaujinamų parametrų dalis	VRAM poreikis	Tikslumas (lyginant su pilnu derinimu)	Pastabos
Pilnas derinimas	100 %	Didžiausias	Aukščiausias	Reikalauja daug epochų ir didelių GPU resursų
SFT (pilnas)	100 %	Didelis	Aukštas	Mažiau epochų
LoRA	≪ 1 %	Žemas	~1-2 % praradimas	Paprasta integruoti, gerai tinka daugeliui užduočių
QLoRA	≪ 1 % (adapteriai)	Labai žemas	~2 % praradimas	4-bitų bazė + 16-bitų LoRA
LOMO + LoRA/QLoRA	≪ 1 %	Minimalus	~2 % praradimas	Skirta ribotiems GPU resursams

Dėl kompiuterinių resursų stokos negalima taikyti į pilno mokymo metodus, todėl svarbiausi modeliai treniravimo etapui lieka LoRA ir QLoRA

1.5 lentelė. LoRA ir QLoRa spartos palyginimas

Metodas	Modeliai	Resursai	Treniravimo laikas
LoRA	Mistral,, GPT-2, GPT-3	Reikalinga panaši sistema kaip ir pilnam treniravimui	iki **3×** didesnis sparta negu pilnas treniravimo metodas [28]
QLoRA	LLaMA 65 B (Guanaco)	vienas 48 GB GPU (pvz., A100)	treniravimas gali trukti parą[46]

Praktikoje, jei modelis tilps turimoje atmintyje, LoRA aukštesnis per vienetinį žingsnį pralaidumas leidžia jį treniruoti greičiau nei QLoRA. QLoRA stiprybė – galimybė *fine-tune* labai didelius modelius ribotuose resursuose, nors kvantizacijos papildomas sudėtingumas sulėtina treniravimą.

1.8. Lietuvių kalbai treniruotų modelių apžvalga

Lt-LLaMA 2 (7 B / 13 B)[49] – pirmasis visiškai pilnai pertreniruotas (atnaujinti visi modelio svoriai, o ne tik adapteriai) atviras LLM lietuvių kalbai. Mokymas vyko dviem etapais.

Pirmasis etapas tęstinis *pre-training* su ~1,9 mlrd. lietuviškų žetonų iš *CulturaX* korpuso, apimančio spaudą, „Vikipediją“, forumus ir kt.

Sekantis RLFH su dviem sintetinės kilmės rinkiniais:

- Alpaca-LT 52 k – anglų „Alpaca“ instrukcijos automatiškai išverstos ir GPT sugeneruoti atsakymai (*Self-Instruct* metodika).
- LTQAv1 7 k – ekspertų sudarytas Q/A rinkinys apie Lietuvos kultūrą.

Pilnas parametrų atnaujinimas leido 13 B variantui sumažinti perplexity nuo 13,9 iki 3,45, o 7 B – nuo 17,5 iki 3,81; modelis užėmė pirmąsias vietas lietuviškuose ARC, HellaSwag ir TruthfulQA testuose[49]. Svarbiausia – kartu su svoriais publikuoti ir visi mokymo bei vertinimo rinkiniai, todėl Lt-LLaMA 2 tapo pagrindiniu atskaitos tašku lietuvių NLP tyrimuose.

Nors Lt-LLaMA 2 buvo treniruotas pilnu modelio parametrų atnaujinimu (t. y. nebuvo taikytas PEFT metodas), šis darbas kartu pateikia išsamią apžvalgą apie kitų regioninių LLM tendencijas. Remiantis jų analize, net 19 iš 20 nagrinėtų atvirojo kodo Europos kalbų modelių buvo treniruoti pilnu režimu, o LoRA, PEFT metodais – tik pavieniai atvejai[49]. Tačiau autorių teigimu, būtent PEFT metodai, nepaisant kiek mažesnio tikslumo, leidžia išlaikyti bazinio modelio architektūrą, sumažina treniravimo kaštus ir padaro eksperimentus įmanomus ribotų išteklių aplinkoje, todėl jie gali tapti strategiškai svarbūs tolimesniam mažųjų kalbų NLP vystymui.

1.9. Išvados

Išsami nagrinėtos literatūros apžvalga parodė, kad pilnai pertreniruotas „Neurotechnology“ *Lt-LLaMA 2* modelis tapo kertiniu lietuvių NLP modeliu. Jo mokymo schema yra nuoseklus treniravimas naudojant ~1,9 mlrd. CulturaX žetonų ir vėlesnis RLHF etapą papildęs, sintetiškai sugeneruotų instrukcijų rinkinys – leido gerokai sumažinti teksto suvokimą ir užimti pirmaujančias pozicijas lietuviškuose *ARC, HellaSwag ir TruthfulQA* bandymuose[49]. Vis dėlto pats modelio mokymas kainavo aštuonių H100 GPU resursus ir tūkstančius valandų, todėl toks kelias nėra tvarus, kas ir pabrėžia efektyvesnių adaptacijos strategijų, nagrinėjimų svarbą šiame darbe.

Priešingai, PEFT įkūnijantys metodai LoRA ir QLoRA leidžia atnaujinti mažiau nei vieną procentą modelio svorių, pagrindinį mokymosi tinklą išlaikant 4–16 bitų formatu. Literatūroje cituojami projektai rodo, kad toks pritaikymas išsaugo didžiąją dalį tikslumo, o eksperimentus galima vykdyti viename A100 ar net silpnesnėse GPU aplinkose. Vadinasi pilno *pretraining* privalumai laikytini disproporcingai brangiais, kai turimi šiuolaikiniai PEFT scenarijai gali bandyti juos pakeisti.

Tolesni tyrimai atsiremia į tai, kad LLaMA 3.2[20] jau siūlo 128 tūkst. žetonų kontekstą, platesnį > 20 kalbų žodyno dengimą ir geresnį lietuvių kalbos supratimą nei LLaMA 2[15], o Gemma 3[23] papildomai pasižymi 256 tūkst. *unigram* žodynu ir multimodalumu. Šios architektūros pateikia pakankamai tvirtą pagrindą, kad LoRA adapterių treniravimas galėtų pasiekti bent dalį Lt-LLaMA 2 rezultatų už mažiau nei dešimtadalį resursams reikalingų išlaid;. Taip pat, jos leidžia eksploatuoti ilgo konteksto užklausas ir morfologiškai palankesnę tokenizaciją.

Galiausiai, Lt-LLaMA 2 atvejis įrodė, jog sintetiniai duomenys nesvarbu, ar tai LLM sugeneruotos instrukcijos, ar atgaliniu vertimu gautos paralelės, išlieka patikimiausia strategija mažų išteklių kalbai. Todėl planuojant LoRA adaptaciją LLaMA 3.2 ir Gemma 3 modeliams, dėmesys bus

skiriamas naujų aukštos kokybės instrukcijų, dialogų ir naujų klausimų ir atsakymų rinkinių kūrimui, pasitelkiant automatizuotus vertimo metodus.

Teorinė ir empirinė analizė rodo, kad praktiškiausias kelias tolesniame etape yra LoRA pagrindu pritaikyti LLaMA 3.2 ir Gemma 3 modelius lietuvių kalbai. Sistemingai lyginti jų našumą su pilnai ištreniruotu Lt-LLaMA 2 ir vertinti, kaip adapterių metodai, duomenų kilmės bei sugeneruotos duomenų rinkinio dalys veikia modelių generuojamo teksto kokybę, kalbos supratimą ir gebėjimą spręsti semantiškai sudėtingas užduotis, siekiant pademonstruoti adaptuotų modelių konkurencingumą ir efektyvumą.

2. Duomenys ir metodai LLM treniravimui

2.1. LLM SFT duomenų rinkiniai

Kuriami sintetiniai rinkiniai, naudotasi egzistuojančiais rinkiniais ir jų hibridiniais variantais.

Lietuvių kalbai pasižyminčiai ribotais viešais ištekliais pasirinkta mišraus rinkinio strategija: žmogaus kuruotos instrukcijos (Aya) pateikia semantiškai griežtus pavyzdžius, didelio masto tinklo tekstai (mC4-LT) užpildo bendrinę leksiką, o CulturaX prideda dialektų ir kultūrinių kontekstų. Galiausiai, temines spragas užpildo sintetiniai pavyzdžiai, generuoti Kiln aplinkoje.

2.1.1. Aya duomenų rinkinio paruošimas

Aya duomenų rinkinys yra atvirai prieinama kolekcija, skirta daugiakalbių modelių mokymui. Pagrindinis tikslas yra spręsti kalbos spragą NLP srityje, teikiant aukštos kokybės, žmogaus kuruojamus nurodymų sekimo duomenis 65 kalbomis, įskaitant lietuvių kalbą. Aya duomenų rinkinio rinkimo procesas apėmė išsamų bendradarbiavimą su pasauliniu bendradarbių tinklu per Aya anotacijų platformą. Ši platforma palaikė 182 kalbas ir dialektus, leidžiant sklandžiai kalbantiems asmenims iš 119 šalių teikti aukštos kokybės anotacijas.

Aya duomenų rinkinys:

- Lietuvių kalbos duomenų rinkinys turi 916 įrašų.
- Lietuvių kalbos duomenų rinkinio dydis yra apie 1.28 MB.
- Anglų kalbos duomenų rinkinys turi 3944 įrašus.
- Anglų kalbos duomenų rinkinio dydis yra apie 4.80 MB.

„Aya Collection“ yra išsamus duomenų rinkinių rinkinys, kuris yra žymiai didesnis ir įvairesnis nei Aya duomenų rinkinys. Jis apima įvairias dalis ir dydžius.

2.1 lentelė "Aya Collection" rinkinio pasiskirtymas ir jų dydžiai

Dalis	Įrašai	Dydis (GB)
train_english	1 000 000	10
validation_english	200 000	2
test_english	100 000	1
train_lithuanian	500 000	5
validation_lithuanian	100 000	1
test_lithuanian	50 000	0.5

Aya kolekcija, kurią sudaro 513 milijonų įrašų 114 kalbų, buvo sukurta transformuojant ir verčiant esamus duomenų rinkinius į nurodymų ir atsakymų poras. Ši kolekcija yra viena iš išsamiausių daugiakalbių nurodymų sekimo patobulinimo duomenų bazių, kuri šiuo metu yra prieinama. Aya užtikrina žmogaus anotuotus nurodymų–atsakymų pavyzdžius, kas ypač svarbu tobulinant kalbos modelio gebėjimą reaguoti į konkrečias instrukcijas. Tačiau pasirodė, kad vien šio rinkinio gali nepakakti plačiam kalbos masyvui sudaryti, todėl nuspręsta duomenis papildyti kitais šaltiniais.

2.1.2. mC4 duomenų rinkinio paruošimas

„Multilingual Colossal Clean Crawled Corpus“ (mC4) – tai itin didelis tinklalapių turinio rinkinys, išfiltruotas iš „Common Crawl“ archyvo. Nors jį sudaro daugybė kalbų, mus domina būtent lietuviškoji dalis.

- Apimtis: ~11,2 mln. lietuviškų įrašų (iš ~160 tūkst. skirtingų svetainių).
- Teminis spektras: nuo naujienų portalų iki asmeninių tinklaraščių, todėl turinys labai įvairus.

Pavyzdinė analizė (50 000 dokumentų):

- Vidutinis teksto ilgis: 2 257,75 simbolio
- Mediana: 1 625,00 simbolių
- Unikalių domenų: 12 401
- Vidutinis žodžių skaičius dokumente: 266,35
- Vidutinis sakinių skaičius: 29,00
- Vidutinis žodžio ilgis: 6,73 simbolio

Ši statistika rodo, jog mC4 pasižymi ne tik gausia apimtimi, bet ir didele temų, stilių, žodyno įvairove, galinčia žymiai praplėsti kalbos modelio žinias.

Aya rinkinys yra palyginti nedidelis, mC4 suteikia daug didesnį tekstų masyvą – padeda išmokti platesnio žodyno, semantinių konstrukcijų ir padidina modelio atsparumą įvairių stilių turiniui.

CulturaX – tai papildomas kultūrinių tekstų rinkinys, praplečiantis modelio supratimą apie regioninę specifiką, tradicijas bei istorinius šaltinius. Jis apima įvairius regioninius duomenis iš Baltijos šalių bei kitų Rytų Europos dalių:

- Ryšys su C4: dalis duomenų buvo papildomai paimta iš C4 archyvo, tačiau peržiūrėta kultūros paveldo specialistų, išmetant abejotinos kokybės fragmentus.
- Kalbos: be lietuvių, rinkinyje yra latvių, lenkų, baltarusių tekstų.
- Apimtis: ~2,5 mln. straipsnių, esė, regioninių naujienų, paveldosaugos dokumentų ir pan.
- Specifinės ypatybės: retų terminų, dialektų, etnografinės leksikos pavyzdžiai, kurie paprastai neras vietos bendriniuose duomenų rinkiniuose.

CulturaX padeda modeliui susipažinti su ne tik kasdiene, bet ir kultūriškai reikšminga kalbine medžiaga, pvz.: vietovardžių, liaudies papročių, kitų unikalių elementų pavadinimais. Tai praplečia generuojamų tekstų universalumą bei tikslumą, ypač kalbant apie kultūrinius ar istorinius kontekstus.

Trijų duomenų rinkinių – Aya, mC4 (lietuviška dalis) ir CulturaX – kombinacija sudaro tvirtą pagrindą mūsų kalbos modelio mokymui. „Aya Dataset“ suteikia tikslių, žmogaus kuruojamų nurodymų–atsakymų pavyzdžių, mC4 užtikrina milžinišką bendrinio lietuvių kalbos turinio spektrą, o CulturaX papildo kultūros ir regioniniais tekstais. Tokia strategija leidžia plačiai apimti ne tik standartinius kalbos aspektus, bet ir specifinius, retesnius atvejus.

2.1.3. Sintetinio lietuviško duomenų korpuso generavimas

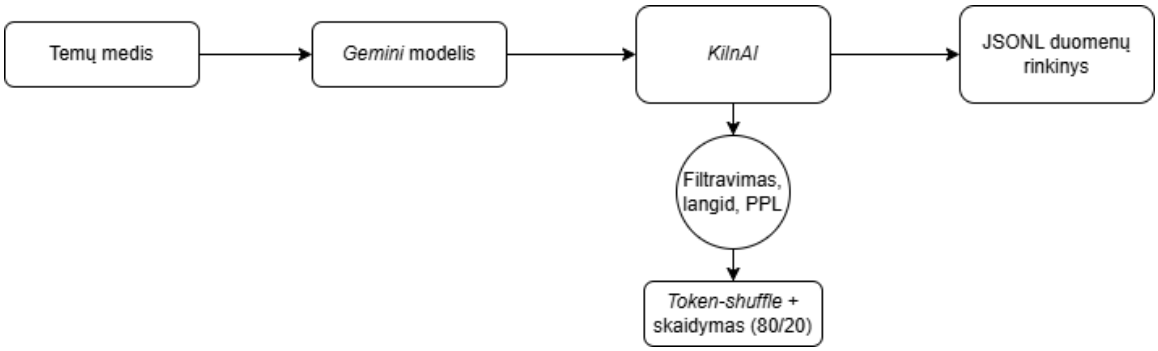
Siekiant padidinti treniravimo duomenų įvairovę ir aprėpti lietuvių kalbos vartosenos kontekstus, kurie sunkiai pasiekiami viešuose rinkiniuose (pvz., specializuotas diskursas, retorika, nišiniai terminai), buvo sukurtas dirbtinis lietuviškas duomenų rinkinys naudojant Kiln – atvirojo kodo sintetinių duomenų generavimo sistemą. Šis įrankis leidžia sąveikauti su dideliais LLM modeliais

(šiuo atveju Gemini) per temų kūrimo medžio principu sąsają ir generuoti tekstus su RLFH elementais.

Generavimo eiga:

- Gemini (kviečiamas per Kiln sąsają), dėl aukštos kokybės lietuvių kalbos suvokimo ir vertėjavimo galimybių.
- Temų medis (iki 2 lygių gylio), sugeneruotas automatiškai su papildomomis nišinėmis sritimis (pvz., „žaidimų lokalizacija“, „Lietuvos teisė“).
- **N**urodymas generuoti šiuolaikiškus, įvairių stilių tekstus (naujienos, pokalbiai, techninis turinys).
- Pridėtas žmogaus kuravimas: „Venk angliškų skolinių, linksniuok lietuviškas pavardes“ – padėjo pagerinti morfologinį natūralumą.

Duomenys treniravimui turi būti išskaidomi 80/20 principu. 80 pro. Treniravimo duomenys, 20 proc. tikrinimo duomenys. Detalesnis vaizdas rodomas plane **2.1 pav.**



2.1 pav. Sintetinio duomenų generavimo darbo eiga

Ištestavum Kiln sintetiniai rinkiniai pasižymėjo labai vieninga stilistika, sklandžia sakinių struktūra ir morfologiškai tiksliu lietuvių kalbos panaudojimu. Tai suteikė galimybę mokyti modelį palankiomis sąlygomis, be triukšmo iš žiniatinklio šaltinių, bei buvo ypač tinkamas pirmiesiems *fine-tuning* etapams

2.1.4. Duomenų rinkinių panaudojimas

Kalbos modelio pritaikymas lietuvių kalbai reikalauja ne tik žalių duomenų, bet ir kruopščiai subalansuoto bei semantiškai įvairaus rinkinio. Šiame projekte panaudoti keturi skirtingo pobūdžio duomenų šaltiniai:

- Sintetinis duomenų rinkinys, generuotas su Kiln sistema ir Gemini modeliu;
- Aya instrukcijų sekimo rinkinys;
- Lietuviškoji mC4 korpuso dalis (C4-LT);
- CulturaX, apimantis regioninį ir kultūrinį turinį.

Šių šaltinių derinimas įgalina tiek instrukcijų semantiką, tiek realios kalbos stilių.

