НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут прикладного системного аналізу

Кафедра штучного інтелекту

«На правах рукопису»
УДК 004.93'1:811.161.2'322

До захисту допущено:
В. о. завідувачки кафедри
_____ Ірина ДЖИГИРЕЙ
«___»_____20__ р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-науковою програмою «Комп'ютерні науки»

зі спеціальності 122 «Комп'ютерні науки»

на тему: «Методи мультикласового сентимент-аналізу на основі мовних моделей в умовах змішаного використання мов»

Виконав:
студент ІІ курсу, групи КІ-41мн
Гіловянц Кирил Дмитрович _____

Науковий керівник:
доцент кафедри ШІ, к.т.н.
Шаповал Наталія Віталіївна _____

Рецензент:
професор кафедри ММСА, д.т.н., професор,
Недашківська Надія Іванівна _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2026 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий інститут прикладного системного аналізу
Кафедра штучного інтелекту

Рівень вищої освіти – другий (магістерський)

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-наукова програма «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
В. о. завідувачки кафедри
_____ Ірина ДЖИГИРЕЙ
«30» січня 2026 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Гіловянцу Кирилу Дмитровичу

1. Тема дисертації «Методи мультикласового сентимент-аналізу на основі мовних моделей в умовах змішаного використання мов», науковий керівник дисертації Шаповал Наталія Віталіївна, к.т.н, затверджені наказом по університету від «30» березня 2026 р. №1315-с.

2. Термін подання студентом дисертації «15» травня 2026 року

3. Об’єкт дослідження процес автоматичного визначення тональності українсько-російських код-змішаних текстів соціальних мереж засобами сучасних мовних моделей

Предмет дослідження методи адаптації мовних моделей до мультикласового сентимент-аналізу код-змішаних текстів з урахуванням дисбалансу класів і мовної структури повідомлень.

4. Вихідні дані розширений корпус COSMUS+ (33 267 текстів) із автоматичною розміткою через API великих мовних моделей; дотреновані encoder-моделі mBERT, XLM-RoBERTa Base, XLM-RoBERTa Large, UkrRoBERTa, Modern-LiBERTa; україноцентричні decoder-only моделі MamayLM 4B і MamayLM 12B; результати експериментів

5. Перелік завдань, які потрібно розробити

1) проаналізувати предметну область сентимент-аналізу, мультикласової класифікації та явища код-змішування в українсько-російському контексті;

2) виконати огляд наукових джерел із сентимент-аналізу, мовних моделей трансформерної архітектури, методів дотренування й параметрично-ефективної адаптації;

3) дослідити структуру корпусу COSMUS, проблему дисбалансу класів та сформувати розширений корпус COSMUS+ на основі Telegram-даних;

4) розробити методику експериментального дослідження: обрати моделі, метрики оцінювання та протокол тестування на COSMUS і COSMUS+;

5) провести експерименти з комерційними API-моделями у режимах zero-shot і few-shot, локальними encoder-моделями з повним дотренуванням та decoder-моделлю MamayLM у режимах промптингу й QLoRA;

6) розробити й експериментально перевірити пайплайн CSSent для адаптації encoder-моделей до код-змішаних текстів, виконати абляційний аналіз його компонентів;

7) проаналізувати результати, виконати оцінювання калібрування ймовірностей, LIME/SHAP-пояснюваність і якісний аналіз показових помилок, сформулювати висновки;

8) оформити пояснювальну записку магістерської дисертації та підготувати ілюстративний матеріал.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу презентація дипломної роботи

7. Орієнтовний перелік публікацій тези на онлайн-конференції «Перевірені часом і нові наукові парадигми XXI століття»; стаття в періодичному виданні «Штучний інтелект»

8. Дата видачі завдання «30» січня 2026 року

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз предметної області, уточнення теми, мети, об’єкта, предмета та завдань дослідження	23.03.2026 – 30.03.2026	Виконано
2	Огляд наукових джерел із сентимент-аналізу, код-змішування та мовних моделей	31.03.2026 – 06.04.2026	Виконано
3	Аналіз корпусу COSMUS, структури класів, мовних груп і проблеми дисбалансу даних	07.04.2026 – 13.04.2026	Виконано
4	Розроблення методики експериментального дослідження, вибір моделей, метрик і протоколу оцінювання	14.04.2026 – 20.04.2026	Виконано
5	Проведення експериментів з API-моделями, локальними encoder-моделями та MamayLM	21.04.2026 – 27.04.2026	Виконано
6	Розроблення та експериментальна перевірка пайплайну CSSent, виконання абляційного аналізу	28.04.2026 – 04.05.2026	Виконано
7	Аналіз результатів, побудова порівняльних таблиць і рисунків, підготовка висновків	05.05.2026 – 10.05.2026	Виконано
8	Оформлення пояснювальної записки	11.05.2026 – 14.05.2026	Виконано

Студент Кирил ГІЛОВЯНЦ

Науковий керівник Наталія ШАПОВАЛ

РЕФЕРАТ

Магістерська дисертація: 118 с., 13 рис., 16 табл., 38 посилань.

СЕНТИМЕНТ-АНАЛІЗ, КОД-ЗМІШУВАННЯ, УКРАЇНСЬКА МОВА, МОВНІ МОДЕЛІ, COSMUS, ТРАНСФОРМЕР, CSSENT, QLORA, MODERN-LIBERTA, MAMAYLM

Об'єкт дослідження – процес автоматичного визначення тональності коротких українсько-російських код-змішаних текстів із соціальних мереж.

Предмет дослідження – методи адаптації мовних моделей до мультикласового сентимент-аналізу код-змішаних текстів з урахуванням дисбалансу класів і мовної структури повідомлень.

Мета роботи – розробити та експериментально перевірити комплексний підхід до сентимент-аналізу українсько-російських код-змішаних текстів.

Методи дослідження – дотренування трансформерних енкодерних моделей, промптинг, QLoRA, дистиляція знань, аугментація даних, аналіз калібрування та LIME/SHAP-пояснюваність.

Наукова новизна полягає у запропонованому комплексному підході CSSent для адаптації енкодерних моделей, який поєднує мовно-зумовлену класифікаційну голову, фокусну функцію втрат, допоміжне визначення мови, аугментацію та дистиляцію знань. Показано, що ефект CSSent залежить від базової моделі: для XLM-RoBERTa Large Macro F1 зростає з 0,6500 до 0,6684-0,6691, тоді як для Modern-LiBERTa приріст не є універсальним.

Експерименти виконано на COSMUS і COSMUS+. Порівняно API-моделі, локальні енкодерні моделі, MamayLM у режимах промптингу та QLoRA, а також CSSent-конфігурації; найкращий результат на COSMUS отримала Gemini 3.0 Flash ukr few-shot (Macro F1 = 0,6768), близький результат показала Modern-LiBERTa (0,6742), а MamayLM 12B після QLoRA досягла 0,6654.

ABSTRACT

Master’s thesis: 118 pages, 13 figures, 16 tables, 38 references.

SENTIMENT ANALYSIS, CODE-SWITCHING, UKRAINIAN LANGUAGE, LANGUAGE MODELS, COSMUS, TRANSFORMER, CSSENT, QLORA, MODERN-LIBERTA, MAMAYLM

Research object – automatic sentiment classification of short Ukrainian-Russian code-mixed social media texts.

Research subject – methods for adapting language models to multiclass sentiment analysis of code-mixed texts under class imbalance and language-structure variation.

The aim of the work – to develop and experimentally evaluate a combined approach to sentiment analysis of Ukrainian-Russian code-mixed texts.

Research methods – transformer encoder fine-tuning, prompting, QLoRA, knowledge distillation, data augmentation, calibration analysis and LIME/SHAP explainability.

The scientific novelty lies in the proposed combined CSSent approach for adapting encoder models, which combines a language-conditioned classification head, focal loss, auxiliary language identification, augmentation and knowledge distillation. CSSent is backbone-dependent: for XLM-RoBERTa Large, Macro F1 increases from 0.6500 to 0.6684-0.6691, while for Modern-LiBERTa the improvement is not universal.

The experiments are conducted on COSMUS and COSMUS+. The thesis compares API-based LLMs, local encoder models, MamayLM with prompting and QLoRA, and CSSent configurations; the best COSMUS result is achieved by Gemini 3.0 Flash ukr few-shot (Macro F1 = 0.6768), followed closely by Modern-LiBERTa (0.6742), while MamayLM 12B after QLoRA reaches 0.6654.

ЗМІСТ

ВСТУП

Соціальні[1] мережі стали одним із головних джерел коротких публічних реакцій на суспільні події, державні сервіси, медіаконтент і повсякденний досвід. Для українського онлайн-простору характерною є не лише емоційна насиченість таких текстів, а й код-змішування, тобто використання українських і російських мовних елементів в одному повідомленні або в межах одного обговорення; у міжнародній літературі це явище описують термінами code-switching і code-mixing. До таких проявів належать суржик, цитування іншою мовою, перемикання мов між реченнями та змішування мов усередині речення. У таких умовах сентимент-аналіз ускладнюється: модель має визначати не тільки позитивну, негативну чи нейтральну тональність, а й випадки mixed, коли в одному повідомленні співіснують протилежні тональності.

Актуальність дослідження зумовлена тим, що більшість готових моделей і бенчмарків орієнтовані на монолінгвальні або англомовні сценарії, тоді як українсько-російські код-змішані тексти мають специфічну лексику, нестандартну орфографію, політичний і соціальний контекст, а також сильний дисбаланс класів. Особливо складним є клас mixed: у початковому корпусі COSMUS [1] він становить лише невелику частку прикладів, але саме він часто є найбільш інформативним для аналізу реальних реакцій користувачів.

Метою магістерської дисертації є розроблення та експериментальна перевірка комплексного підходу до мультикласового сентимент-аналізу українсько-російських код-змішаних текстів на основі сучасних мовних

[1] Тут і нижче використано такий інструмент штучного інтелекту як чат-бот з генеративним штучним інтелектом Gemini, виключно для корегування та редагування тексту, створеного автором цієї дипломної роботи, на основі автоматизованої перевірки граматики, структури та стилю, що відповідає Політиці використання штучного інтелекту для академічної діяльності в КПІ ім. Ігоря Сікорського (протокол №11 Вченої ради КПІ ім. Ігоря Сікорського від 11 грудня 2023 р.).

моделей. Запропонований підхід реалізовано у вигляді пайплайну CSSent, який поєднує адаптацію encoder-моделей з урахуванням мовної структури повідомлень, дисбалансу класів і складності категорії mixed.

Для досягнення мети в роботі було: проаналізовано предметну область сентимент-аналізу, мультикласової класифікації та код-змішування; досліджено структуру початкового корпусу COSMUS і сформовано розширений корпус COSMUS+; оцінено комерційні API-моделі у zero-shot і few-shot режимах; виконано повне дотренування локальних encoder-моделей; перевірено MamayLM 4B/12B [2] у режимах промптингу та QLoRA; запропоновано й протестовано CSSent; проведено аналіз калібрування, пояснюваності LIME/SHAP [3, 4] і показових помилок моделей.

Об’єктом дослідження є процес автоматичного визначення тональності коротких українсько-російських код-змішаних текстів із соціальних мереж. Предметом дослідження є методи адаптації мовних моделей до мультикласового сентимент-аналізу в умовах змішаного використання мов, дисбалансу класів і наявності категорії mixed.

У роботі використано методи обробки природної мови, повного дотренування трансформерних encoder-моделей, промптингу, параметрично-ефективного дотренування QLoRA, дистиляції знань, аугментації даних, аналізу калібрування ймовірностей та LIME/SHAP-пояснюваності.

Наукова новизна роботи полягає у запропонованому та експериментально перевіреному комплексному підході CSSent до адаптації encoder-моделей для мультикласового сентимент-аналізу українсько-російських код-змішаних текстів. CSSent поєднує мовно-зумовлену класифікаційну голову, зважену фокусну функцію втрат [5], допоміжну задачу визначення мови, аугментацію код-змішаних прикладів і дистиляцію знань від LLM через м'які мітки. Таке поєднання дає змогу в межах однієї схеми навчання одночасно враховувати мовну структуру повідомлення, дисбаланс класів, складність категорії mixed і невпевненість моделі-вчителя. У роботі показано, що ефективність такого комбінованого підходу залежить

від базової encoder-основи: він покращує XLM-RoBERTa Large [6], але не є універсальним способом підвищення якості для Modern-LiBERTa [7]. Додатковим внеском є формування COSMUS+, порівняння локальних і API-моделей у єдиному протоколі та комплексний аналіз, який поєднує агреговані метрики, F1 класу mixed, калібрування, пояснюваність і розбір показових випадків.