Rinkinių apjungimo logika:

- Aya vertingas dėl anotuotų „instruct“ stiliaus pavyzdžių, bet jo apimtis nepakankama platesniam modeliui;
- mC4 suteikė didelę apimtį, bet turėjo triukšmo, todėl prieš naudojimą buvo atliktas filtravimas pagal kalbos atpažinimą ir perpleksiškumą;

- CulturaX papildė modelį specifiniais vardais, terminais, tradicinėmis konstrukcijomis, pritaikomais tik Rytų Europos kalbinėje erdvėje;
- Kiln sintetinis rinkinys buvo naudojamas kaip „švarus startas“ – įgalinantis efektyvų modelio adaptavimą, ypač LoRA *fine-tune* etapui.

Darbe visi kalbos modeliai treniruoti taikant vieningą „sintetinio branduolio“ metodiką – tai reiškia, kad visi eksperimentai paremti dirbtinai sugeneruotu Kiln sintetiniu duomenų rinkiniu, sukurtu specialiai lietuvių kalbos modeliui. Šis rinkinys pasižymėjo aukšta stilistine kokybe, semantiniu nuoseklumu ir mažu triukšmo lygiu, todėl tapo patikimu pagrindu modelių tobulinimui.

Papildomai nepamiršta hipotezė dėl duomenų praplėtimo strategijos, kurią sudaro filtruotas hibridinis korpusas (C4-LT, Aya, CulturaX) – naudotas tik su vienu Gemma 1B modeliu, siekiant įvertinti lankstesnio semantinio spektro įtaką modeliui.

2.1 lentelė. Pagrindiniai duomenų rinkiniai

Duomenų šaltinis	Kilmė / Tipas	Apimtis (tokenai)	Paskirtis
Kiln sintetinis	Gemini / curated	~7.7M	Visų modelių mokymas (LoRA)
mC4 (lietuvių)	Common Crawl	~50M ištraukta	Papildymas hibridiniame variante
Aya (instruktiniai)	Žmogaus anotuotas	~2M (LT + vertimai)	Panaudota tik filtruotame hibride
CulturaX	Kultūrinis turinys	~2.5M	Praturtintas semantinis testas

2.2 lentelė. Duomenų naudojimo struktūra pagal modelius

Modelis	Naudoti duomenys	Metodas	Pastabos
LLaMA 3.2 1B	Kiln sintetinis	LoRA	Bazinis eksperimentas
LLaMA 3.2 1B Instruct	Kiln sintetinis	LoRA	Instruct tuning testas
Gemma 1B	Kiln sintetinis	LoRA	Pirmas bandymas
Gemma 1B (hibridinis)	Kiln + atfiltruotas mC4/Aya/CulturaX	LoRA	Tik vienas modelis naudotas su mišiniu
Gemma 4B	Kiln sintetinis	LoRA	Padidintas modelio pajėgumas, tik Kiln

Skaidymas ir paruošimas:
- Treniravimas ir testavimas paskirtas: 80% / 20%
- Sintetinio rinkinio atranka ribojama iki ~2M-7M tokenų (GPU taupymo tikslais)
- Hibridinis rinkinys buvo sudaromas tik vienam eksperimentui, filtravus pagal:
    - kalbos ID (FastText, Lingua ≥ 0.95 *confidence*)
    - *perplexity* (KenLM-LRT, PPL < 200)
    - dublikatus (SHA-1 ir MinHash)

2.1.5. Modelių treniravimo parametrai

Llama 3.2 ir Gemma 3 modeliai buvo treniruojami naudojantis TogetherAI platforma dėl jų kompiuterių spartos ir palaikomų funkcijų. Deja, dėl skirtingų modelių architektūrų hiperparametrai turėjo skirtis (**2.3 lentelė**, **2.4 lentelė**), .

2.3 lentelė. LLaMA 3.2 modelių treniravimo parametrai

Modelis	Bėgimas	Epochs	Batch size	LoRA rank	LoRA alpha	Learning rate	Scheduler	Trainable modules	Pastabos
LLaMA 3.2 1B	First pass	1	32	64	128	1e-5	cosine	all-linear	Sintetiniai duomenys
LLaMA 3.2 1B	Second pass	2	32	32	64	1e-5	cosine	all-linear	Sintetiniai duomenys

2.4 lentelė. Gemma 3 modelių treniravimo parametrai

Modelis	Bėgimas	Epochs	Batch size	LoRA rank	LoRA alpha	Learning rate	Scheduler	Trainable modules	Pastabos
Gemma 3 1B	First pass	1	8	64	128	1e-5	cosine	all-linear	Sintetiniai duomenys
Gemma 1B 3 (Hybrid)	One pass	1	8	64	128	1e-5	cosine	all-linear	Hibridinis testas
Gemma 3 4B	Second pass	2	8	32	64	1e-5	cosine	all-linear	Sintetiniai duomenys
Gemma 3 4B	First pass	1	8	64	128	1e-5	cosine	all-linear	Sintetiniai duomenys

Pasirinkti hiperparametrai atsižvelgia į modelio dydį, GPU išteklius ir eksperimentų tikslus. *First pass* metu visiems modeliams naudotas didesnis LoRA rangas (64), siekiant išnaudoti maksimalų semantinį potencialą. *second pass* sumažintas iki 32, kad pagerėtų parametrų efektyvumas ir imituotų *instruct-tuning* scenarijų.

LLaMA 3.2 modelis, veikiantis FP16/BF16 režimu, pilnai užkrauna ~24 GB (1 mlrd × 2 baitai). Su 80 GB VRAM (H100/L40) *batch size* = 32 įmanomas be atminties trūkumo.

Gemma 3 struktūra su LoRA adapteriais generuoja kur kas didesnes atminties sąnaudas: vienam 1B bandymui(*pass*) su *batch size* = 8 prireikia ~20–25 GB (svoriai, gradientai, adapterių būsena, aktyvacijos). Todėl didinti iki 32 reikštų > 80 GB VRAM, kas viršija H100 galimybes, ir eksperimentuose buvo naudojamas *batch size* = 8.

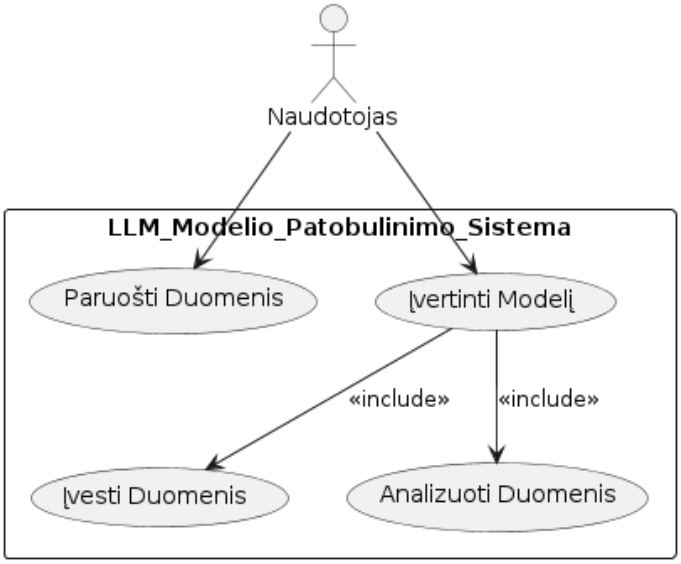
Visuose eksperimentuose pasirinktas *cosine learning rate scheduler* (*warmup ratio* = 0, *cycles* = 0.5), *weight decay* = 0, o visi *all-linear* sluoksniai pažymėti treniruojamaisiais. Tai leido efektyviai valdyti atminties suvartojimą per LoRA adapterius, neišardant išmoktų svorių struktūros.

2.2. Alternatyvios architektūros (atmesti variantai)

Pradinėje projekto fazėje buvo svarstoma naudoti ir kitas lietuvių kalbai pritaikytas didesnes kalbos modelių architektūras, tokias kaip LLaMA 2 7B/13B, mGPT, ar EuroLLM, pasitelkiant QLoRA bei kitus parametrų efektyvumo metodus. Taip pat planuota treniruoti šiuos modelius didesniuose duomenų korpusuose, pasitelkiant sujungtus Aya, mC4 ir CulturaX šaltinius.

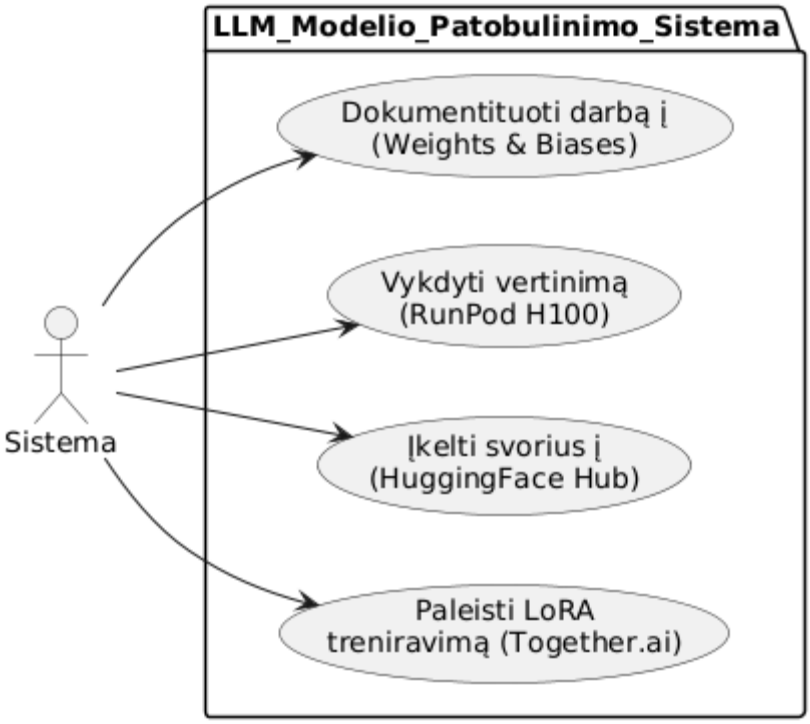
Tačiau dėl realių GPU resursų ribojimų (laiko, atminties ir eksperimentų kaštų), šie modeliai nebuvo taikyti praktikoje. Visas dėmesys buvo sutelktas į lengvesnių architektūrų (1B ir 4B) LoRA treniravimą su sintetiniais duomenimis.

2.3. Projekto Dizainas



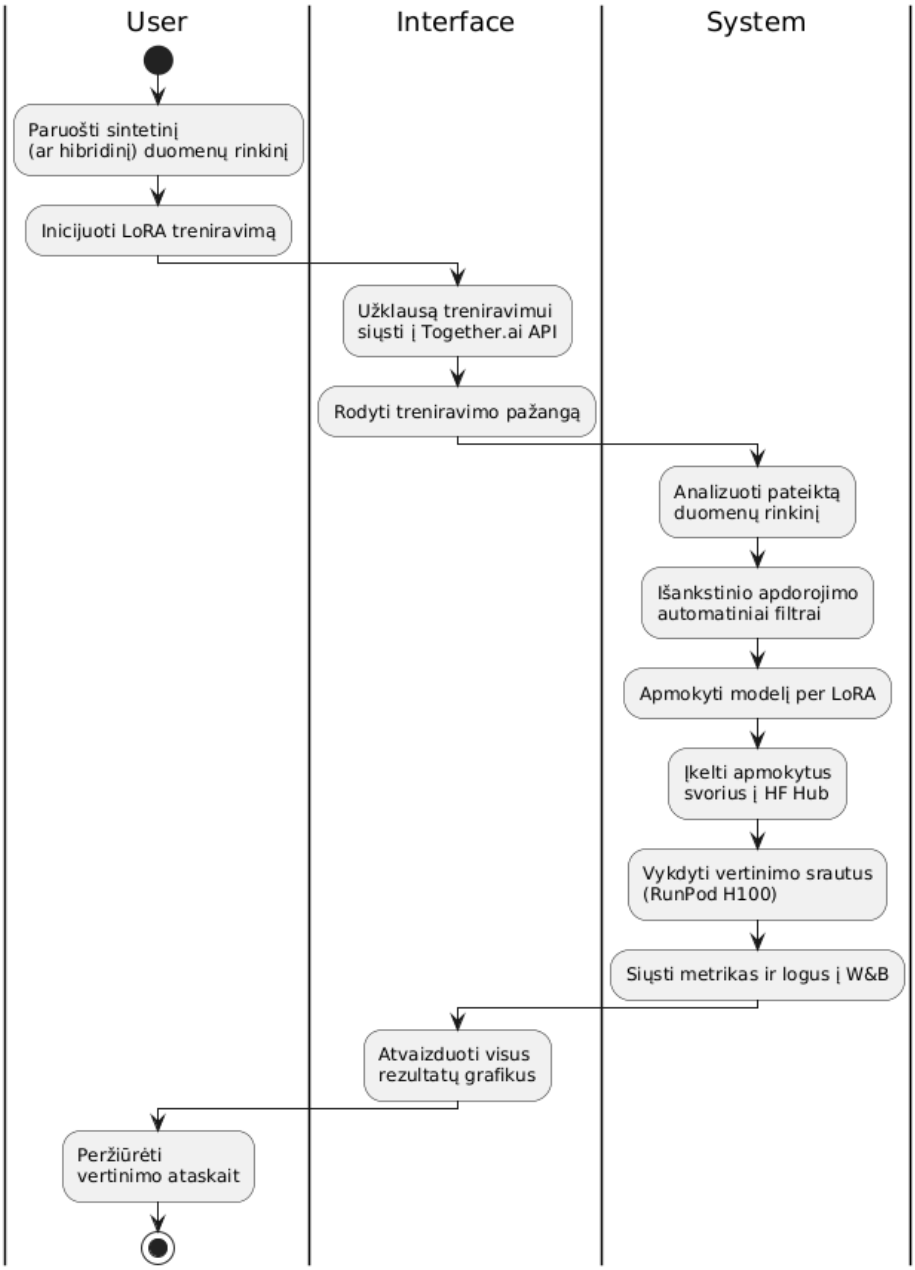
2.2 pav. Naudotojo *Use Case* diagrama

Naudojimo atvejo diagrama aprašo sąveikas tarp naudotojo ir LLM modelio patobulinimo (angl. *Fine-Tuning*) sistemos. Pagrindinės naudotojo veiklos apima duomenų paruošimą ir modelio vertinimą. „Duomenų paruošimo“ etape naudotojas surenka ir suformatuoja duomenų rinkinį, užtikrindamas, kad jis būtų tinkamas mokymui. Šis žingsnis yra labai svarbus, nes jis nustato pagrindą visam patobulinimo procesui. „Modelio vertinimo“ etape naudotojas vertina patobulinto modelio našumą. Naudotojas pateikia naujus testavimo duomenis, o sistema analizuoja šiuos duomenis, kad įsitikintų, jog jie tinkami vertinimui. Sąveika parodo naudotojo vaidmenį tiek rengiant duomenų rinkinį, tiek vertinant modelio našumą.



2.3 pav. Sistemos *Use Case* diagrama

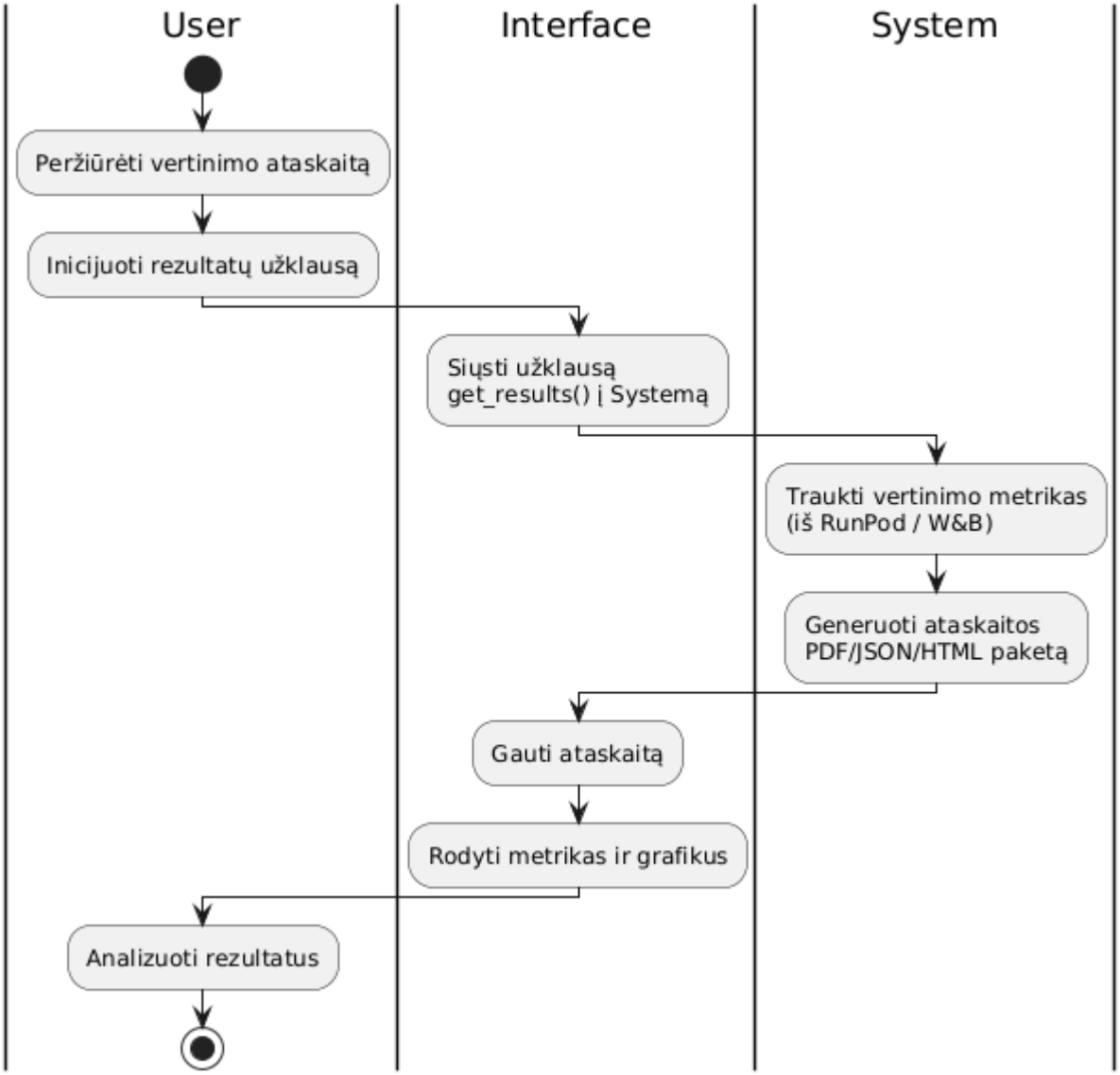
Sistema automatizuoja visą LLM modelio patobulinimo eigą: per Together.ai API ji inicijuoja LoRA fine-tuning užklausą, perduodama paruoštą sintetinį (ir, prireikus, hibridinį) duomenų rinkinį bei hiperparametrų konfigūraciją. Po to, baigus treniravimą, automatiškai įkelia apmokytų svorių paketą su atnaujintu modeliu į HuggingFace, tuomet iš pasinaudoja nauju modeliu ir jį įtraukia ir paleidžia standartizuotus vertinimo procesus per RunPod, su H100 GPU pagalba (TruthfulQA, MMLU, ARC-LT ir kt.), o visos treniravimo bei vertinimo metrikos kartu su gedimais ir anomalijomis keliauja į Weights & Biases stebėsenos skydelius – taip užtikrinant vientisą, patikimą ir visiškai automatizuotą modelio adaptavimo bei vertinimo procesą.