Практичне значення роботи полягає в отриманні відтворюваного порівняння стратегій адаптації мовних моделей для українсько-російського код-змішаного контенту. Результати можуть бути використані для побудови систем моніторингу суспільних настроїв, аналізу реакцій у Telegram, попереднього сортування великих масивів коментарів і вибору компромісу між якістю, вартістю та контрольованістю моделі.

РОЗДІЛ 1 ТЕОРЕТИЧНІ ЗАСАДИ МУЛЬТИКЛАСОВОГО СЕНТИМЕНТ-АНАЛІЗУ КОД-ЗМІШАНИХ ТЕКСТІВ

1.1 Актуальність та постановка задачі

Стрімкий розвиток соціальних мереж та месенджерів створив безпрецедентний обсяг текстових даних, що відображають суспільні настрої, емоційний стан та ставлення користувачів до різноманітних подій, товарів, послуг та суспільних явищ. Автоматичний аналіз тональності (сентимент-аналіз) цих текстових масивів є одним із ключових завдань обробки природної мови (Natural Language Processing, NLP), що має значне практичне застосування в маркетингу, політичному аналізі, моніторингу суспільних настроїв та системах підтримки прийняття рішень.

Для української мови задача сентимент-аналізу ускладнюється низкою специфічних факторів. По-перше, українська належить до мов із обмеженими ресурсами (low-resource languages) у контексті NLP, що означає значно менший обсяг розмічених корпусів, передтренованих моделей та дослідницьких робіт порівняно з англійською, китайською чи іспанською мовами. По-друге, унікальною особливістю українського цифрового простору є поширене явище код-змішування, тобто одночасне використання української та російської мов в одному тексті, що формує так званий «суржик». Це лінгвістичне явище є наслідком тривалої двомовності українського суспільства та створює додаткові виклики для NLP-систем, які традиційно орієнтовані на одномовні тексти.

Сучасні трансформерні моделі забезпечують високу якість класифікації для англійської мови, проте для мов з обмеженими ресурсами ситуація суттєво відрізняється. Українська мова, незважаючи на створення корпусу Kobza обсягом ~60 мільярдів токенів [7] та появу спеціалізованих моделей, все ще потребує розвитку інструментарію для задач класифікації тональності. Моделі, треновані на одномовних корпусах, часто не справляються зі

змішаними текстами, що знижує практичну придатність систем аналізу тональності.

За даними дослідження [1], яке представило датасет COSMUS (COde-Switched MUltilingual Sentiment) обсягом 12 224 тексти з Telegram-каналів, значна частина повідомлень із українських соціальних мереж містить елементи код-змішування. Автори анотували тексти за тональністю (positive, negative, neutral, mixed) та мовною належністю (uk, ru, code-switching). Вони провели початковий бенчмаркінг із few-shot промптингом та дотренуванням mBERT [8] і UkrRoBERTa [9]; найкращий результат оригінального дослідження становив приблизно 73,6% accuracy / F1 ≈ 0,64 для UkrRoBERTa із додатковою аугментацією на кшталт word substitution [1], а простіше plain fine-tuning у поточному відтворенні дає нижчий результат, тому що частина приросту в оригінальному дослідженні була пов'язана саме з аугментацією, а не лише з вибором архітектури. Однак мультикласова класифікація сентименту в таких умовах залишається недостатньо дослідженою проблемою, і порівняльна оцінка найновіших україноцентричних архітектур, таких як MamayLM та Modern-LiBERTa, для код-змішаного контенту досі відсутня.

Водночас з'явилися нові україноцентричні моделі, які ще не були систематично представлені в відкритому COSMUS-бенчмарку для цієї конкретної задачі. Encoder-модель Modern-LiBERTa [7], натренована на корпусі Kobza, є наймасштабнішою україноцентричною encoder-моделлю. Decoder-only MamayLM [2], побудована на базі Gemma 3, є однією з перших великих україноцентричних генеративних моделей. Тому доцільним є порівняння цих моделей із мультилінгвальними encoder-підходами (XLM-RoBERTa) та комерційними API-моделями саме на код-змішаних даних COSMUS.

Актуальність цього дослідження визначається потребою порівняти різні типи мовних моделей на єдиному корпусі, а також перевірити спеціалізовану схему адаптації для код-змішаного сентимент-аналізу. У

таких текстах модель має працювати одночасно з мовною неоднорідністю повідомлень, дисбалансом класів і семантичною складністю категорії mixed. Тому в роботі розглядається комплексний підхід CSSent для encoder-моделей, спрямований на врахування цих властивостей задачі, а також порівнюються альтернативні режими використання мовних моделей: API-промптинг, повне дотренування encoder-моделей, промптинг MamayLM і QLoRA-дотренування.

Окрему практичну важливість має порівняння спеціалізованих локальних encoder-моделей із великими комерційними LLM, такими як Gemini 3.0 Flash. Комерційні LLM мають сильні zero-shot і few-shot можливості, але їх використання пов'язане з вартістю API-запитів, залежністю від зовнішнього сервісу, обмеженим контролем над протоколом інференсу та потенційними обмеженнями щодо приватності даних. Натомість encoder-моделі на кшталт Modern-LiBERTa або XLM-RoBERTa Large є значно компактнішими, можуть розгортатися локально й дотреновуватися під конкретну задачу. Тому науково й практично важливим є не лише пошук найвищого значення Macro F1, а й оцінювання компромісу між якістю, розміром моделі, вартістю обчислень і контрольованістю застосування.

1.2 Сентимент-аналіз: основні підходи та виклики

1.2.1 Визначення та еволюція підходів до аналізу тональності

Сентимент-аналіз (аналіз тональності, opinion mining) є підгалуззю обробки природної мови, що займається автоматичним визначенням суб'єктивної інформації в текстових даних, зокрема ідентифікацією емоцій, думок, оцінок та ставлення автора до об'єкта висловлювання. Історично в аналізі тональності можна виділити кілька ключових напрямів.

Перший напрям (з кінця 1990-х): лексичні методи та методи на основі правил. Ранні системи базувалися на використанні сентимент-словників, тобто списків позитивної та негативної лексики [10]. Наприклад, системи SentiWordNet, AFINN та VADER присвоювали словам значення полярності. У дослідженні [11] було представлено один із перших rule-based методів для аналізу відгуків українською мовою. Ці системи були інтерпретованими та не потребували розмічених даних, проте гірше враховували контекст, іронію, сарказм та доменну специфіку.

Другий напрям (з початку 2000-х): класичне машинне навчання. Революційна робота [12] продемонструвала перевагу методів ML (Naive Bayes, SVM, Random Forest) над лексичними підходами. Їхня робота базувалася на вручну створених ознаках: bag-of-words, TF-IDF, n-грами. В роботі [13] досліджували тренди тональності в українських та російських новинних статтях, застосовуючи саме традиційні методи ML.

Третій напрям (з 2013 року): нейронні мережі та глибоке навчання. Впровадження розподілених представлень слів (word embeddings, як-от Word2Vec, GloVe) дозволило кодувати семантичні відношення. З'явилися рекурентні мережі (LSTM, GRU) та згорткові мережі для текстів [14], які встановили нові стандарти якості. Вони здатні краще моделювати послідовні залежності та контекст заперечень.

Четвертий напрям (з 2018 року): трансформерні моделі та LLM. Архітектура Transformer та парадигма «pre-train, then fine-tune» [6, 8] радикально змінили ландшафт NLP. Авторегресійні decoder-only моделі (великі мовні моделі, LLM), що генерують текст послідовно, передбачаючи кожен наступний токен на основі попередніх, довели здатність вирішувати задачі класифікації у режимах zero-shot або few-shot.

1.2.2 Мультикласова класифікація тональності

Більшість досліджень традиційно зосереджуються на бінарній класифікації (positive/negative), проте реальні тексти мають значно складніший емоційний спектр. Мультикласова схема розширює палітру до 3 або 4 категорій: бінарна (positive/negative), трикласова (positive/negative/neutral), чотирикласова (positive/negative/neutral/mixed).

У контексті даного дослідження використовується чотирикласова схема класифікації, прийнята в датасеті COSMUS [1]:

- Positive: виражає позитивне ставлення, задоволення, радість.
- Negative: виражає негативне ставлення, критику, розчарування.
- Neutral: фактологічний текст без вираженого емоційного забарвлення.
- Mixed: одночасно містить позитивні ТА негативні елементи.

Перехід до мультикласової схеми суттєво ускладнює задачу. Зростає міжкласова неоднозначність (межі між «слабо позитивним» і «нейтральним» стають розмитими), внутрішньокласова варіативність, а також загострюється проблема дисбалансу класів [15].

Клас mixed є найскладнішим. Він передбачає розпізнавання одночасної присутності позитивних та негативних сигналів в одному тексті, що часто виникає при оцінці різних аспектів об'єкта (aspect-level sentiment). Моделі часто помиляються, відносячи такі тексти до negative (через домінування критики) або positive. Суттєвим викликом є також дисбаланс: у датасеті COSMUS клас mixed становить близько 5% від загальної кількості текстів (608 з 12 224), а на тестовій вибірці лише 60 із 1 223 зразків, що дає співвідношення найбільшого до найменшого класу приблизно 7,7:1.

У наявних дослідженнях близькі проблеми розв'язують кількома групами методів. Для аспектно орієнтованого сентимент-аналізу

використовують моделі, які розділяють оцінки різних аспектів одного об'єкта, а не зводять повідомлення до однієї загальної тональності [15]. Для мультикласових задач з дисбалансом застосовують вагові коефіцієнти класів, змінені функції втрат або фокусування на складних прикладах [5]. Для код-змішаних текстів досліджують мовну сегментацію, багатомовні представлення та явне врахування точок перемикання мов [16, 17]. Отже, для класу mixed потрібне не лише збільшення кількості даних, а й поєднання механізмів, які враховують дисбаланс, мовну структуру та семантичну амбівалентність повідомлення.

Метрики та дисбаланс для мультикласової оцінки. Дисбаланс класів негативно впливає на навчання моделей, оскільки при мінімізації загальної помилки модель зміщується до мажоритарного класу. Для подолання цієї проблеми можуть застосовуватися стратегії на рівні даних (Data augmentation, oversampling) або на рівні алгоритмів (Weighted Cross-Entropy Loss, Focal Loss).

З огляду на дисбаланс класів, accuracy використовується лише як допоміжна інтерпретована метрика, що показує загальну частку правильних відповідей. Основною метрикою дослідження є Macro F1-score: вона обчислює F1 для кожного класу окремо, а потім усереднює ці значення з однаковою вагою для positive, negative, neutral і mixed. Такий підхід є принциповим для COSMUS, оскільки за сильного дисбалансу класів агрегована точність може маскувати помилки на міноритарній категорії mixed. Отже, Macro F1 не дозволяє моделі отримати високу підсумкову оцінку лише завдяки правильному розпізнаванню мажоритарних класів.

Сентимент-аналіз для мов з обмеженими ресурсами. Українська мова, як мова з відносно обмеженими ресурсами, потребує особливих стратегій дослідження. Крос-лінгвальне трансферне навчання [18, 19] дозволяє інтегрувати знання з ресурсно багатих мов. Zero-shot підходи також надають альтернативу без використання розмічених даних цільовою мовою [20].

У контексті українського сентимент-аналізу в роботі [21] було порівняно дотренування BERT та XLM-RoBERTa для бінарної класифікації, причому найкращий результат отримала XLM-RoBERTa. У [22] трансформерні архітектури застосовано вже для трикласової класифікації. Саме тому mBERT, XLM-RoBERTa та україноцентричні encoder-моделі обрано як природну основу для подальшого дослідження: вони вже показали придатність для українського сентимент-аналізу, але їхня поведінка на чотирикласовій код-змішаній задачі з категорією mixed лишалася недостатньо перевіреною.

1.2.3 Практична інтерпретація класу mixed

Клас mixed у чотирикласовій схемі сентимент-аналізу не є простим проміжним станом між positive і negative. Його коректна інтерпретація передбачає, що в межах одного повідомлення справді присутні обидві протилежні тональності, а не лише невпевненість анотатора або відсутність чіткого класу. Саме тому mixed потрібно відрізняти від neutral: нейтральне повідомлення може містити інформацію про негативну подію, але не містити авторської оцінки, тоді як mixed поєднує різні тональності. Наприклад, повідомлення може підтримувати певне рішення, але критикувати спосіб його реалізації; дякувати за допомогу, але одночасно висловлювати незадоволення організацією; позитивно оцінювати одну частину події і негативно іншу.