2.4 pav. *Activity* diagrama tobulinti modeliui

Pirmiausia naudotojas paruošia sintetinį (arba hibridinį) duomenų rinkinį ir per sąsają inicijuoja LoRA treniravimą. Sąsaja (CLI/Kiln UI) gauna šią užklausą ir perduoda ją Together.ai API, tada

realiu laiku atvaizduoja treniravimo pažangą. Gavusi duomenis, Sistema automatiškai analizuoja ir išankstinio apdorojimo filtrais paruošia korpusą, atlieka LoRA fine-tuning, įkelia naujai apmokytus svorius į HuggingFace, vykdo standartizuotus vertinimo procesus RunPod kartu su H100 GPU aplinkoje ir siunčia visus treniravimo bei vertinimo logus į Weights & Biases. Po to, Sąsaja surenka grąžintas metrikas ir grafikus, juos pateikia naudotojui, kuris galiausiai peržiūri vertinimo ataskaitą ir gali priimti tolesnius sprendimus dėl modelio adaptacijos ar papildomų eksperimentų.



2.5 pav. *Activity* diagrama gauti rezultatus

Naudotojo rezultatus gavimo eiga prasideda, kai Naudotojas per sąsają inicijuoja rezultatų užklausą. Sąsaja iškviečia sistemos API metodą (*get_results()*), kuriuo paprašo surinkti visus treniravimo ir vertinimo duomenis. Sistema tuomet užklausia *RunPod* bei *Weights & Biases* duomenų saugyklas, apdoroja gautas metrikas ir grafikus, sugeneruoja galutinę ataskaitą (pvz.: PDF arba JSON) ir perduoda ją atgal Sąsajai. Galiausiai Sąsaja pateikia interaktyvius grafikus ir skaitmeninius rodiklius Naudotojui, kuris juos peržiūri ir interpretuoja, priimdamas sprendimus dėl tolimesnių tolimesnių modelio optimizavimo veiksmų.

2.4. Funkciniai ir nefunkciniai reikalavimai

Projekto metu norint atrinkti tinkamus iš anksto apmokytus kalbos modelius, kuriuos būtų galima toliau pritaikyti lietuvių kalbos užduotims, numatyti šie kriterijai:

- Architektūra – priežastinis kalbos modelis (angl. *causal language model*)
- Modelis privalo būti paremtas autoregresiniu dekoderiu, t. y. generuoti tekstą seka iš kairės į dešinę. Toks reikalavimas leidžia modelį sklandžiai adaptuotis prie instrukcijų (angl. *instruction following*) bei prie kitų generavimo užduočių.
- Siekiant užtikrinti, kad modelio mokymo procesas neviršytų turimų skaičiavimo išteklių (GPU atminties, mokymo laiko), modelio dydis neturėtų viršyti ~5 mlrd. Parametrų.
- Modelis turi turėti priemones arba galimybę efektyviai pritaikyti tokenizatorių lietuvių kalbai (pvz.: *BPE, SentencePiece*).
- Pirmenybė teikiama modeliams, kurie jau yra matę arba gali būti nesunkiai adaptuoti prie lietuvių kalbos duomenų.
- Modelis turi būti pritaikytas PEFT metodams (QLoRA, LoRA) ar panašiems parametrų efektyvumui optimizuoti skirtiems metodams. Modelis turėtų būti suderinamas su tokiomis bibliotekomis ir leisti užšaldyti dalį svorių arba juos adaptuoti be didelių kodo pakeitimų.
- Modelis galėtų būti testuojamas tarptautiniu mastu taikomomis metrikomis – pavyzdžiui *Truthful QA, ARC, WinoGrande, MMLU, HellaSwag, GSM8K*.

2.4.1. Funckiniai

- Sistema verčia angliškąją duomenų rinkinio dalį į lietuvių kalbą.
- Išversti duomenys integruojami su esamais lietuvių kalbos tekstais, sudarant dvikalbį duomenų rinkinį.
- Pritaikomas modelio specifinis individualus tokenizatorius, naudojant sujungtą duomenų rinkinį.
- Modelis (Gemma 3, Llama 3.2) patobulinamas naudojant naujai pateiktus duomenis per programinę proceso eigą.
- Sistema įvertina modelio našumą, lygindama senojo modelio ir patobulinto modelio atsakymus.
- Rezultatai surenkami ir išsaugomi.

2.4.2. Nefunkciniai

- Sistema efektyviai tvarko didelius duomenų rinkinius, įskaitant dalinę perdirbtą c4, Aya ir dirbtinai pagamintas kolekcijas.
- Nauji duomenų rinkiniai nesukelia katastrofiško užmaršumo.
- Sistema sugeba teikti patikimus, semantiškai teisingus rezultatus.

2.5. Įrankiai

Projektas vykdomas debesų kompiuterijos platformoje, kuri užtikrina galingus skaičiavimo resursus:

- NVIDIA H100 arba L40 GPU su 48–80 GB VRAM
- 8 vCPU procesoriai
- 94 GB operatyvinės atminties (RAM)
- 200 GB sistemos diskas
- 500 GB tinklo talpos diskas
- Ubuntu 20.04 LTS operacinė sistema

Programavimo aplinka ir įrankiai:

- IDE: *Visual Studio Code* (*VSCode*) – pasirinktas dėl lankstumo, plėtinių ekosistemos ir *Python/Git* integracijos. Naudotas ir *Windsurf*, dėl integruoto DI asistento prototipavimo metu.

- Versijų kontrolė: *Git* – naudojamas kodo versijavimui ir komandiniam darbui
- Aplinkos valdymas: Python virtualios aplinkos (venv) – izoliuotoms priklausomybėms ir reproducibility

Programavimo kalbos ir pagrindinės bibliotekos:
- Python – pagrindinė programavimo kalba
- PyTorch – giliojo mokymosi sistema su CUDA palaikymu GPU spartinimui
- Transformers (Hugging Face) – LLM modelių apdorojimui, treniravimui ir inferencijai
- Datasets – efektyviam duomenų įkėlimui, transformacijai ir valdymui
- PEFT (LoRA) – lengvam modelių adapterių integravimui
- TRL – reinforcement learning arba SFT scenarijams su instrukcijomis
- SentencePiece – daugiaformanės kalbos tokenizatorių generacijai

Duomenų generavimas ir filtravimas:
- Kiln – sintetinės duomenų generavimo aplinkai (topic tree + curation UX)
- OpenRouter SDK – API valdymas Gemini / GPT-4 modeliams
- FastText ir Lingua – kalbos identifikavimui, naudojant confidence > 0.95
- KenLM – naudotas tik pradinei perplexity analizei; pašalintas iš vėlesnio pipeline dėl priklausomybių valdymo, tačiau galėtų būti integruojamas alternatyviai
- MinHash (Datasketch) ir SHA-1 – semantinių bei tikslių dublikatų identifikavimui ir šalinimui

Mokymui ir eksperimentų vykdymui:
- TogetherAI – pagrindinis treniravimo backend su LoRA palaikymu (paleistas per API)
- BitsAndBytes – FP16 / BF16 ir kvantizuoto svorio palaikymui, jei reikia
- Accelerate – treniravimo paskirstymui ir efektyvesniam resursų naudojimui
- HuggingFace Hub – modelių svorių publikavimui ir versijavimui

Vertinimui ir rezultatų sekimui:
- RunPod (Eval mode) – inferencijos job’ų vykdymas H100 aplinkoje
- Evaluate (HF) – metrikų skaičiavimui: TruthfulQA, MMLU, ARC-LT, GSM8K, ir kt.
- Weights & Biases – treniravimo eigai, anomalijų aptikimui ir rezultatų dashboard’ams

Duomenų apdorojimo įrankiai:
- Pandas ir NumPy – struktūrizuotų duomenų analizavimui
- Scikit-learn – vektorizacijai, paprastoms ML procedūroms, filtravimui
- Matplotlib – duomenų ir modelio rezultatų vizualizacijai

3. Eksperimentinis LLM treniravimo įvertinimas ir rezultatai

3.1. Duomenų apdorojimas

Norint sėkmingai pritaikyti kalbos modelius lietuvių kalbos užduotims, vien sujungti žalius duomenis iš įvairių šaltinių (pvz.: Aya, mC, CulturaX) nepakanka. Dalis tekstų gali būti neautentiškai lietuviški arba prastos kokybės (pvz., daugiakalbiai, kupini triukšmo). Be to, potencialiai milžiniški duomenų kiekiai reikalauja sistemingos filtravimo ir rengimo strategijos, užtikrinančios, kad galutinis rinkinys reprezentuotų kokybišką, suderintą ir kalbiškai taisyklingą turinį. Šiame skyriuje aprašomi pagrindiniai duomenų apdorojimo etapai.

3.1.1. Duomenų filtravimas

Vienas iš pirmųjų žingsnių, taikytų šiam projektui, yra perpleksiškumo (toliau – PPL) filtravimas, leidžiantis įvertinti, kiek duotasis tekstas atitinka mokomąją kalbos struktūrą.

3.1.2. Kalbos modelių kūrimas (KenLM)

Parengti du 5-gram lietuvių kalbos modeliai (pvz.: LRT ir Vikipedijos tekstais pagrįsti), naudojant KenLM biblioteką.

Šie n-gram modeliai leidžia apytikriai apskaičiuoti sakinio tikimybę bei sukuria PPL rodiklį, kuris rodo, kaip „lauktinas“ ar „neįprastas“ tekstas yra Lietuvos kalbinei terpei.

3.1.3. PPL slenkstinė reikšmė

Apibrėžus tam tikrą perpleksiškumo slenkstį (pvz., 500, 1000 ar kitą empiriniu būdu nustatytą vertę), dokumentai, kurių PPL gerokai viršija šią ribą, atmetami kaip galimai netinkantys lietuvių kalbai (pvz., smarkiai sumaišyta kalba, automatiškas vertimas, įvairios rašybos klaidos).

Tekstai, atitinkantys priimtiną PPL, žymimi kaip „tikėtina lietuviški“ ir perkeliami į kitus tikrinimo etapus.

3.1.4. Platus duomenų pritaikomumas

Nors PPL metodas negali aptikti visų nekokybiškų ar maišytų kalbų atvejų, jis efektyviai išfiltruoja labiausiai išsiskiriančius triukšmingus dokumentus.

Galutiniame mokymo rinkinyje lieka daugiau kalbiškai nuoseklių pavyzdžių, kas itin svarbu modelio treniravimo stabilumui.

3.1.5. Kalbos atpažinimas

Kadangi mC4 ir CulturaX tekstuose gali būti netipiško turinio (tarptautinių svetainių duomenų, vertimų ar kitų kalbų priemaišų), reikalingas papildomas kalbos atpažinimo žingsnis.

3.1.6. Lingua biblioteka

Naudojama, pavyzdžiui, lingua ar kita panaši kalbos detekcijos biblioteka, leidžianti su geru tikslumu atskirti, ar dokumentas yra parašytas lietuviškai, ar pramaišytas kitomis kalbomis.

Tekstai, kuriuose lietuvių kalba nustatoma tik kaip antraeilis variantas (arba jos pasitikėjimo lygis žemesnis už nustatytą slenkstį), pašalinami iš mokymo bazės.

3.1.7. Ne lietuvių kalbos turinys

Aptikus kitų kalbų segmentus (anglų, rusų, lenkų ar kt.), tekstas gali būti atmestas arba toliau vertinamas atsižvelgiant į duomenų strategiją (pavyzdžiui, kartais trumpi skolinti segmentai gali būti toleruotini, tačiau didelė jų apimtis – ne).

Taip išvengiama situacijų, kai modelis klaidingai „įsimena“ kitų kalbų struktūras kaip lietuviškas.

3.1.8. Kombinuotas panaudojimas

Praktikoje kalbos atpažinimas dažnai derinamas su PPL filtravimu. Gali būti, kad tekstas turės priimtiną PPL bet bus identifikuotas kaip mišrios kalbos; arba atvirkščiai – atrodys lietuviškas, tačiau pasirodys esąs neįprastai didelio PPL. Tokie neatitikimai sukelia papildomas tikrinimo procedūras.

3.1.9. Netinkamo turinio ir dublikatų pašalinimas

Kad galutinis mokymo rinkinys reprezentuotų įvairialypę, bet ne kenksmingą (pvz., kupiną necenzūrinės leksikos) lietuvių kalbos terpę, atliekami keli papildomi filtravimo žingsniai.

3.1.10. Žalingo turinio filtras

Pasitelkiant išankstinį nepageidaujamų žodžių sąrašą, atmetami įrašai, kuriuose gausu keiksmažodžių, įžeidžiančių frazių ar kito nepriimtino turinio.

Šis veiksmas užtikrina, kad modelis nebūtų linkęs generuoti neetiško ar vulgaraus teksto.

3.1.11. Dublikatų aptikimas

Tekstai, kurie yra identiški ar beveik identiški (pavyzdžiui, naujienų pranešimai, rodomi keliose svetainėse be esminių pakeitimų), pašalinami, kad per daug nereprezentuotų tų pačių frazių modeliui.

Dublikatams identifikuoti gali būti naudojami maišymo, MD5, ar net paprasta panašumą matuojanti heuristika.

3.1.12. Pernelyg trumpų (arba nepilnaverčių) įrašų pašalinimas

Įrašai, kuriuos sudaro tik keli žodžiai, antraštės be turinio ar neaiškios atkarpos, taip pat dažniausiai išmetami. Toks šalinimas neleidžia modeliui „mokytis“ beprasmių fragmentų ir gerina bendrą duomenų kokybę.

3.1.13. Galutinių mokymo duomenų formavimas

Atlikus visus filtravimo etapus (PPL, kalbos atpažinimą, blogų žodžių ir dublikatų šalinimą), suformuojamas atitinkamai mokymo, validacijos ir testavimo rinkinys. Šis skirstymas yra kritiškai svarbus:

Mokymo rinkinys:

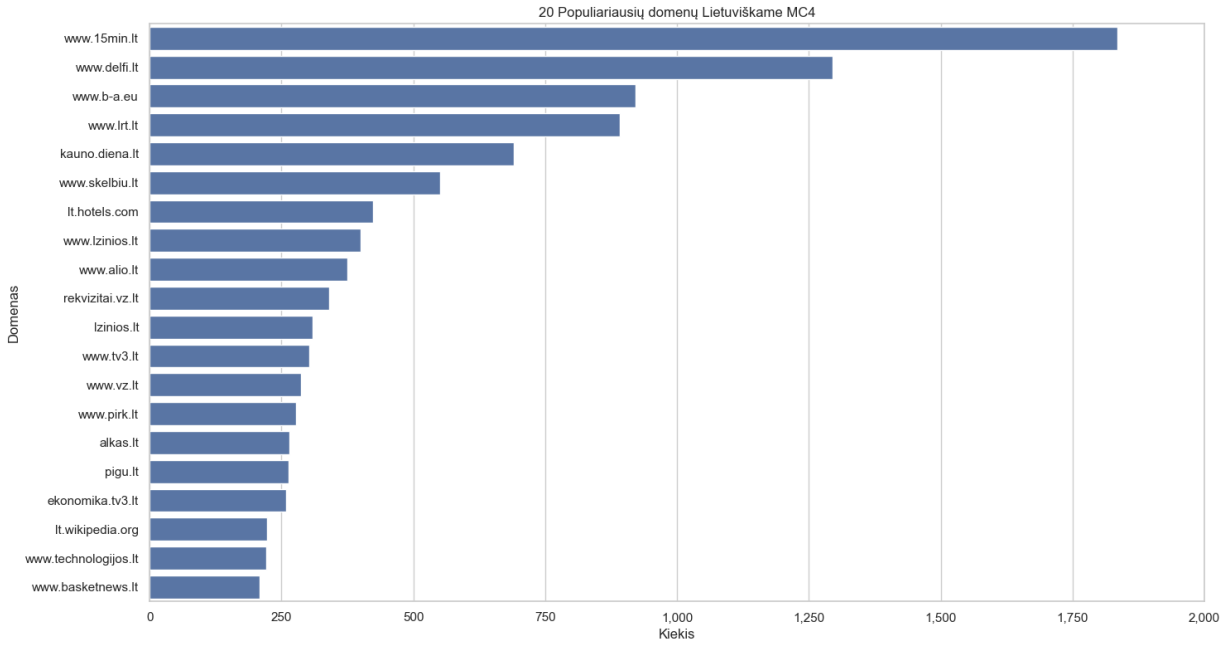
- Didžiausias duomenų kiekis, skirtas modelio parametrams optimizuoti.