Для моделей машинного навчання така відмінність є складною з кількох причин. По-перше, mixed часто має aspect-level природу: позитивна та негативна оцінки можуть стосуватися різних аспектів одного об'єкта. Якщо модель переважно орієнтується на найбільш емоційно насичені слова, вона може звести весь текст до одного домінантного сигналу. По-друге,

mixed може бути довшим за інші класи, оскільки авторові потрібно вмістити в повідомлення кілька змістових центрів. По-третє, у соціальних мережах амбівалентність часто виражається не формальною конструкцією «з одного боку, з іншого боку», а короткими розмовними протиставленнями, іронією, вставками іншою мовою або зміною стилю всередині повідомлення.

Окремою проблемою є межа між mixed і negative. У політичних, воєнних або кризових темах негативний компонент часто є лексично сильнішим за позитивний. Якщо текст містить підтримку певної дії, але значну частину повідомлення займає критика, модель може схилятися до negative. Зворотна помилка також можлива: коротке позитивне слово або емодзі на початку повідомлення може маскувати подальшу негативну оцінку. Тому для mixed недостатньо просто знайти слова з позитивною й негативною тональністю в одному тексті; потрібно оцінити, чи обидві полярності є змістовно релевантними для авторської позиції.

У межах цієї роботи клас mixed відіграє методологічну роль стрес-тесту для моделей. Якщо модель демонструє високу accuracy, але майже не розпізнає mixed, її практична якість для реального соціального дискурсу є обмеженою. Саме тому в інтерпретації результатів поряд із загальними метриками окремо аналізується F1 для класу mixed, а для ключових експериментів також подається F1 за класами positive, negative, neutral і mixed. Така логіка відповідає і прикладним потребам: у моніторингу суспільних настроїв амбівалентні повідомлення часто є особливо важливими, оскільки вони вказують не на просте схвалення чи засудження, а на конфліктні або перехідні оцінки аудиторії.

1.3 Код-змішування в обробці природної мови

Визначення код-змішування та його типи. Код-змішування є стратегією мовного спілкування, при якій мовець або автор використовує елементи двох або більше мов у межах одного комунікативного акту [23, 24]. Це явище широко поширене у білінгвальних та мультилінгвальних спільнотах по всьому світу.

У лінгвістиці розрізняють кілька типів код-змішування.

1. Inter-sentential code-switching (перемикання між реченнями): чергування фрагментів різних мов у різних реченнях. Наприклад: «Вчора ходив у кіно. Фильм был отличный, рекомендую всем».
2. Intra-sentential code-switching (змішування всередині речення): вставка лексичних елементів однієї мови в граматичну структуру іншої: «Я замовив этот товар и це було the best решение».
3. Tag-switching: вставка тегів, вигуків або фіксованих фраз іншою мовою. Наприклад: «Дякую за відповідь, но к сожалению це не те, що я шукав».
4. Congruent lexicalization (код-змішування інтеграційного типу): використання спільних лексичних елементів на стику мов: «Я пофіксив цей баг вчора» (інтеграція англійського слова з українською морфологією).

Суржик: лінгвістичні особливості. Суржик є специфічною формою код-змішування, що характерна для українського мовного простору і виникла внаслідок тривалого співіснування та взаємовпливу української та російської мов, зокрема в умовах історичного минулого. На відміну від «класичного» код-змішування, де мовець свідомо перемикається між системами, суржик є формою гібридизації («mixed language»). Елементи обох мов тісно інтегровані на усіх рівнях: фонетичному, морфологічному, лексичному та синтаксичному.

Основні лінгвістичні характеристики суржику охоплюють:

- лексичну інтерференцію, тобто використання російських лексем з українською морфологією (напр., «шо» замість «що», одночасне вживання «від» та «от» в одному тексті);
- морфологічний мікс: закінчення (флексії) однієї мови з основами (коренями) іншої, а також використання кальок синтаксичних побудов іншої мови;
- прагматичне перемикання: вживання стійких виразів чи вигуків іншою мовою заради експресивності.

У контексті соціальних мереж та Telegram-каналів суржик є надзвичайно поширеним явищем. Неформальний та швидкий стиль спілкування значно знижує самоцензуру, тож автори можуть навіть не усвідомлювати моменту перемикання між мовами.

1.3.1 Виклики код-змішування для NLP-систем

Код-змішані тексти створюють значні труднощі для систем обробки природної мови [24, 25]:

1. Ідентифікація мови на рівні токена: стандартні детектори мови працюють на рівні документу або речення, і класифікують код-змішаний текст як одну-єдину мову. Для коректної обробки необхідна ідентифікація мови на рівні окремих слів або n-грам.
2. Проблеми з токенізацією: підслівна токенізація (WordPiece, SentencePiece), навчена переважно на одномовних корпусах, часто некоректно розбиває слова і створює занадто довгі, неінтерпретовані послідовності субтокенів.
3. Семантичне представлення: мультилінгвальні моделі (mBERT, XLM-RoBERTa), хоча й знають обидві мови, зазвичай не мають

достатнього представлення саме змішаних контекстів, де семантика народжується на стику обох мов.

4. Зміна емоційного забарвлення на межі мов: у код-змішаних текстах сентиментний сигнал (емоція) може змінюватися разом зі зміною мови, особливо коли автор додає критику рідною чи експресивною мовою.

5. Обмежені ресурси: корпуси код-змішаних текстів, необхідні для fine-tuning, залишаються рідкістю. COSMUS є одним із перших масштабних ресурсів для українсько-російського напряму код-змішування.

Попередні дослідження код-змішаних текстів виокремлюють кілька напрямів подолання цих труднощів: визначення мови на рівні токенів або фрагментів, сегментацію код-змішаного повідомлення, використання багатомовних представлень, спеціальні бенчмарки для код-змішаних задач і аугментацію прикладів із контрольованими точками перемикання мов [16, 17]. Окремий напрям становлять підходи, які враховують мовну структуру під час навчання, наприклад поступове навчання або добір прикладів за часткою ресурсно багатої мови [26]. У цій роботі ці ідеї не копіюються буквально, а використовуються як мотивація для CSSent: модель має отримувати не тільки текст, а й додаткову інформацію про мовну структуру повідомлення.

У дослідженні [16] було запропоновано алгоритм, який використовує точки код-змішування у тексті та проводить аналіз тональності навколо них, перевершивши базову модель на іспансько-англійських даних на понад 11%. Це доводить, що врахування механіки код-змішування може бути дуже продуктивним.

Для українсько-російського код-змішаного середовища не слід розглядати лише як шум або технічну проблему токенізації. Перемикання між мовами може бути частиною прагматики висловлювання: автор може використовувати російські вставки для іронії, цитування, дистанціювання,

відтворення чужої позиції або стилістичного підсилення оцінки. У таких випадках мовна ознака стає додатковим сигналом для sentiment-моделі, оскільки тональність формується не тільки окремими словами, а й тим, як саме мови поєднуються в одному повідомленні.

Додатковим ускладненням є те, що суржик часто не має стабільної орфографічної норми. Один і той самий російсько-український елемент може бути записаний кількома способами, з помилками, фонетичною передачею або навмисною стилізацією. Для transformer-моделей це означає підвищену фрагментацію на субтокени й більшу залежність від контексту. Тому надмірна нормалізація таких текстів може прибрати саме ті ознаки, які несуть сентиментний сигнал.

1.3.2 Існуючі дослідження сентимент-аналізу для код-змішаних текстів

Сентимент-аналіз код-змішаних текстів активно досліджувався переважно для мовних пар у Південній Азії (Hinglish), іспансько-англійських (Spanglish) [17] та арабсько-англійських змішаних текстів [27]. У таких сценаріях код-змішування розглядається як окремий виклик для NLP: багатомовне переднавчання саме по собі не гарантує якісних репрезентацій для текстів із перемиканням мов [16]. Це пов’язано з тим, що в одному повідомленні поєднуються лексичні, морфологічні та прагматичні сигнали різних мов [25]. Для систематичного оцінювання таких задач було також запропоновано бенчмарк GLUECoS [28].

Для українсько-російської пари досліджень критично бракує. Праця [1] є ключовою, де систематично розглядається 4-класовий сентимент-аналіз на змішаних текстах з Telegram. У даному дослідженні, за аналогією до [1], поняття code-switching та code-mixing не розділяються і використовуються як

загальний термін «код-змішування». Використання їхнього датасету відкриває можливості для застосування сучасних україноцентричних моделей (MamayLM, Modern-LiBERTa) та встановлення нових результатів на таких специфічних текстах.

1.3.3 Мовна ознака у моделюванні код-змішування

Для задачі сентимент-аналізу мовна належність тексту може розглядатися на кількох рівнях. Найпростіший рівень – document-level language ID, коли все повідомлення позначається як українське, російське або змішане. Така ознака корисна для описової статистики, але вона втрачає внутрішню структуру повідомлення. У код-змішаному тексті значущим може бути не лише те, що в ньому є дві мови, а й те, де саме відбувається перемикання, які слова належать до кожної мови і чи збігається зміна мови зі зміною оцінного тону.

Другий рівень – token-level або phrase-level мовна структура. Він дозволяє побачити, що окремі російські, українські або суржикові фрагменти можуть виконувати різні функції. Частина вставок є технічно нейтральною: це назви, цитати або сталі фрази. Інші вставки мають експресивну функцію: автор може перейти на іншу мову для сарказму, імітації чужої позиції, підсилення критики або розмовного зниження стилю. Для sentiment-моделі це означає, що мовна ознака не є самостійною міткою тональності, але може змінювати спосіб інтерпретації лексичних маркерів.

Третій рівень – функціональний. На цьому рівні важливо не лише визначити, до якої мови належить фрагмент, а й зрозуміти його роль у висловлюванні. Наприклад, російський фрагмент може бути прямою цитатою, і тоді сентимент автора може бути спрямований не на зміст цитати, а на сам факт її використання. Український фрагмент може містити оцінку, а

російський – об'єкт критики. У коротких Telegram-коментарях межа між авторською оцінкою, цитатою, сарказмом і реакцією на попередній контекст часто не позначена явно. Тому модель має робити висновок не за окремим словом, а за поєднанням лексичних, мовних і позиційних сигналів у всьому повідомленні.

У цій роботі мовною ознакою називається інформація про те, до якої мовної категорії належить повідомлення або його фрагмент: української, російської чи змішаної. Така ознака не є самостійним індикатором тональності: наявність російських слів не означає негативний клас, а наявність українських слів не означає позитивний клас. Її роль інша – допомогти моделі зберегти інформацію про мовну структуру повідомлення та використати її разом із лексичними й позиційними ознаками.

Саме тому в CSSent мовна ознака використовується двома способами. По-перше, допоміжна задача визначення мови регуляризує encoder і змушує його зберігати інформацію про мовну належність тексту. По-друге, мовно-зумовлена класифікаційна голова передає цю ознаку до частини моделі, яка прогнозує клас тональності. Така схема не прив'язує sentiment-клас до мови, а надає моделі додатковий контекст для інтерпретації змішаного повідомлення.

З цієї ж причини в роботі обрано обережний препроцесинг. Надмірне виправлення суржику, автоматичний переклад або нормалізація до однієї літературної мови могли б зменшити шум для токенізатора, але водночас стерти ознаки перемикання мов, стилізації та прагматичного контрасту. Тому в роботі зберігаються суржикові форми, емодзі, розмовна орфографія та частина нестандартних конструкцій.

1.4 Попередньо треновані моделі для української мови

1.4.1 Мультилінгвальні encoder-моделі

Для задачі класифікації код-змішаних текстів мультилінгвальні моделі пропонують значну перевагу у здатності до крос-лінгвального трансферу, оскільки вони натреновані на великих корпусах одночасно кількома мовами.

mBERT (Multilingual BERT) є базовою архітектурою від Google [8], яка тренувалася на текстах Вікіпедії 104 мовами. Модель містить ~178M параметрів. Для української мови mBERT має суттєві обмеження: словник (119K токенів) фрагментує українські слова на 4-6 субтокенів. До того ж, «curse of multilinguality» знижує загальну якість представлень через обмежену ємність моделі для розподілу по 100+ мовах. Тексти Вікіпедії надто відмінні від сленгу та суржику соціальних мереж.

XLM-RoBERTa (XLM-R) є вдосконаленою архітектурою від Meta [6], яка покращила mBERT шляхом використання більшого словника (250K токенів SentencePiece), що краще оптимізовано для різних писемностей, та значно більшого тренувального корпусу (2.5 TB CommonCrawl замість корпусу Вікіпедії). Для української мови важливо, що XLM-R спирається не лише на енциклопедичні тексти, а й на ширший веб-корпус, тому ця модель стала однією з найуспішніших мультилінгвальних основ для задач розуміння природної мови (Natural Language Understanding, NLU), тобто задач, де модель має інтерпретувати зміст тексту, а не генерувати новий текст

1.4.2 Україноцентричні encoder-моделі

Розвиток українських корпоративних та відкритих ініціатив зумовив появу моделей, спеціально тренованих на українських текстах.