Validacijos rinkinys:

- Skirtas hiperparametrams derinti bei ankstyvam pritaikymo kokybės įvertinimui.
- Testavimo rinkinys:
- Atidedamas galutiniam patikrinimui, siekiant nešališko modelio veikimo vertinimo.
- Be to, gali būti pritaikytas viešai prieinamiems testų rinkiniams (Truthful QA, ARC, MMLU, HellaSwag ir t. t.), jei siekiama tarptautinio palyginamumo.

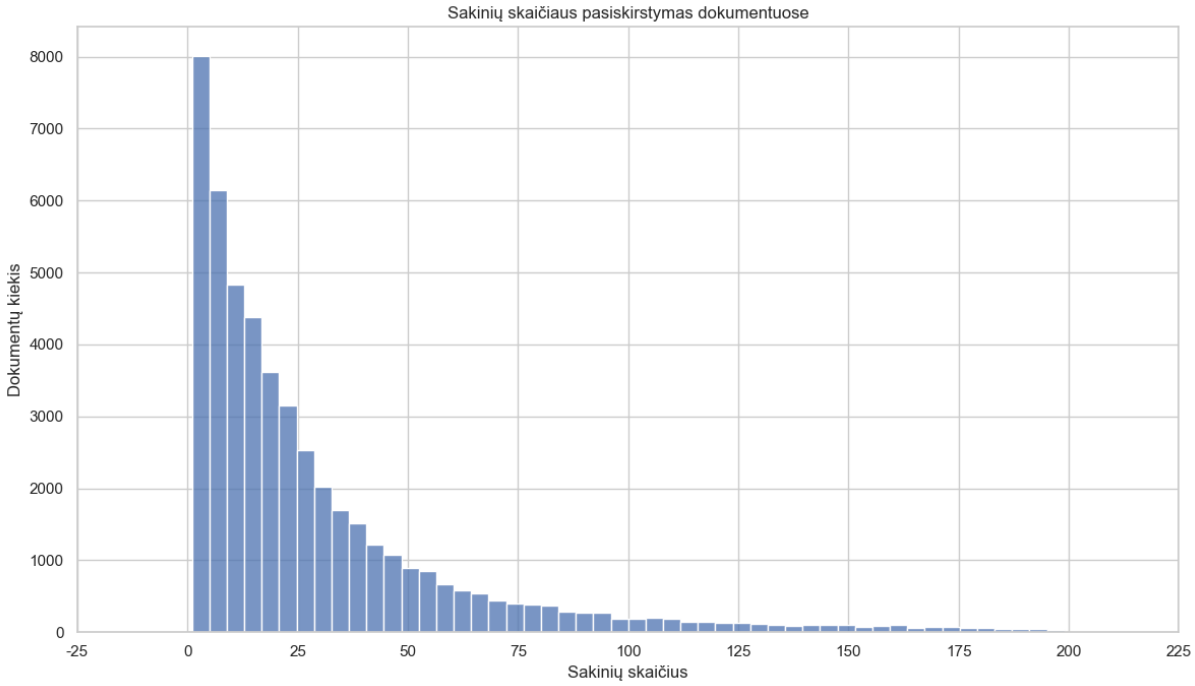
3.1.14. Lietuviško mC4 duomenų rinkinio analizė

Duomenų rinkinio pavyzdžiai:



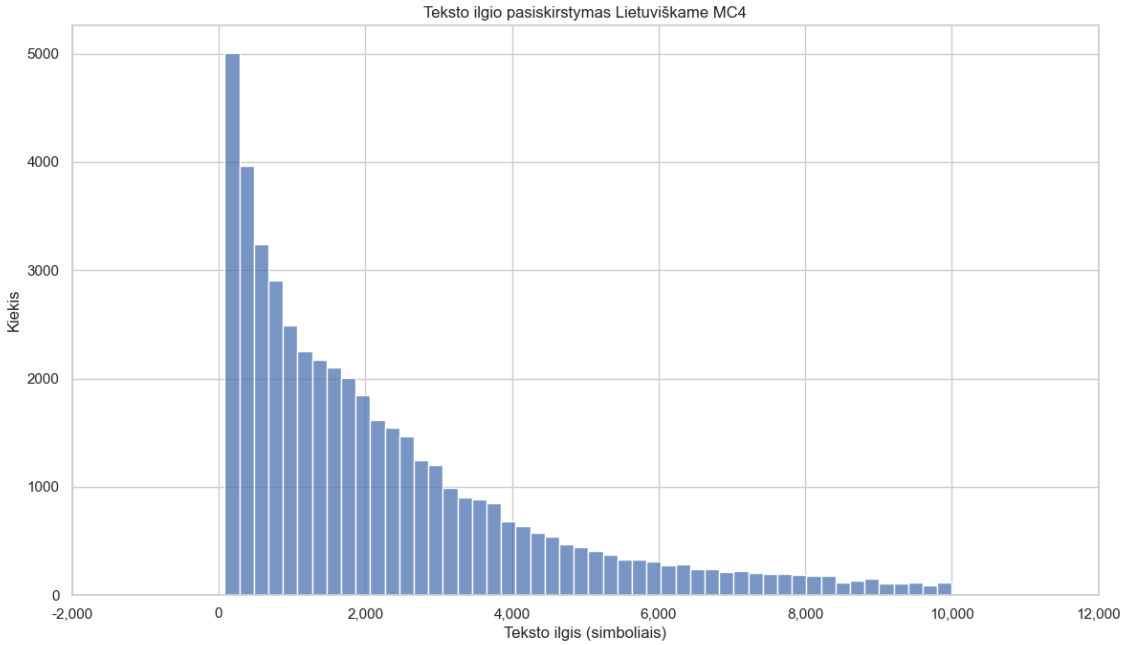
3.1 pav. 20 populiariausių domenų Lietuvos mC4

Grafikas rodo 20 dažniausiai pasitaikančių domenų mC4 duomenų rinkinyje. Pirmauja „15min.lt“, „delfi.lt“ ir „b-a.eu“, kurie generuoja didžiausią dokumentų kiekį.



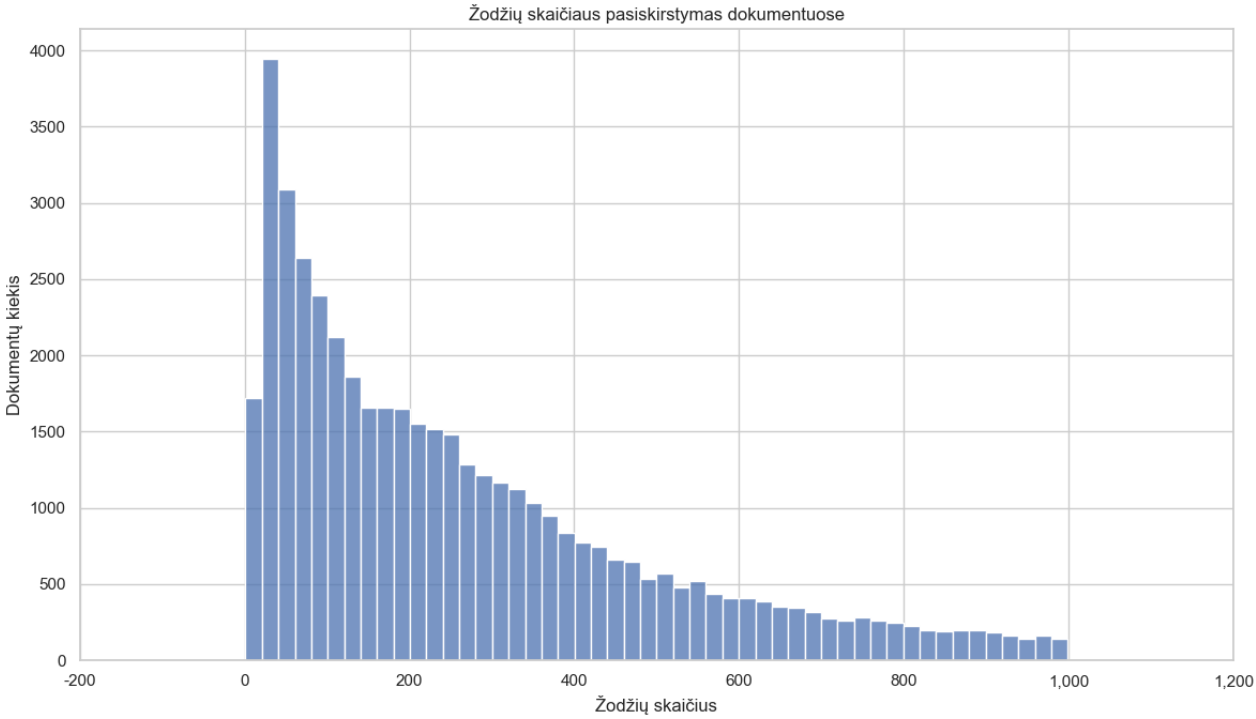
3.2 pav. mC4 Sakinių skaičiaus pasiskirstymas dokumentuose

Šiame grafike pavaizduotas sakinių skaičiaus pasiskirstymas dokumentuose. Daugelyje dokumentų yra iki 25 sakinių, o ilgesni dokumentai pasitaiko žymiai rečiau.



3.3 pav. Teksto ilgio pasiskirstymas

Grafikas atspindi tekstų ilgio pasiskirstymą Lietuvos mC4 duomenų rinkinyje. Dauguma tekstų yra gana trumpi (iki 2000 simbolių), tačiau kai kurie iš jų pasiekia iki 12 000 simbolių.



3.4 pav. Žodžių skaičiaus pasiskirstymas dokumentuose mC4 duomenų rinkinyje

Šiame grafike parodytas žodžių skaičiaus pasiskirstymas dokumentuose. Matyti, kad dauguma dokumentų turi iki 200 žodžių, o dokumentų skaičius mažėja su didėjančiu žodžių skaičiumi.

3.2. Dirbtinio (sintetinio) lietuviško duomenų korpuso generavimas

Po neapdorotų duomenų filtravimo buvo nuspręsta papildyti rinkinį – ir kartu sumažinti jo priklausomybę nuo realių tekstų šaltinių – dirbtinai sugeneruotais lietuviškais pavyzdžiais. Tam pasitelkiau Kiln – atvirojo kodo įrankį, siūlantį interaktyvią sintetinės duomenų generavimo aplinką ir glaudžiai integruotą tiesioginį eksportą į JSONL formatą. Kiln išsiskiria tuo, kad leidžia vienu paspaudimu paleisti *topic-tree* generavimo režimą, kur LLM modeliai automatiškai sukuria temų hierarchiją ir kiekvienai temai parengia pavyzdžius ir naudotojui belieka juos patvirtinti arba atmesti. Taip pat, patvirtinus Kiln gali sukurti tvarkingą ir paruoštą SFT treniravimo duomenų rinkinį.

3.2.1. Užduoties konfigūravimas Kiln aplinkoje

- Sukurta duomenų kūrimo užduotis aplinkoje, nurodytą kurti lietuviškus tekstus. Nurodyta modelį generuoti įvairiapusį, šiuolaikine lietuvių kalba parašytą tekstą (dialogai, naujienų santraukos, techniniai straipsniai, kultūra ir t. t.).
- Pasirinktas Gemini modelis(dėl geros lietuvių kalbos kokybės ir prieinamumo). Modelis buvo kviečiamas per Kiln aplinką, todėl papildomo kodo nereikėjo.
- Kiln automatiškai sugeneravo 180 temų (iki trijų medžio gylio lygių). Vėliau rankiniu būdu pridėta 12 trūkstamų nišinių (pvz., „žaidimų lokalizacija“, „Lietuvos teisė“) tematikų.
- Papildomai nurodyta instrukcija duomenų rinkinio kūrimui: „Venk angliškų žodžių, nebent tai terminai be lietuviško atitikmens; užtikrink, kad visos pavardės būtų sulietuvintos linksniuojant.“ Šis parametras ženkliai sumažino vėlesnį netaisyklingų formų šalinimą.
- Kiekvienam lapiniam mazgui sugeneruota po 20 pavyzdžių, visas ciklas kartotas tris kartus, su tarpine rankine peržiūra („Interactive Curation UX“) po kiekvieno ciklo.

2 lentelė. Duomenų surinkimo ir filtravimo eiga

Žingsnis	Tikslas	Rezultatas
G1	Pirma generavimas (≈ 11 000 įrašų)	Pirminis rinkinys su nedidele nefiltruoto anglų teksto priemaiša
G2	Rankinis valymas	Pašalinta 1,2 % netipinių įrašų, sujungtos dublikato temos
G3	Antras generavimas (≈ 4 000 įrašų)	Užpildytos temų spragos, subalansuotas ilgis
F1	Automatiniai filtrai (*FastText* + *Lingua*)	121 ne lietuviška eilutė pažymėta ir pašalinta
F2	Dedup-SHA-1	30 visiškų dublikatų pašalinta
F3	Eilučių su netinkamais žetonais skaičius	Išskirtinių eilučių nenustatyta

3 lentelė. Galutinio korpuso apimtis

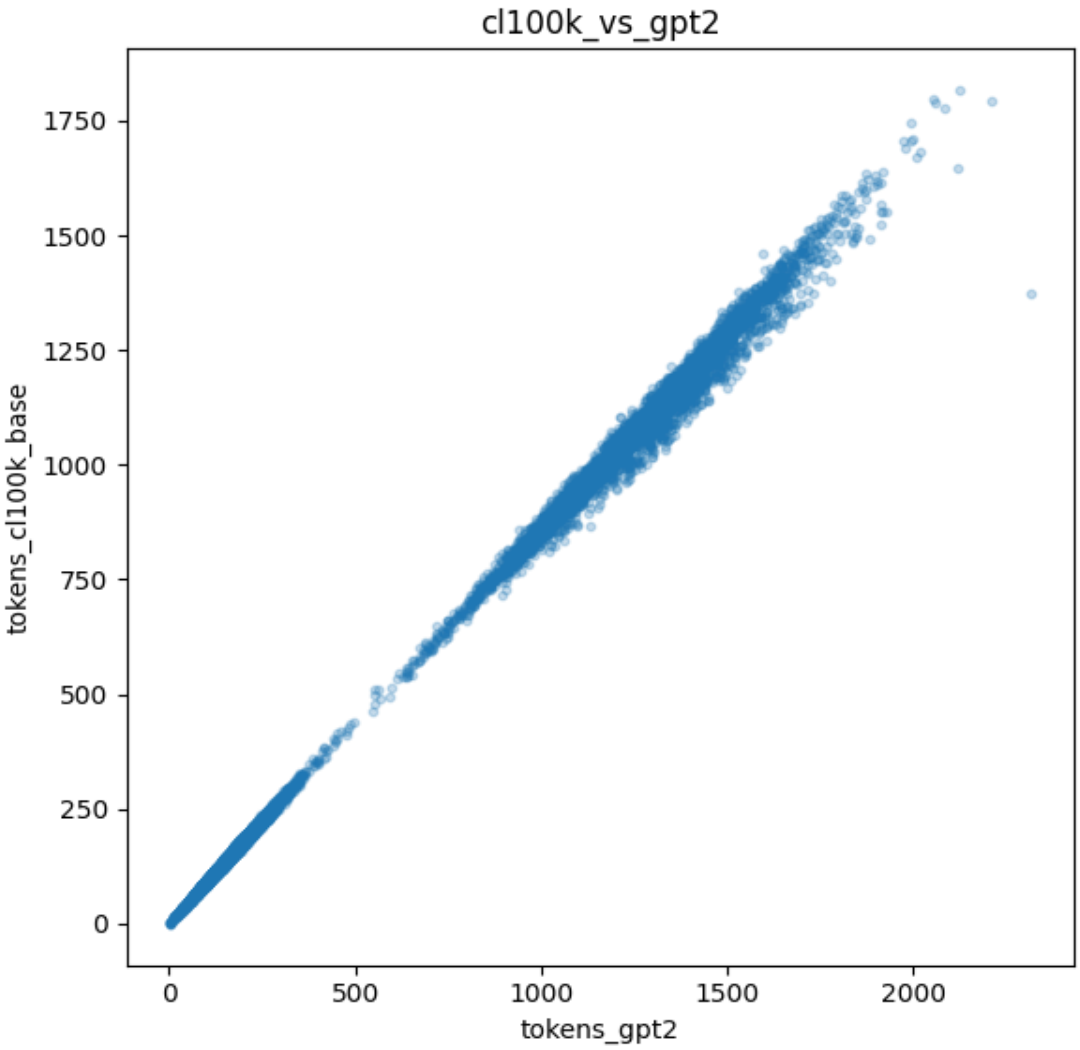
Metrika (cl100k_base)	Reikšmė
Eilučių skaičius	14 810
Vidutinis eil. ilgis (žetonais)	521
Bendras žetonų skaičius	7 719 737
Naudota mokymui	≈ 2 000 000 ž. (atsitiktinai atrinkta po balansavimo)

Atranka į 2 000 000 žetonų padaryta siekiant sumažinti GPU mokymo laiką ir atitikti skaičiavimo taisyklėms (žetonų skaičius stipriai priklausė nuo modelio originalaus tokenizatoriaus).

3.2.2. Kokybės metrika

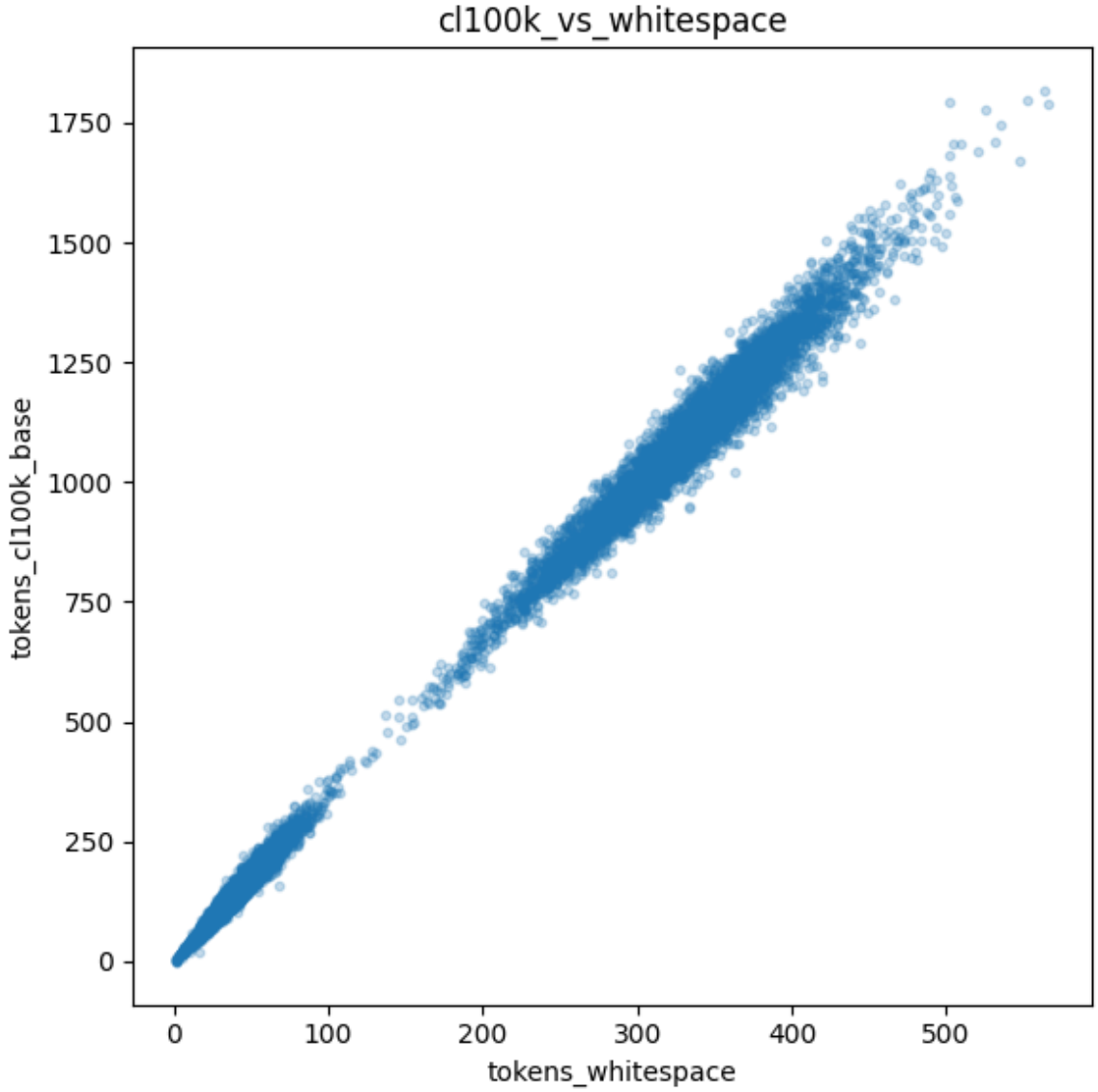
- Dublikatai: 0,2 %
- Ne lietuvių turinys: 0,8 %
- Išskirtinumai: 0 %
- PPL mediana: 190 (kenLM-LRT modelis)

Visa analizė automatiškai eksportuota į vieną duomenų failą, o probleminiai pavyzdžiai į atskirus ir naudojami tolesniame validacijos žingsnyje.



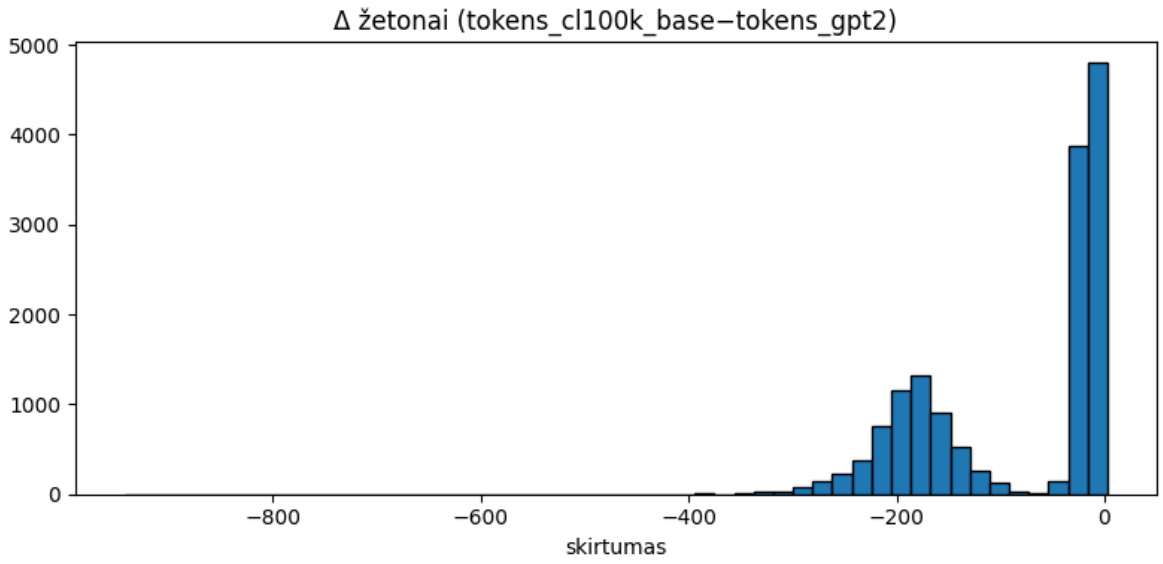
3.5 pav. cl100k_base vs gpt2 žetonų skaičius

Šiame sklaidos grafike pavaizduotas žetonų kiekio palyginimas tarp GPT-2 (x ašis) ir cl100k_base (y ašis) tokenizatorių, taikytų kiekvienai iš 14 810 sintetinių eilučių. Diagrama aiškiai rodo stiprią tiesinę koreliaciją tarp abiejų metodų: beveik visi taškai išsidėstę aplink pagrindinę įstrižainę. Tai reiškia, kad GPT-2 ir cl100k_base daugeliu atvejų pateikia labai panašų žetonų skaičių. Vis dėlto, viršutinėje dalyje pastebimas nežymus GPT-2 šuolis: kai kurios eilutės turi iki 20 % daugiau žetonų nei cl100k_base. Tai atitinka ir bendrą statistinį santykį – GPT-2 vidutiniškai sukuria 1.16 karto daugiau žetonų nei cl100k_base. Toks skirtumas reikšmingas optimizuojant mokymo kaštus – pasirinkus ekonomiškesnį cl100k_base, galima sutaupyti GPU atminties be reikšmingos informacijos praradimo.



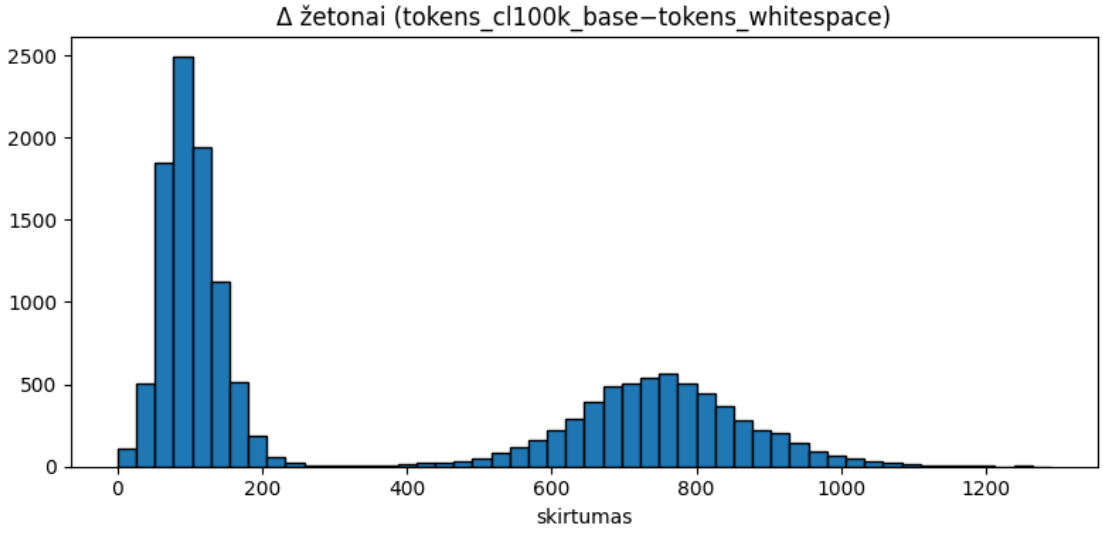
3.6 pav. cl100k_base vs whitespace žetonų skaičius

Šiame grafike žetonų skaičius gautas paprastu tarpo skaidymu (*whitespace*, x ašis) lyginamas su cl100k_base (y ašis). Ir čia ryškėja stipri koreliacija, tačiau pastebimai platesnė sklaida nei GPT-2 atveju – ypač vidutinio ilgio tekstuose. Tai rodo, kad cl100k_base ženkliai smulkiau skaido tekstą nei klasikinis žodžių skaidymas: pvz., 300 „žodžių“ *whitespace* tokenizatoriuje dažnai atitinka 800+ žetonų cl100k_base. Tokia disproporcija ypač išryškėja sintetinėse pastraipose su gausia morfologine struktūra (linksniai, priesagos). Tai įrodo, kad *whitespace* nėra pakankamai tikslus žetonų matas – nors tinkamas kaip žaliavos analizės įrankis, realiam modelio dydžiui įvertinti jis per daug supaprastintas



3.7 pav. Žetonų skirtumas tarp cl100k_base ir GPT-2

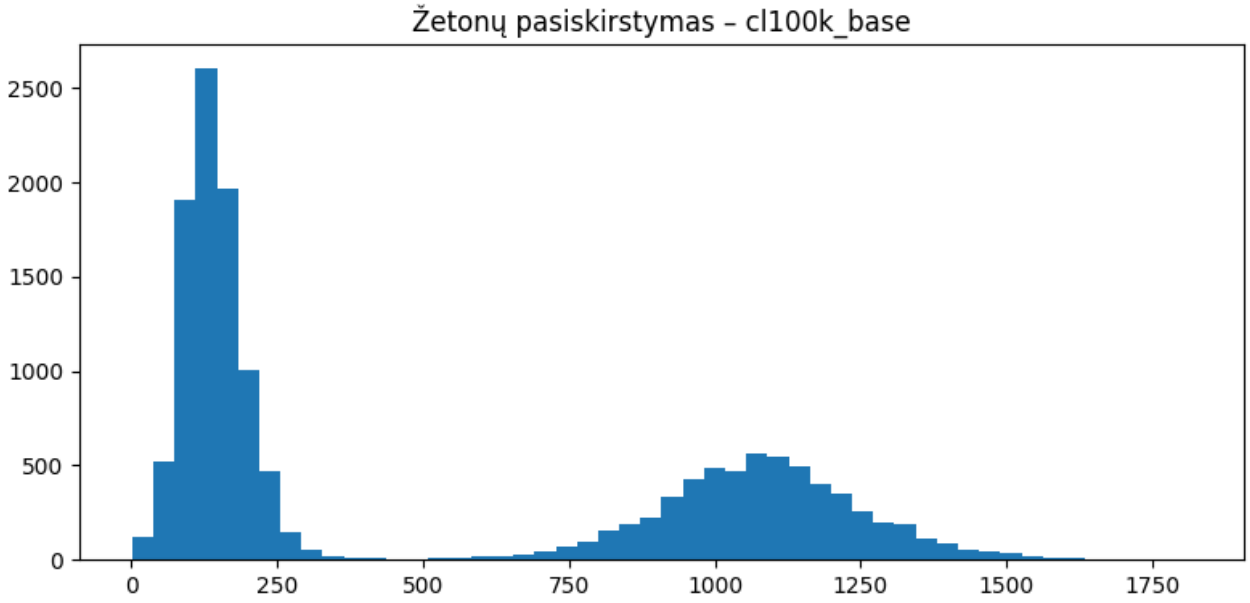
Ši delta histograma rodo skirtumų pasiskirstymą tarp cl100k_base ir GPT-2 tokenizatorių kiekvienai duomenų eilutei. Neigiamų reikšmių pasiskirstymas rodo, kad GPT-2 dažniausiai generuoja daugiau žetonų nei cl100k_base – skirtumas dažniausiai svyruoja nuo -100 iki -300 žetonų, su piku apie -150. Tai atitinka ir anksčiau minėtą vidutinį 16 % skirtumą. Šis rezultatų pasiskirstymas labai glaudus – duomenų beveik nėra dešinėje pusėje (t. y., atvejų, kai cl100k_base viršija GPT-2). Todėl galima teigti, kad cl100k_base yra ekonomiškesnis pasirinkimas, jei siekiama sumažinti žetonų skaičių be reikšmingo turinio suskaidymo praradimo.



3.8 pav. Žetonų skirtumas tarp cl100k_base ir *whitespace* tokenizavimo

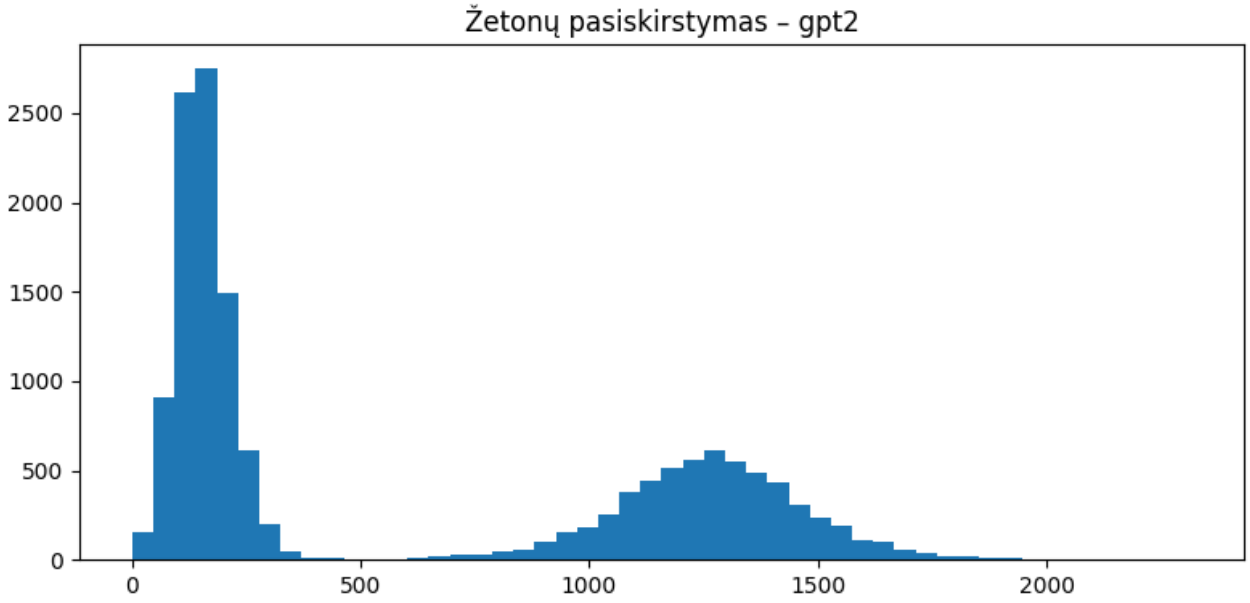
Ši histograma rodo žymiai platesnį ir dvimodalį pasiskirstymą. Dauguma skirtumų yra teigiami, o tai reiškia, kad cl100k_base beveik visada sukuria daugiau žetonų nei tarpai pagrįstas skaidymas. Pirmasis pasiskirstymo pikas (apie +150 žetonų) atitinka trumpesnius tekstus, o antrasis – apie +800 žetonų – ilgus sakinius su sudėtinga morfologija ir daug prielinksnių, galūnių. Šis rezultatas išryškina, kiek smulkesnį segmentavimą atlieka šiuolaikiniai BPE pagrindo tokenizatoriai (pvz., cl100k_base),

lyginant su paprastu žodžių skaidymu. Dėl šios priežasties tarpas (whitespace) gali būti naudojamas tik apytiksliam žodžių skaičiui įvertinti, bet ne realiam modelio žetonų suvartojimui apskaičiuoti.



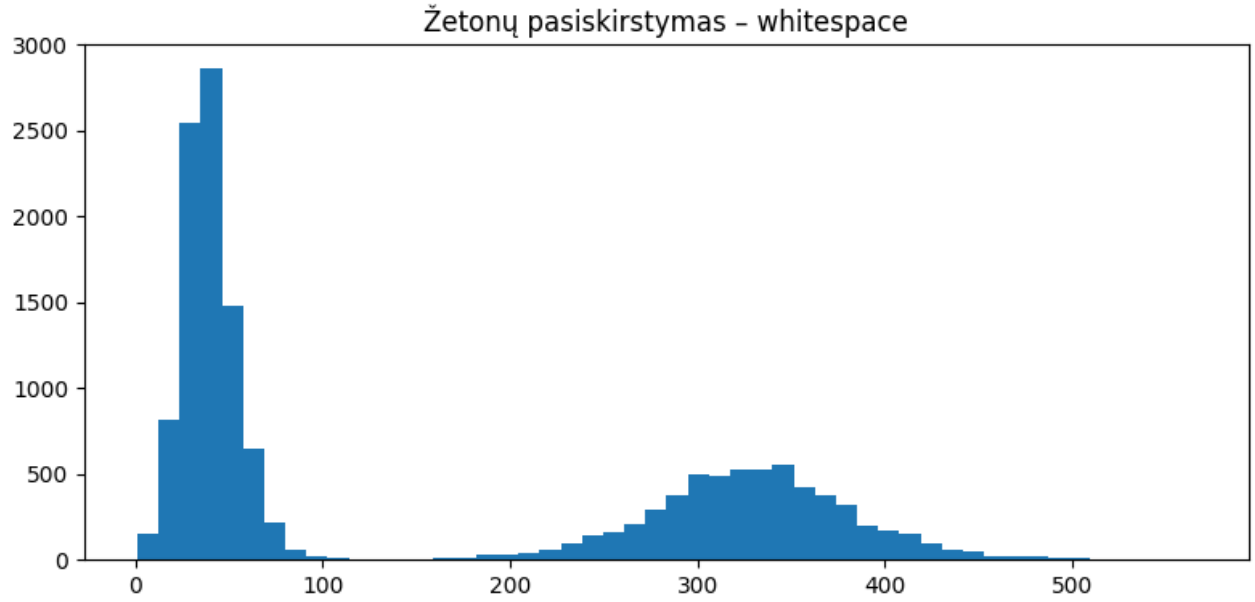
3.9 pav. Žetonų pasiskirstymas naudojant cl100k_base.