UkrRoBERTa є encoder-моделлю, навченою переважно на українських текстах (~60GB) [9]. Має SentencePiece словник на 52 000 токенів, оптимізований саме для української морфології. Її перевагою є низька фрагментація слів (близько 1.2–1.5 токена на слово). Корпус включає тексти з різних доменів інтернет-контенту, близьких за стилем до соціальних мереж, при цьому маючи компактний розмір (~125M параметрів).

Modern-LiBERTa Large є найновішою масштабною encoder-моделлю на базі архітектури ModernBERT, тренована на великому українському корпусі "Kobza" (близько 60 млрд токенів). Модель містить приблизно 410M параметрів та включає низку архітектурних інновацій:

- RoPE (Rotary Position Embeddings): метод кодування позицій токенів через обертання векторів, що дозволяє ефективно обробляти послідовності до 8192 токенів;
- чергування local та global attention для балансу між обробкою локального контексту та глобальних залежностей;
- GeGLU (Gated GELU Linear Unit): вдосконалена функція активації, що поєднує механізм гейтування з нелінійністю GELU для покращення потоку інформації;
- Flash Attention 2: оптимізований алгоритм обчислення уваги, що суттєво зменшує споживання пам'яті та прискорює обчислення.

Цей набір дозволяє ефективно і глибоко розуміти українські тексти, роблячи її наймасштабнішою україноцентричною encoder-моделлю 2025 року.

1.4.3 Decoder-модель MamayLM

MamayLM v1.0 є однією з перших масштабних відкритих україноцентричних великих мовних моделей (decoder-only LLM), розроблених в інституті INSAIT. Базуючись на архітектурі Gemma 3 від Google DeepMind, MamayLM адаптована під специфіку української мови та даних різних доменів.

Модель доступна у розмірах 4B та 12B параметрів. У цьому дослідженні MamayLM оцінено у трьох режимах: prompting для 4B, prompting для 12B, а також параметрично-ефективне дотренування 4B і 12B через QLoRA. Завдяки 4-бітній квантизації NF4 inference цієї моделі може виконуватися на споживчих відеокартах із 8+ GB VRAM. При цьому QLoRA-дотренування в роботі виконувалося на потужніших GPU, зокрема A100 40 GB для 4B і RTX PRO 6000 з 102 GB VRAM для 12B.

Додаткове тренування базової Gemma 3 на українських текстах дало MamayLM кращі показники на вітчизняних бенчмарках. У контексті задач класифікації ця LLM здатна вирішувати їх у режимі zero-shot і few-shot (без оновлення ваг) через спеціальні інструктивні промпти.

1.4.4 Порівняльний аналіз моделей для українського сентимент-аналізу

Різнорідні моделі демонструють різні переваги:

1. Мультилінгвальні моделі (XLM-R Base, XLM-R Large) мають сильну здатність вловлювати крос-лінгвальні зв'язки, які можуть бути критично важливими для код-змішаних текстів (UA-RU).

2. Україноцентричні енкодери (UkrRoBERTa, Modern-LiBERTa Large) краще розпізнають типові українські структури, сленг та морфологічні варіації, характерні для месенджерів.

3. Великі decoder-only моделі MamayLM 4B і MamayLM 12B пропонують два різні сценарії використання: інструктивний промптинг без оновлення ваг і QLoRA-дотренування на цільовому датасеті. Це дає змогу перевірити, чи здатна відкрита україноцентрична генеративна модель працювати як стабільний класифікатор тональності, а також наскільки спеціалізоване дотренування змінює її якість порівняно з простим prompting.

Дослідження NLU-бенчмарків вказують на перевагу україноцентричних моделей на "чистих" українських текстах. Проте на специфічних даних з російською та код-змішуванням ця перевага може нівелюватись, що робить порівняльний експеримент необхідним для встановлення кращої архітектури.

1.4.5 Практичний компроміс між локальними моделями та API LLM

Великі комерційні LLM, такі як Gemini 3.0 Flash, можуть демонструвати сильну якість навіть без дотренування на цільовому корпусі, але така перевага супроводжується залежністю від зовнішнього сервісу, вартості API-запитів, обмежень доступу, зміни версій моделі та меншого контролю над протоколом інференсу.

Локальні encoder-моделі мають іншу природу компромісу. Вони значно компактніші, не мають широкої генеративної здатності, але для класифікаційної задачі це не обов'язково є недоліком. Після дотренування encoder працює як спеціалізований класифікатор: він швидший, дешевший у масовому інференсі, відтворюваніший і контрольованіший. Для моніторингу

великих масивів коротких повідомлень ці властивості можуть бути не менш важливими, ніж невелика різниця в Macro F1.

MamayLM займає проміжну позицію. Це локальна decoder-only LLM, тому її можна порівнювати і з API LLM, і з локальними класифікаторами, але її роль у роботі окрема: вона показує, наскільки відкрита україноцентрична генеративна модель може бути адаптована до класифікації через промптинг або QLoRA.

Отже, практичний компроміс у цій роботі полягає не лише у виборі конкретної моделі, а й у виборі режиму її використання. API LLM виконують роль сильного зовнішнього орієнтира, локальні encoder-моделі – роль контрольованих класифікаторів для масового інференсу, а MamayLM оцінюється як відкрита генеративна модель у режимах промптингу та QLoRA-дотренування.

1.5 Методи адаптації мовних моделей

1.5.1 Повне дотренування (Full Fine-Tuning)

Класичний підхід до адаптації попередньо натренованих мовних моделей полягає у повному оновленні всіх параметрів моделі (Full Fine-Tuning) на цільовому розміченому датасеті (downstream task).

Для encoder-моделей (BERT, RoBERTa, ModernBERT) адаптація передбачає додавання класифікаційної голови (classification head) поверх останнього прихованого шару. Зазвичай це один або два лінійні шари з dropout-регуляризацією:

$$\hat{y} = \mathrm{softmax}\big(W_2 \cdot \mathrm{ReLU}\big(W_1 \cdot h_{[CLS]} + b_1\big) + b_2\big),$$

де $h_{[CLS]}$ – векторне представлення спеціального токена [CLS], яке акумулює семантичну інформацію про всю вхідну послідовність. W та b – ваги та зсуви відповідних повнозв'язних шарів.