Ši histograma vaizduoja, kaip pasiskirsto žetonų skaičius sintetinėje lietuviškoje duomenų aibėje, taikant OpenAI cl100k_base tokenizatorių. Akivaizdžiai matomi du pagrindiniai pasiskirstymo „kalnai“ (bimodalumas): pirmasis – apie 150–250 žetonų, antrasis – apie 1000–1200 žetonų. Šis bimodalumas rodo, kad generuoti tekstai pasiskirstę tarp trumpų (pvz., apibrėžimai, antraštės, citatos) ir ilgų (išplėstiniai aprašymai, straipsnių ištraukos, pasakojimai) įrašų. Tokio pobūdžio pasiskirstymas leidžia efektyviau išmokyti modelį tiek glaustų, tiek išsamių konstrukcijų.



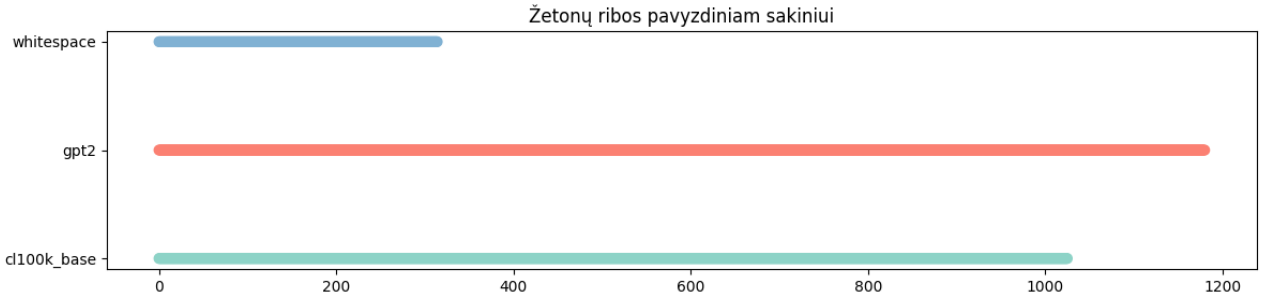
3.10 pav. Žetonų pasiskirstymas naudojant GPT-2 tokenizatorių.

GPT-2 tokenizatoriaus atveju pastebima labai panaši bimodalinė struktūra kaip ir 15 pav., tačiau antrasis kalnas yra platesnis ir kiek labiau pasislinkęs į dešinę. Tai patvirtina, kad GPT-2 vidutiniškai išskaido tekstus į daugiau žetonų nei cl100k_base – tai atitinka ir skaitinius rodiklius (gpt2 vidurkis: 606 žetonai, cl100k_base: 521 žetonas). Tokie skirtumai daro įtaką praktiniam treniravimo procesui: jei modelis būtų treniruojamas GPT-2 pagrindu, būtų reikalingi didesni atminties resursai. Visgi, abi histogramų formos rodo, kad generuoti duomenys buvo pakankamai įvairūs ir neturėjo nenatūralių pailgėjimų ar trumpinimų.



3.11 pav. Žetonų pasiskirstymas naudojant whitespace tokenizaciją

Skirtingai nuo cl100k_base ar GPT-2 tokenizatorių, ši histograma atspindi žetonų skaičių, gautą paprastu žodžių atskyrimu pagal tarpus. Nors taip gauti žetonai nėra tinkami modelio treniravimui, jie naudingi kaip orientacinis tekstų ilgio matas. Čia taip pat ryškus bimodalus pasiskirstymas: pirmasis pikas – apie 30–60 žodžių, antrasis – apie 250–400 žodžių. Tai atitinka duomenų logiką – trumpi ir ilgi pavyzdžiai generuoti tolygiai, siekiant lavinti modelį tiek trumpiems, tiek išsamiems kontekstams. Tačiau lyginant su kitomis tokenizacijomis, čia bendras žetonų skaičius yra kelis kartus mažesnis – tai iliustruoja ribotą praktinę *whitespace* metodo vertę treniruojant modelius.



3.12 pav. Žetonų ribų palyginimas tame pačiame sakinyje

Pateiktas palyginimas tarp trijų tokenizavimo metodų, taikytų tam pačiam sakiniui: vizualiai parodyta, kiek skirtingų žetonų generuoja kiekvienas metodas ir kaip šios ribos išsidėsto.

- *Whitespace* (mėlyna), kur mažiausiai žetonų, kiekvienas atitinka vieną „žodį“.

- GPT-2 (raudona), kur tankus skaidymas – beveik dvigubai daugiau žetonų nei whitespace.
- cl100k_base (žalia), kuri panaši į GPT-2, tačiau šiek tiek kompaktiškesnė, ypač lietuvių kalboje.

3.2.3. Duomenų eksportavimas ir tolesnis naudojimas

- Kiln eksportavo pasirinktą duomenų pogrupį į JSONL failą.
- Eksportas įkeltas į mokymosi proceso eigą.

Proceso eiga:

- Pirmiausia *token-shuffle* skriptas atsitiktinai perdėlioja eilučių seką,
- tada – 80 / 20 % paskirstymas (treniravimas ir tikrinimas).
- Galutinis treniravimo failas sudaro 2 000 000 žetonų ± 2 %, o tikrinimas apie 100 000 žetonų.

Šiuo būdu parengtas korpusas SFT darbams naudoti.

3.3. Atrinktų modelių treniravimas

Naudota sintetiniai duomenys ir jų mišinys kartu su perdirbtais duomenų rinkiniais. Llama 3.2 1B ir 3B modeliai buvo naudojami užšaldyti daliai jų svorių ir įdiegti nauji duomeny pasinaudojant LoRA metodu.

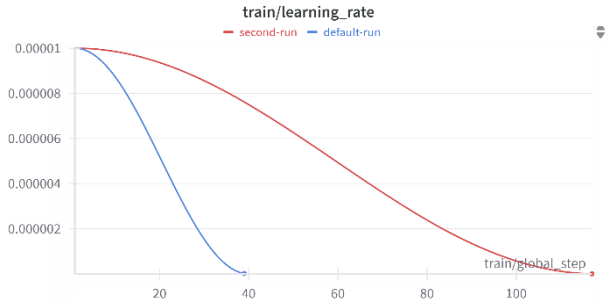
Atsižvelgta į modelių bei sintetinių duomenų ir vadovautasi sudarytu metodologijos dalyje planu.

3.3.1. Llama-3.2-1B-Base modelio treniravimas

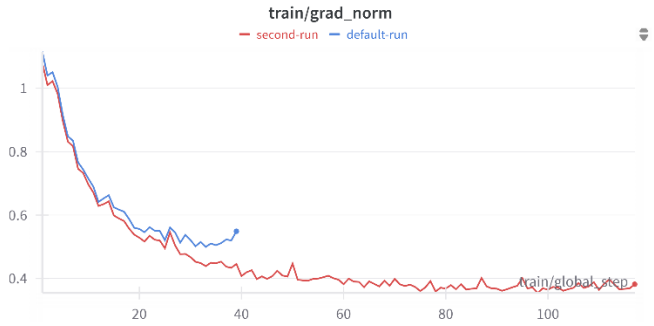
Pradžioje naudojantis jau minėtais metodais, buvo pasirinkta ištreniruoti bazinį (ne pokalbiams skirtą) modelį.

Nagrinėjant treniravimą **(3.13 pav. - 3.15 pav.)** pateikiami bazinio LLaMA 3.2 1B modelio treniravimo metrikų grafikai, atvaizduojantys dviejų skirtingų treniravimo paleidimų (*runs*) eigą: pirmojo (mėlyna kreivė) ir antrojo (*second-run*, raudona kreivė). Abu bandymai buvo vykdomi su tais pačiais hiperparametrais (*LoRA rank* = 64, alpha = 128, *learning rate* = 1e-5 ir kt.), tačiau skyrėsi treniravimo trukme: pirmasis paleidimas buvo vykdytas 1 epochai, o antrasis – 3 epochoms:

3.13 pav. Treniruotės nuostolio mažėjimo palyginimas tarp pirmojo ir antrojo bandymo su Llama 3.2 1B Base modeliu



3.14 pav. Mokymosi greičio kreivės – skirtingų epochų poveikis treniruojant Llama 3.2 1B Base modelį



3.15 pav. Gradientų normos pokytis treniruotės metu treniruojant Llama 3.2 1B Base modelį

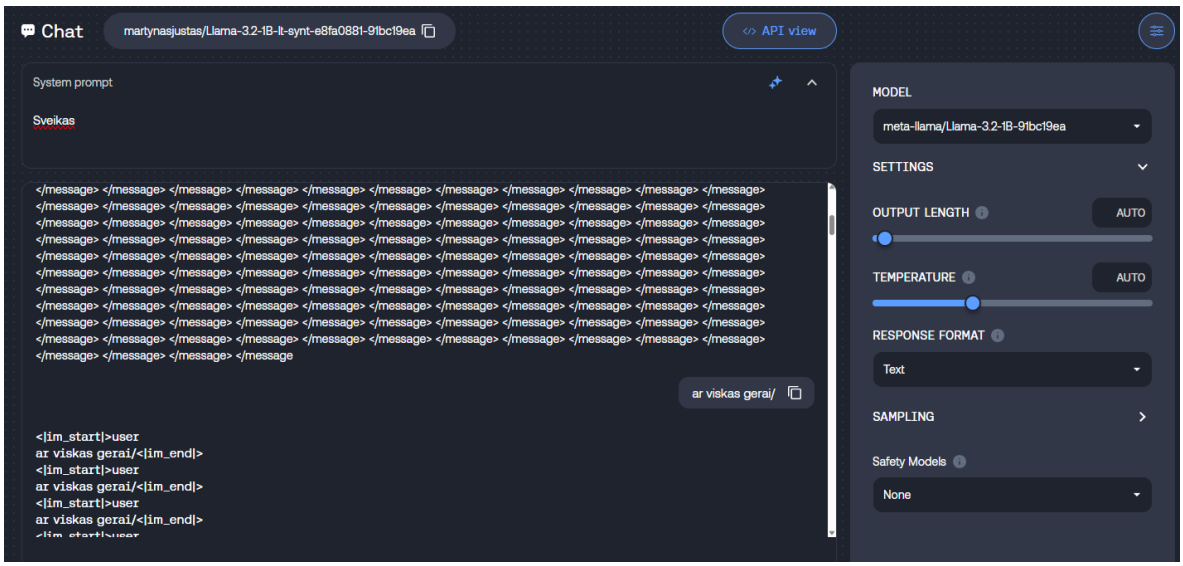
3.13 pav. vaizduoja nuostolio (angl. *loss*) reikšmių mažėjimą treniruotės metu. Matoma, kad *second-run* atveju treniravimo eiga buvo ilgesnė (daugiau nei 110 žingsnių) ir pasiekė žemesnę nuostolio reikšmę (~1.45), tuo tarpu pirmasis bandymas baigėsi ties ~40 žingsnių su aukštesniu *loss* (~1.55). Tai rodo, jog ilgesnis treniravimo laikotarpis leido modeliui labiau prisitaikyti prie duomenų.

3.14 pav. pateikiama mokymosi greičio kreivė. Kadangi naudotas kosinuso mokymosi greičio planuotojas, abu bandymai rodo nuoseklų greičio mažėjimą. Tačiau *second-run* kreivė leidžiasi palaipsniui per visą treniruotės trukmę, tuo tarpu pirmojo bandymo mokymosi greitis ženkliai sumažėjo jau po 30–40 žingsnių, atspindėdamas trumpesnį treniravimo laikotarpį.

3.15 pav. iliustruoja gradientų normos dinamiką – svarbų indikatorių, leidžiantį vertinti treniravimo stabilumą. Abiejų bandymų pradžioje matomas staigus gradientų normos mažėjimas, o vėliau – stabilizacija. Antrojo bandymo metu normos vertė stabiliai laikėsi ties ~0.4–0.5, tuo tarpu pirmojo bandymo metu paskutinėje fazėje pastebimas nedidelis išsibarstymas. Tai leidžia daryti prielaidą, jog ilgesnis treniravimas suteikė didesnį gradientų stabilumą bei patikimesnį svorių konvergavimą.

Ilgesnė treniravimo trukmė (3 epochos vietoje 1) padėjo modeliui pasiekti mažesnį *loss*, tolygiau valdyti mokymosi greitį ir išlaikyti stabilesnę gradientų normą, todėl *second-run* rezultatai laikytini palankesniais tolimesniam modelio tobulinimui.

Treniravimas pavyko sklandžiai, truko apie 10 minučių ir modelis sugebėjo įsisavinti suteiktus žetonus. Tačiau buvo abejonių ar toks modelis gebės įsisavinti instrukcijų sekimo elgseną. Deja bandymo metu (**3.16 pav.**) matoma, kad tai nepavyko.



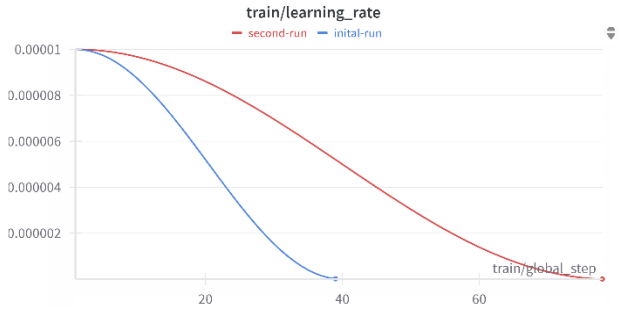
3.16 pav. Bazinio modelio išbandymas naudojantis TogetherAI vartotojo sąsaja

Tai nutinka dėl treniravimo stokos, duomenų stokos, arba modelio dydžio, šiuo atveju modeliai netinkami testavimui dėl savo haliucinacijų ir neišmokimo sekti pokalbio roboto taisyklių, tokių kurios egzistuoja SFT duomenų rinkiniuose.

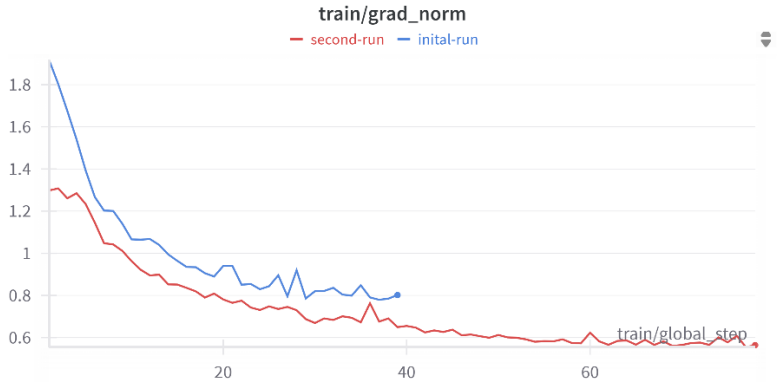
3.3.2. Llama-3.2-1B-Instruct modelio treniravimas

Treniravimo parametrai išliko tie patys kaip bazinio modelio atveju, o tai buvo *LoRA rank* = 64, alpha = 128, *learning rate* = 1e-5, *batch size* = 32, *cosine scheduler* ir 1 arba 3 epochos, žemiau pateikiu glaustą, bet aiškų tekstą po visais LLaMA-3.2-1B-Instruct modelio treniravimo grafikais (**3.17 pav.; 3.18 pav.**; **3.19 pav.**):

3.17 pav. Treniruotės nuostolio mažėjimo palyginimas tarp pirmojo ir antrojo bandymo su Llama-3.2-1B-Instruct modeliu



3.18 pav. Mokymosi greičio kreivės – skirtingų epochų poveikis treniruojant Llama-3.2-1B-Instruct modelį



3.19 pav. Gradientų normos pokytis treniruotės metu treniruojant Llama-3.2-1B-Instruct modelį

Šį kartą pirmasis bandymas truko 5 minutes, o antrasis dvigubai ilgiau 10 minučių. Treniravimo metu abiejų LLaMA-3.2-1B-Instruct modelio bandymų nuostolio reikšmės nuosekliai mažėjo, tačiau ilgesnis *second-run* pasiekė žemesnę galutinę reikšmę (~2.1), kas rodo efektyvesnį prisitaikymą prie instrukcijų tipo duomenų. Mokymosi greičio kreivė atskleidžia skirtumą tarp treniravimo trukmės: trumpesnis bandymas greitai pasiekė nulį, o ilgesnis palaipsniui mažino greitį per visą trajektoriją, taip išnaudodamas visas tris epochas.

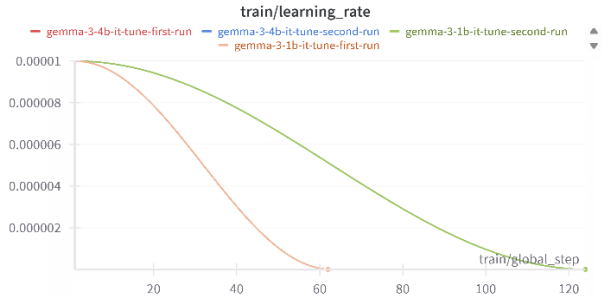
Gradientų normos dinamika rodo, kad pradžioje abiejų bandymų reikšmės buvo gana aukštos, tačiau ilgiau treniruotas modelis pasiekė stabilesnę normą be reikšmingų šuolių pabaigoje. Tai rodo sklandesnį ir patikimesnį modelio svorių prisitaikymą ilgesnės treniruotės metu.

3.3.3. Gemma-3-1b-it ir gemma-3-4b-it modelių treniravimas

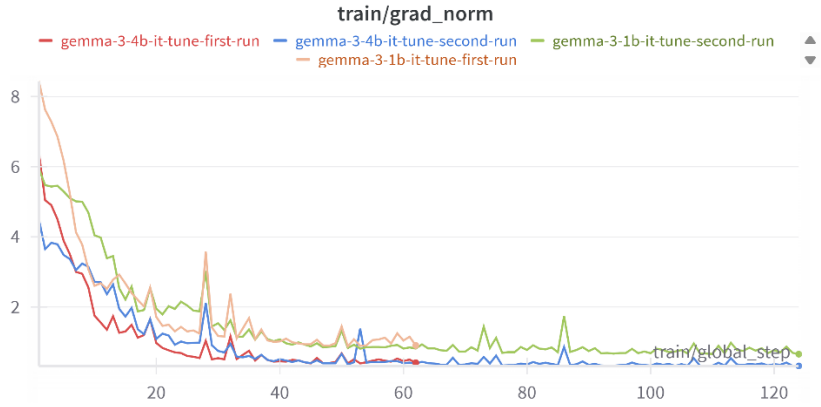
Taip pat, lyginami dviejų skirtingo dydžio modelių Gemma-3-1B-it ir Gemma-3-4B-it treniravimo rezultatai, naudojant tą patį sintetinį duomenų rinkinį. Buvo atlikti po du bandymus kiekvienam modeliui, siekiant įvertinti treniravimo stabilumą ir optimizacijos efektyvumą.

Pirmojo bandymo metu naudotas LoRA rangas 64 ir *batch size* 8, treniruojant vieną epochą, o antrojo bandymo metu – mažesnis LoRA rangas 32 ir dviejų epochų trukmė. Abu eksperimentai taikė kosinuso mokymosi greičio planavimą be apšilimo fazės. Toliau pateikiamuose grafikuose parodyti nuostolio, mokymosi greičio ir gradientų normos pokyčiai visų treniravimo paleidimų metu (**3.20 pav.; 3.21 pav.; 3.22 pav.)**.

3.20 pav. Treniruotės nuostolio mažėjimo palyginimas tarp Gemma 3 1b ir 4b modelių



3.21 pav. Mokymosi greičio kreivės – skirtingų epochų poveikis treniruojant Gemma 3 1b ir 4b modelius



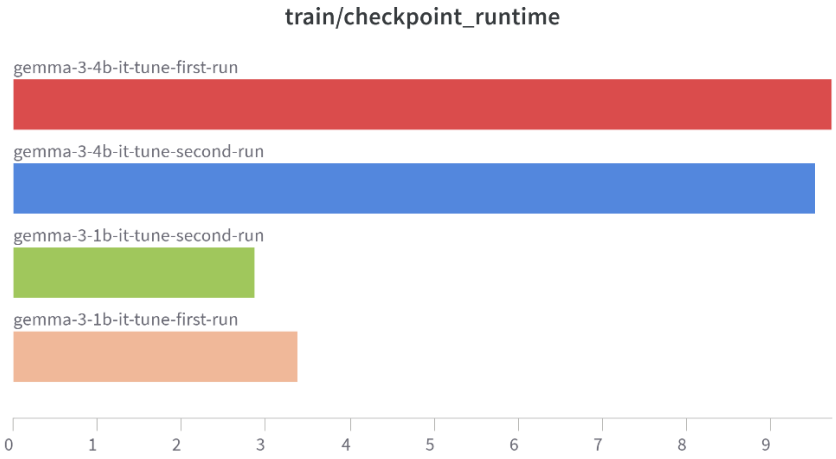
3.22 pav. Gradientų normos pokytis treniruotės metu treniruojant Gemma 3 1b ir 4b modelius

Mažiausią treniravimo nuostolį pasiekė Gemma 3 4B modelio antrasis bandymas (mėlyna linija), kuris per dvi epochas stabiliai konvergavo iki ~1.1. Tuo tarpu Gemma 3 1B antrasis bandymas (žalia kreivė) buvo nestabilus ir rezultatai liko aukštesni, nepaisant ilgesnės treniruotės. Pirmasis 1B bandymas (oranžinė) buvo stabilesnis, bet konvergavo lėčiau.

Mokymosi greičio kreivės visiems paleidimams visiškai persidengia, nes naudotas tas pats *cosine scheduler*. Gradientų normos grafike matyti, kad didžiausi svyravimai pradžioje fiksuoti su 4B modeliu (raudona kreivė), tačiau vėliau normos susistabilizavo ir 4B modelio antras bandymas išsiskyrė kaip tvirčiausias.

Bendrai, 4B modelis, pradžioje sunkesnis, bet ilgainiui pasiekė geresnį rezultatą ir stabilesnį treniravimą nei jo mažesnė versija 1B modelis.

Treniravimo laikas (*checkpoint runtime*) kiekvienam bandymui fiksuotas siekiant įvertinti resursų poreikius skirtingo dydžio modeliams. Žemiau pateiktame grafike (**3.23 pav.**) pavaizduoti visų keturių Gemma modelio treniravimo bandymų trukmės skirtumai.

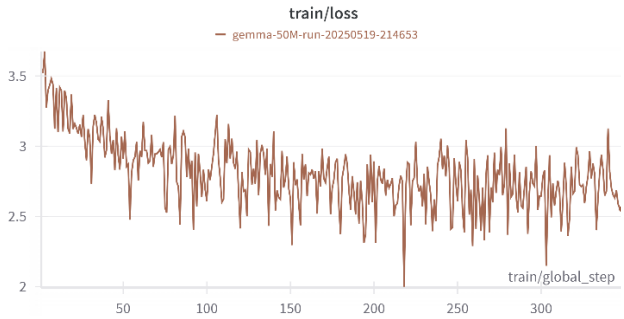


3.23 pav. Gemma 3 1B ir 4B modelių treniravimo trukmės (*checkpoint runtime*) palyginimas tarp

Gemma 3 4B modelio treniravimas užtruko ženkliai ilgiau nei 1B modelio, nepriklausomai nuo epochų skaičiaus. Abu 4B bandymai viršijo 9 min. trukmę, tuo tarpu 1B modelio treniravimas buvo daugiau nei dvigubai trumpesnis.

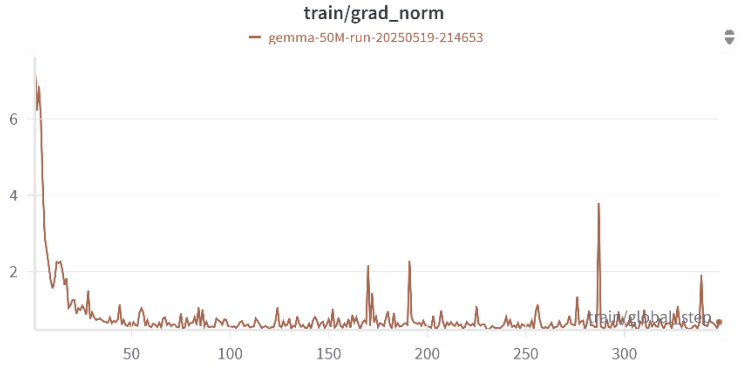
3.3.4. Gemma-3-1b-it modelio treniravimas su hibridiniu duomenų rinkiniu

Paskutinio eksperimento metu **(3.24 pav.; 3.25 pav.; 3.26 pav.**) Gemma-3-1B-it modelis buvo treniruojamas su specialiai parengtu hibridiniu duomenų rinkiniu, kurį sudarė filtruoti ir sujungti generuoti, mC4-LT, Aya bei CulturaX duomenys. Naudota duomenų apimtis siekė ~50 milijonų žetonų, o treniravimas atliktas taikant tą pačią LoRA metodiką bei kosinuso mokymosi greičio planavimą.



3.24 pav. Treniravimo nuostolio dinamika su Gemma-3-1b-it-50M modeliu

3.25 pav. Gemma-3-1b-it-50M **m**okymosi greičio mažėjimas



3.26 pav. Gradientų normos pokytis modelio Gemma-3-1b-it-50M treniravime

Treniravimo nuostolis krito gana nuosekliai, tačiau išliko aukštesnis nei ankstesnių bandymų metu (~2.4 - 2.6), o reikšmingas triukšmas rodo, kad hibridinis rinkinys, nors ir turtingesnis semantiškai, gali būti mažiau vientisas. Gradientų normos grafike matomas aiškus stabilizavimasis po pradinio šuolio – tai reiškia, kad modelis pamažu prisitaikė prie įvairesnių duomenų, tačiau pavieniai pikai gali signalizuoti apie lokalius duomenų neatitikimus ar perėjimus tarp skirtingų stilių.

3.4. Treniruotų LLM lyginimas ir įvertinimas

Šiame etape buvo atliktas kalbos modelių palyginimas, siekiant įvertinti, kaip eksperimentiškai ištreniruoti modeliai (Gemma ir LLaMA versijos) veikia įvairiose lietuviškai adaptuotose vertinimo užduotyse. Testavimas atliktas naudojant standartizuotus testų rinkinius(angl. *benchmark*): MMLU, ARC Easy, TruthfulQA ir kitus, vertinančius bendrąją semantinę orientaciją, pasaulio žinių taikymą, loginį mąstymą bei atsakymų teisingumą.

Lyginimui pasirinkti ir viešai prieinami ir jau minėti modeliai iš Neurotechnology, tai Lt-LLaMA-2-7B ir Lt-LLaMA-2-13B versijos, kurios buvo iš anksto pritaikytos lietuvių kalbai bei testuotos su tais pačiais vertinimo rinkiniais. Lt-LLaMA-2-13B modelis, turintis apie 12.7 milijardo parametrų, pasiekė vidutinį tikslumą 35.23%, o atskirose užduotyse gavo 26.44% MMLU, 54.5% ARC Easy ir 35.23% TruthfulQA. Tuo tarpu mažesnis Lt-LLaMA-2-7B modelis, turintis 6.9 milijardo parametrų, pasiekė vidutinį tikslumą 32.09%, su 26.01% MMLU, 43.18% ARC Easy ir 41.38% TruthfulQA. Abu modeliai buvo testuoti su bfloat16 tikslumo tipu.

Šie rezultatai naudojami kaip atskaitos taškas, su kuriuo palyginami šiame darbe išbandyti ir lietuviškai PEFT pritaikyti Gemma ir LLaMA modeliai.

3.4.1. LLaMA modelių rezultatų analizė

Norint įvertinti LLaMA 3.2 modelių veikimą su lietuvių kalba, buvo testuojami keli variantai: 1B ir 3B modelių versijos, taip pat eksperimentiškai treniruotas su sintetiniais duomenimis LT-SynTune-LLaMA-3.2-1B-Instruct modelis, paruoštas šio tyrimo metu. Testavimas atliktas naudojant kelis vertinimo aspektus: tikslumo metrikas (TruthfulQA MC1/MC2), generavimo kokybės rodiklius (BLEU, ROUGE), bei užduočių sprendimo testus (ARC, MMLU). Toliau pateikiamos lentelės apibendrina šių modelių rezultatus (**3.1 lentelė**; **3.2 lentelė**).

3.1 lentelė. LLaMA 3.2 modelių vertinimas TruthfulQA ir generavimo testuose

Model	truthfulqa_lt_mc1/acc	truthfulqa_lt_mc2/acc	generation/bleu_acc	generation/rouge1_acc	generation/rougeL_acc
LT-SynTune-LLaMA-3.2-1B-Instruct	0.2326	0.4299	0.2705	0.284	0.284
LLaMA-3.2-1B-Instruct	0.2277	0.4295	0.2681	0.2766	0.2705
LLaMA-3.2-3B-Instruct	0.2656	0.4597	0.3231	0.2534	0.2497

LLaMA modelių rezultatus vertinant teisingų atsakymų tikimybę (TruthfulQA) bei generuoto teksto kokybę (BLEU, ROUGE). Kaip matyti, geriausius rezultatus TruthfulQA MC1 ir BLEU testuose pasiekė 3B modelis, tačiau LT-SynTune 1B modelis pranoko tiek bazinį 1B, tiek 3B modelį ROUGE-1 ir ROUGE-L rodikliuose, kas rodo geresnį stilistinį suderinamumą. Tai atspindi fine-tune įtaką atsakymų formuluotei, nors semantinis pranašumas vis dar priklauso nuo modelio dydžio.

3.2 lentelė. LLaMA 3.2 modelių rezultatai loginio mąstymo ir žinių testuose (ARC, MMLU)

Model	ARC (acc)	ARC (acc_norm)	MMLU (avg acc)
just-gud/LT-SynTune-Llama-3.2-1B-Instruct	0.2723	0.2816	0.2797
meta-llama/Llama-3.2-1B-Instruct	0.2706	0.2854	0.2711
meta-llama/Llama-3.2-3B-Instruct	0.3186	0.3321	0.3883

Užduotyse, reikalaujančiose loginio mąstymo (ARC) bei plačių žinių (MMLU). Numatytai, LLaMA 3.2 3B modelis pasiekė aukščiausius rezultatus visuose rodikliuose – tai atspindi jo architektūrinį pranašumą. Tuo tarpu LT-SynTune 1B modelis, nors nežymiai nusileido normintame ARC tikslume, lenkė bazinį 1B modelį MMLU užduotyje, kas rodo, jog *fine-tune* padėjo šiek tiek pagerinti semantinį apibendrinimą.

3.4.2. Gemma modelių rezultatų analizė

Vertinant Gemma modelių gebėjimą spręsti užduotis lietuvių kalba, buvo taip pat testuoti keli šio modelio variantai. Tiek mažesnio dydžio (1B), tiek didesnio (4B) versijos, su skirtingais *fine-tune* lygiais. Analizei naudoti tie patys penki pagrindiniai testai – ARC (lt), MMLU (lt), TruthfulQA MC1, TruthfulQA MC2 ir TruthfulQA Generation, kurie leidžia įvertinti tiek loginį mąstymą, tiek kalbinio atsako kokybę bei semantinį tikslumą. Toliau pateikiama lentelė apibendrina skirtingų Gemma modelių rezultatus (**3.3 lentelė**).

3.3 lentelė. Gemma 1B ir 4B modelių rezultatai lietuvių kalbos vertinimo užduotyse

Model	ARC (lt)	MMLU (lt)	TruthfulQA MC1	TruthfulQA MC2	TruthfulQA Generation
Gemma 4B Finetune	0.50842	0.469642	0.35496	0.53545	0.37821
Gemma 4B Finetune v2	0.508	0.470181	0.35741	0.53458	0.37576
Gemma 4B No Finetune	0.48359	0.488434	0.35741	0.54014	0.38678
Gemma 1B Finetune	0.3186	0.46199	0.27417	0.46679	0.28397
Gemma 1B Finetune v2	0.31397	0.279603	0.27417	0.46515	0.27907
Gemma 1B No Finetune	0.29251	0.278163	0.28152	0.46154	0.28274

Rezultatų skirtumai tarp Gemma 1 B ir Gemma 4 B modelių pirmiausia paaiškinami pačiu modelio masteliu. Didesnis 4 B variantas, turintis keturis kartus daugiau parametrų. Visuose bandymuose naudota ta pati LoRA schema ir ta pati kosinuso mokymosi greičio kreivė, tad jų rezultatus galima lyginti tiesiogiai. Lyginant matosi, kad beveik visose užduotyse (ARC, TruthfulQA, generacinės ROUGE/BLEU metrikos) šie modeliai startuoja iš aukštesnės „bazės“.

Mažesnio Gemma 1 B modelio atveju dirbtinis *fine-tune* duomenų rinkinys (≈ 7,7 M žetonų) davė didžiausią naudą: MMLU šoktelėjo net 18 pp, ARC – ~2,6 pp, o TruthfulQA MC2 taip pat kilstelėjo. Tai rodo, kad smulkesniam modeliui trūko semantinio papildymo, kurį kompensavo kruopščiai sudarytas sintetins duomenų rinkinys. Verta atsižvelgti, kad LoRA rango sumažinimas iki 32 (antrasis 1 B bandymas) pagerino gradientų stabilumą, bet nebegeneravo papildomos naudos „TruthfulQA“ metrikose – vadinasi, didesnis rangas 64 išnaudojo adapterių pajėgumą geriausiai.

Su 4 B modeliu paveikslas dviprasmis. Fine-tune pridėjo ~2 pp ARC laimėjimo, tačiau MMLU sumažėjo ~1,8 pp, o ROUGE reikšmės krito iki 1 pp. Tai leidžia daryti prielaidą, kad galbūt ribotas, stilistiškai vientisas rinkinys šiam modeliui nebeprideda naujos informacijos, o labiau „perrašo“ dalį plačios jau turėtos semantikos. Pradiniai gradientų normų šuoliai (iki ~8) rodo, kad adapteriai iš

pradžių agresyviai keitė svorius, po to susistabilizavo, tačiau dalį pradinių faktinių žinių vis tiek užgožė.

3.4.3. Gemma 1B modelio vertinimas su hibridiniu 50M žetonų rinkiniu

Buvo įvertintas Gemma 3 1B modelis, kuris buvo treniruotas naudojant išplėstinį, hibridinį duomenų rinkinį, sudarytą iš sintetinių, mC4-LT, Aya ir CulturaX duomenų. Bendra rinkinio apimtis siekė apie 50 milijonų žetonų, tai gerokai viršijo ankstesnių bandymų duomenų kiekį (kur naudotas 7.7M). Treniruotė buvo vykdyta taikant tą pačią LoRA metodiką ir treniravimo hiperparametrus kaip ir kituose eksperimentuose, siekiant palyginti rezultatus tiesiogiai.

Toliau pateikta lentelė (**3.4 lentelė**) apibendrina šio modelio rezultatus keliose vertinimo užduotyse kaip ir su prieš tai nagrinėtais modeliais: ARC), MMLU, TruthfulQA (MC1, MC2) bei kokybės metrikose (BLEU ir ROUGE-L).

3.4 lentelė. Gemma 1B modelio rezultatai po fine-tune su 50M žetonų hibridiniu duomenų rinkiniu

Model	ARC (acc)	ARC (acc_norm)	MMLU Avg	TruthfulQA MC1	TruthfulQA MC2
Gemma 1B Finetune 50M	0.30682	0.31818	0.30212	0.27785	0.47082
Gemma 1B Finetune v2	0.31397	0.279603	0.27417	0.46515	0.27907

Hibridinio 50 M žetonų rinkinio bandymas parodė, kad Gemma 1B Finetune 50M pasiekia aiškiai solidesnį bendrą balansą nei ankstesnė Finetune v2 versija, kuri buvo treniruota tik su 7,7 M generuotų duomenų. Nors „v2“ dar kiek lenkia naują modelį žaliame ARC tikslume (0,314 > 0,307), 50 M modelis ženkliai pranoksta visur, kur svarbi semantinika ir normalizuotas vertinimas.

ARC (acc_norm) pakilo iki 0,318 (+0,04 p.p.), MMLU – iki 0,302 (+0,028 p.p.), o TruthfulQA MC2 šoktelėjo net +0,192 p.p. (0,471 vs 0,279).

Didžiausias minusas – TruthfulQA MC1 (0,278 < 0,465), rodantis, kad platesnis, bet triukšmingesnis duomenų mišinys padidino riziką pateikti klaidingus ar nepakankamai patvirtintus teiginius. Vis dėlto MC2 ir normuotas ARC (kur griežčiau baudžiama už spėjimus) rodo, jog hibridinis korpusas sustiprino modelio žinių bazę ir loginį pagrįstumą, palikdamas mažiau atsitiktinių atsakymų.

3.5. Treniravimo ir eksperimentų apžvalga

Įgyvendinta visa suplanuota LLM treniravimo grandinė – nuo automatizuoto duomenų filtravimo (PPL + kalbos atpažinimas + dublikatų šalinimas) ir Kiln sintetinio korpuso generavimo iki LoRA adaptacijos Together AI aplinkoje bei vieningo vertinimo susistemintoje aplinkoje (RunPod → HF Evaluate → W & B).

Ne funkciniai tikslai taip pat pasiekti. 1 B modeliai vieną epochą apdorojo maždaug per 4 min., 4 B per 9 min. Visų eksperimentų VRAM poreikis neperžengė 80 GB ribos, o gradientų normos po ~20 žingsnių nusistovėjo, reikšmingo persimokymo nepastebėta.

Rezultatai patvirtino mastelio naudą. Iš LLaMA šeimos 3 B modelis išsiveržė į priekį BLEU (0,323) ir MMLU (0,388) testuose, o LT-SynTune-LLaMA-1B, būdamas keturis kartus mažesnis, lenkė tiek bazinį 1 B, tiek 3 B modelį ROUGE-1/L (0,284 / 0,284), rodydamas geresnį stilistinį tikslumą.

Gemma linijoje 4 B variantas be *fine-tune* išliko stipriausias ARC-LT (0,508) ir TruthfulQA MC2 (0,540), tuo tarpu Gemma-1B, treniruotas su 50 M žetonų hibridiniu rinkiniu, pasiekė geriausią balansą tarp ARC norm (0,318) ir MMLU (0,302) bei pakėlė MC2 tikslumą +0,19 pp lyginant su ankstesne 7,7 M versija, tačiau kiek suprastėjo MC1 (0,278).

Lyginant su pilnai pertreniruotais Neurotechnology Lt-LLaMA-2 modeliais, PEFT adapteriai pasirodė stebėtinai konkurencingi. Gemma-4B (PEFT) aplenkė Lt-LLaMA-2-13B net +22 pp MMLU ir +18 pp MC2, nusileisdama ARC-LT vos 3,7 pp. Gemma-1B-FT 50M, valdančiam tik ~8 % parametrų kiekio, pavyko pakilti virš 13 B varianto MMLU (+3,8 pp) ir MC2 (+11,9 pp), sunaudojant dešimt kartų mažiau VRAM. Be to, LT-SynTune-LLaMA-1B (0,6 % svorių) stilistiškai aplenkė abu Neurotechnology modelius ROUGE metrikose.

Sukurta PEFT strategija leido 1B ir 4 B klasės modeliams pasiekti ar net viršyti 7B ir 13 B pilnai pertreniruotos Lt-LLaMA-2 bazės rezultatus specifinėse lietuvių kalbos užduotyse, išnaudojant < 20 % skaičiavimo resursų. Tolimesnis tobulinimas turėtų koncentruotis į platesnį 4 B fine-tune korpusą ir griežtesnį faktų tikrinimą, siekiant pagerinti TruthfulQA MC1.

Išvados

1. Išanalizavus įvairių atvirojo kodo LLM architektūras ir jas pritaikius lietuvių kalbai, tapo aišku, kad kompaktiški, adapteriais (LoRA ar QLoRA) tobulinami modeliai (*LLaMA 3.2* ir *Gemma 3*) geriausiai atitinka darbo tikslus. Jie leidžia efektyviai išnaudoti ribotus skaičiavimo resursus ir tuo pačiu išlaikyti aukštą semantinį tikslumą. Palyginus 1 ir 4 milijardų parametrų klases su 7 ir 13 mlrd. variantais, paaiškėjo, kad adapterių dėka mažesni modeliai gali pasiekti lygiai tokį pat ar net geresnį našumą svarbiausiose testų kategorijose, kas rodo adaptacijos metodų efektyvumą ir praktinę naudą.
2. Įgyvendinant sekantį uždavinį – pažangių natūralios kalbos supratimo ir generavimo metodų diegimą *LT-SynTune-LLaMA-3.2-1B-Instruct* modelis sugebėjo stilistiškai pagerinti atsakymų kokybę (*ROUGE-1/-L* po +2,7 pp), o *Gemma 3 4 B* versija po LoRA adapterių pasiekė *ARC-LT* 0,508 ir *TruthfulQA MC2* 0,540, t. y. gerokai aplenkė pilnai pertreniruotą 13 B modelį. Tai patvirtina, jog adapterių metodika veikia ir dideliu parametrų skaičiumi pasižyminčiuose modeliuose, suteikdama pridėtinę vertę.
3. Duomenų integravimo ir jų įtakos nagrinėjimas leido patikrinti sinergiją tarp tvarkingai sudaryto 7,7 M žetonų sintetinės korpuso (*Kiln + Gemini*) ir 50 milijonų žetonų hibridinio rinkinio (*Aya + mC4-LT + CulturaX*). *Gemma 3 1 B* modelis su hibridiniu duomenų rinkiniu *MMLU* pakilo nuo 0,279 iki 0,302, *ARC-norm* nuo 0,279 iki 0,318 ir *TruthfulQA MC2* nuo 0,279 iki 0,471, parodė, kad kokybiškas filtruotų tekstų papildymas ženkliai sustiprina modelio žinių bazę.
4. Galiausiai sėkmingai adaptuotas ir sukurtas modelis buvo įvertintas su *ARC-LT, MMLU-LT* ir *TruthfulQA-LT* testais. Geriausius *MMLU* rezultatus (0,388) ir *BLEU* (0,323) pasiekė *LLaMA 3.2 3 B*, o *Gemma 3 4 B* nepralenkiamai įsitvirtino žinių komentavimo metrikose *MC2*. Svarbu, kad visi eksperimentai tilpo į vieną *A100* GPU (< 12 min.), o visi sudaryti procesai (*TogetherAI, RunPod, Weights & Biases*) leidžia operatyviai kartoti tyrimą gavus prieigą prie didesnių resursų. Šie rezultatai patvirtina, jog darbo tikslas buvo pasiektas, pademonstruojant, kad LoRA, PEFT kartu su mišria duomenų strategija leidžia sėkmingai adaptuoti ir patobulinti lietuviškus dialogo modelius su mažesniais kaštais, gaunant praktiškai reikšmingus rezutltatus, ypač lyginant su brangesniais, pilno pertreniravimo reikalaujančiais sprendimais.

Literatūros sąrašas

1. VASWANI, A. ir kt. Attention Is All You Need. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-04-25]. arXiv:1706.03762 [cs]. Prieiga per internetą: <http://arxiv.org/abs/1706.03762>.

2. OPENAI ir kt. GPT-4 Technical Report. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-04-25]. Prieiga per internetą: <https://arxiv.org/abs/2303.08774>.

3. LAMPLE, G. - CONNEAU, A. Cross-lingual Language Model Pretraining. [interaktyvus]. .[s.l.]: arXiv, 2019. [žiūrėta 2025-04-25]. arXiv:1901.07291 [cs]. Prieiga per internetą: <http://arxiv.org/abs/1901.07291>.

4. DEVLIN, J. ir kt. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. [interaktyvus]. .[s.l.]: arXiv, 2019. [žiūrėta 2025-04-25]. arXiv:1810.04805 [cs]. Prieiga per internetą: <http://arxiv.org/abs/1810.04805>.

5. XUE, L. ir kt. mT5: A massively multilingual pre-trained text-to-text transformer. [interaktyvus]. .[s.l.]: arXiv, 2021. [žiūrėta 2025-04-25]. arXiv:2010.11934 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2010.11934>.

6. NAVEED, H. ir kt. A Comprehensive Overview of Large Language Models. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-04-25]. arXiv:2307.06435 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2307.06435>.

7. STANKEVIČIUS, L. - LUKOŠEVIČIUS, M. Testing pre-trained Transformer models for Lithuanian news clustering. [interaktyvus]. .[s.l.]: arXiv, 2020. [žiūrėta 2025-04-25]. arXiv:2004.03461 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2004.03461>.

8. YE, J. ir kt. A Comprehensive Capability Analysis of GPT-3 and GPT-3.5 Series Models. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-04-25]. arXiv:2303.10420 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2303.10420>.

9. LIU, J. ir kt. What Makes Good In-Context Examples for GPT-3? [interaktyvus]. .[s.l.]: arXiv, 2021. [žiūrėta 2025-04-25]. arXiv:2101.06804 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2101.06804>.

10. MILANI FITRIA, K. Information Retrieval Performance in Text Generation using Knowledge from Generative Pre-trained Transformer (GPT-3). In *Jambura Journal of Mathematics* . 2023. Vol. 5, no. 2, p. 327–338. .

11. ALMASI, M. - SCHIØNNING, A. Fine-Tuning GPT-3 for Synthetic Danish News Generation. In *Proceedings of the 16th International Natural Language Generation Conference* [interaktyvus]. Prague, Czechia: Association for Computational Linguistics, 2023. p. 54–68. [žiūrėta 2025-04-26]. Prieiga per internetą: <https://aclanthology.org/2023.inlg-main.4>.

12. MILANI FITRIA, K. Design of text generator application with OpenAI GPT-3. In *JEECOM Journal of Electrical Engineering and Computer* . 2023. Vol. 5, no. 2, p. 135–142. .

13. QI, S. - ZHANG, H. Text Summarization Quality Detection Based on GPT-3. In *Applied and Computational Engineering* . 2023. Vol. 8, no. 1, p. 817–822. .

14. BROWN, T.B. ir kt. Language Models are Few-Shot Learners. [interaktyvus]. .[s.l.]: arXiv, 2020. [žiūrėta 2025-05-20]. Prieiga per internetą: <https://arxiv.org/abs/2005.14165>.

15. TOUVRON, H. ir kt. Llama 2: Open Foundation and Fine-Tuned Chat Models. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-04-26]. arXiv:2307.09288 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2307.09288>.

16. WANG, C. ir kt. Language Models with Transformers. [interaktyvus]. .[s.l.]: arXiv, 2019. [žiūrėta 2025-05-16]. arXiv:1904.09408 [cs]. Prieiga per internetą: <http://arxiv.org/abs/1904.09408>.

17. WORKSHOP, B. ir kt. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-04-26]. arXiv:2211.05100 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2211.05100>.

18. SHLIAZHKO, O. ir kt. mGPT: Few-Shot Learners Go Multilingual. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-04-26]. arXiv:2204.07580 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2204.07580>.

19. JIANG, A.Q. ir kt. Mistral 7B. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-05-20]. Prieiga per internetą: <https://arxiv.org/abs/2310.06825>.

20. GRATTAFIORI, A. ir kt. The Llama 3 Herd of Models. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-05-20]. arXiv:2407.21783 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2407.21783>.

21. HUANG, W. ir kt. An empirical study of LLaMA3 quantization: from LLMs to MLLMs. In *Visual Intelligence* . 2024. Vol. 2, no. 1, p. 36. .

22. TEAM, G. ir kt. Gemma 2: Improving Open Language Models at a Practical Size. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-05-20]. arXiv:2408.00118 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2408.00118>.

23. TEAM, G. ir kt. Gemma 3 Technical Report. [interaktyvus]. .[s.l.]: arXiv, 2025. [žiūrėta 2025-05-20]. arXiv:2503.19786 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2503.19786>.

24. MARTINS, P.H. ir kt. EuroLLM: Multilingual Language Models for Europe. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-05-20]. arXiv:2409.16235 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2409.16235>.

25. ÜSTÜN, A. ir kt. Aya Model: An Instruction Finetuned Open-Access Multilingual Language Model. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-05-20]. arXiv:2402.07827 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2402.07827>.

26. RUST, P. ir kt. How Good is Your Tokenizer? On the Monolingual Performance of Multilingual Language Models. [interaktyvus]. .[s.l.]: arXiv, 2021. [žiūrėta 2025-05-20]. arXiv:2012.15613 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2012.15613>.

27. MIN, B. ir kt. Recent Advances in Natural Language Processing via Large Pre-Trained Language Models: A Survey. [interaktyvus]. .[s.l.]: arXiv, 2021. [žiūrėta 2025-05-16]. arXiv:2111.01243 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2111.01243>.

28. HU, E.J. ir kt. LoRA: Low-Rank Adaptation of Large Language Models. [interaktyvus]. .[s.l.]: arXiv, 2021. [žiūrėta 2025-05-16]. arXiv:2106.09685 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2106.09685>.

29. HUANG, Y. ir kt. Advancing Transformer Architecture in Long-Context Large Language Models: A Comprehensive Survey. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-05-16]. arXiv:2311.12351 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2311.12351>.

30. GUPTA, K. ir kt. Continual Pre-Training of Large Language Models: How to (re)warm your model? [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-05-16]. arXiv:2308.04014 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2308.04014>.

31. WEI, J. ir kt. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-05-16]. arXiv:2201.11903 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2201.11903>.

32. XI, Z. ir kt. Self-Polish: Enhance Reasoning in Large Language Models via Problem Refinement. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-04-26]. arXiv:2305.14497 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2305.14497>.

33. SUN, X. ir kt. Text Classification via Large Language Models. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-04-26]. arXiv:2305.08377 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2305.08377>.

34. ZENG, J. ir kt. TIM: Teaching Large Language Models to Translate with Comparison. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-04-26]. arXiv:2307.04408 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2307.04408>.

35. BAVARIAN, M. ir kt. Efficient Training of Language Models to Fill in the Middle. [interaktyvus]. .[s.l.]: arXiv, 2022. [žiūrėta 2025-04-26]. arXiv:2207.14255 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2207.14255>.

36. QIN, Y. ir kt. Disentangled Representation Learning with Large Language Models for Text-Attributed Graphs. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-04-26]. arXiv:2310.18152 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2310.18152>.

37. HAN, I. ir kt. HyperAttention: Long-context Attention in Near-Linear Time. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-04-26]. arXiv:2310.05869 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2310.05869>.

38. SUN, Y. ir kt. Retentive Network: A Successor to Transformer for Large Language Models. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-05-16]. arXiv:2307.08621 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2307.08621>.

39. BESTA, M. ir kt. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. In *Proceedings of the AAAI Conference on Artificial Intelligence* . 2024. Vol. 38, no. 16, p. 17682–17690. .

40. NIE, S. ir kt. Scaling up Masked Diffusion Models on Text. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-05-20]. Prieiga per internetą: <https://arxiv.org/abs/2410.18514>.

41. BRAUWERS, G. - FRASINCAR, F. A General Survey on Attention Mechanisms in Deep Learning. In *IEEE Transactions on Knowledge and Data Engineering* . 2023. Vol. 35, no. 4, p. 3279–3298. .

42. DAI, Z. ir kt. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. [interaktyvus]. .[s.l.]: arXiv, 2019. [žiūrėta 2025-05-16]. arXiv:1901.02860 [cs]. Prieiga per internetą: <http://arxiv.org/abs/1901.02860>.

43. ALAJRAMI, A. - ALETRAS, N. How does the pre-training objective affect what large language models learn about linguistic properties? [interaktyvus]. .[s.l.]: arXiv, 2022. [žiūrėta 2025-05-16]. arXiv:2203.10415 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2203.10415>.

44. TAKASE, S. ir kt. Spike No More: Stabilizing the Pre-training of Large Language Models. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-05-16]. arXiv:2312.16903 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2312.16903>.

45. MOSLEM, Y. ir kt. Fine-tuning Large Language Models for Adaptive Machine Translation. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-05-16]. arXiv:2312.12740 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2312.12740>.

46. DETTMERS, T. ir kt. QLoRA: Efficient Finetuning of Quantized LLMs. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-05-16]. arXiv:2305.14314 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2305.14314>.

47. LV, K. ir kt. Full Parameter Fine-tuning for Large Language Models with Limited Resources. [interaktyvus]. .[s.l.]: arXiv, 2024. [žiūrėta 2025-05-16]. arXiv:2306.09782 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2306.09782>.

48. ZHANG, L. ir kt. Dissecting the Runtime Performance of the Training, Fine-tuning, and Inference of Large Language Models. [interaktyvus]. .[s.l.]: arXiv, 2023. [žiūrėta 2025-05-16]. arXiv:2311.03687 [cs]. Prieiga per internetą: <http://arxiv.org/abs/2311.03687>.

49. NAKVOSAS, A. ir kt. Open Llama2 Models for the Lithuanian Language. In *Informatica* . 2025. p. 1–22. .