Під час процесу fine-tuning оновлюються як параметри доданої класифікаційної голови, так і параметри самого енкодера. Усі параметри моделі тренуються одночасно з єдиною швидкістю навчання, яка для encoder-моделей зазвичай перебуває в діапазоні $1–5\times10^{-5}$ і обирається з урахуванням архітектури, розміру моделі та стабільності валідаційної метрики. Для досягнення стабільного навчання та уникнення "катастрофічного забування" (catastrophic forgetting), коли модель повністю втрачає попередньо набуті під час pre-training знання, застосовуються наступні практики [29]:

- низький learning rate (порядку $1–5\times10^{-5}$), що дозволяє зберігати попередньо набуті знання;
- Warmup: поступове збільшення learning rate від нуля до встановленого максимуму впродовж перших 10% кроків навчання (warmup_ratio=0.1);
- Linear scheduler для плавного зменшення learning rate після пікового значення;
- Weight decay (0.01) для L2-регуляризації.

Особливою загрозою для невеликих датасетів (<10K зразків), як датасет COSMUS (~12K), є перенавчання (overfitting). Тому критично важливим є застосування ранньої зупинки (early stopping) за метрикою validation macro F1 та посилення dropout до значень 0.2–0.3.

1.5.2 Prompt Engineering

Промпт-інженерія (prompt engineering) є способом адаптації decoder-only LLM без жодної структурної чи параметричної модифікації моделі. Замість навчання ваг, задача формулюється як текстовий запит (prompt).

***Zero-Shot Prompting*:**

"Визначте тональність наступного тексту.

Класи: positive, negative, neutral, mixed.

Текст: "Сьогодні жахливий день"

Тональність:"

Такий метод не потребує розмічених даних, і модель спирається на знання та інструктивні навички, набуті під час попереднього навчання. Проте для моделей до 10B параметрів цей метод страждає нижчою стабільністю відповідей і частою зміною формату (генерація зайвих коментарів тощо).

***Few-Shot Prompting*:**

Few-shot prompting зазвичай є ефективнішим за zero-shot підхід: до інструкції додаються кілька готових прикладів, зазвичай від 1 до 5 на клас, які демонструють моделі бажаний шаблон відповіді та вихідний формат [30]. У цьому випадку:

" Визначте тональність тексту.

Приклад 1: "Чудова погода!" → positive

Приклад 2: "Жахливий сервіс" → negative

Текст: "Ну, нормально, нічого особливого"

Тональність:"

Особливістю MamayLM є здатність використовувати few-shot приклади для досягнення цільової точності без жодного оновлення параметрів, проте модель все ще чутлива до дисбалансу класів та суб'єктивності вибору прикладів для промпту. Загальна точність (accuracy) промптингу на складних завданнях класифікації зазвичай поступається повноцінному down-stream

навчанню [31], що робить дотренування кращим вибором там, де доступний розмічений датасет.

1.5.3 Параметрично-ефективне дотренування (LoRA / QLoRA)

Альтернативою повному дотренуванню є метод LoRA (Low-Rank Adaptation), запропонований [32], який замість оновлення всіх параметрів моделі додає компактні низькорангові матриці до шарів уваги. Це дозволяє дотренувати лише невелику частку параметрів (зазвичай менше 1%), зберігаючи при цьому якість, близьку до повного fine-tuning. Подальшим розвитком цього підходу є QLoRA [33], що поєднує LoRA з 4-бітною квантизацією базової моделі (NF4), значно зменшуючи вимоги до пам'яті порівняно з повним дотренуванням LLM. У цьому дослідженні QLoRA застосовано для MamayLM 4B та MamayLM 12B з однаковою схемою LoRA r=32, alpha=64, effective batch size 16, learning rate 0.0002 і тривалістю 3 епохи. Практично важливо розрізняти інференс і навчання: 4-бітне завантаження моделі може вимагати порівняно небагато пам'яті, однак фактичне дотренування у роботі виконувалося на високопродуктивних GPU.

1.5.4 Постановка задачі адаптації моделей

Вибір способу адаптації мовної моделі залежить від того, яку архітектуру використано і яку практичну мету має система. Для encoder-моделей природним рішенням є повне дотренування з класифікаційною головою, оскільки такі моделі спеціально призначені для задач розуміння тексту. Вони не генерують відповідь послідовно, а формують компактне

представлення вхідного тексту, яке можна напряму оптимізувати під чотири класи тональності. Тому в умовах наявного розміченого корпусу COSMUS full fine-tuning є базовим і найпрямішим методом адаптації encoder.

Prompting має іншу логіку. Він зручний там, де немає можливості навчати модель або коли потрібно швидко отримати baseline. Для комерційних LLM prompting часто дає сильні результати через масштаб моделі та її instruction-tuning. Проте prompting менш контрольований: модель може порушувати формат відповіді, по-різному реагувати на мову інструкції, бути чутливою до складу few-shot прикладів і змінювати поведінку після оновлення API. Для задачі з чотирма класами, де mixed має малий support і складну семантику, prompting без навчання не завжди достатній.

QLoRA займає проміжну позицію між промптингом і повним дотренуванням. Вона дозволяє адаптувати decoder-only LLM до конкретного корпусу, не оновлюючи всі параметри моделі. Для задачі сентимент-аналізу це важливо тому, що генеративна модель має бути перевірена не лише як джерело текстових відповідей, а і як стабільний класифікатор із контрольованою схемою міток.

Загальна задача дослідження полягає у розробленні та експериментальній перевірці комплексного підходу до мультикласового сентимент-аналізу українсько-російських код-змішаних текстів соціальних мереж, який забезпечує надійне розпізнавання чотирьох класів тональності (positive, negative, neutral, mixed) в умовах мовної неоднорідності повідомлень, дисбалансу класів і складності амбівалентної категорії mixed.

Для розв'язання цієї задачі необхідно вирішити такі підзадачі:

- сформувати єдиний експериментальний протокол із чотирма класами тональності, фіксованими розбиттями COSMUS і COSMUS+ та основною метрикою Macro F1;
- розширити корпус COSMUS до COSMUS+ для перевірки стійкості моделей на ширшому розподілі Telegram-повідомлень і

збільшення абсолютної кількості прикладів міноритарного класу mixed;

- розробити для encoder-моделей комплексну схему навчання, яка одночасно враховує мовну ознаку повідомлення (мовно-зумовлена класифікаційна голова та допоміжна задача визначення мови), дисбаланс і складність класу mixed (зважена фокусна функція втрат), знання сильнішої моделі-вчителя (дистиляція м'яких міток) та код-змішану природу даних (аугментація CSPI/SPP);
- для decoder-only моделей перевірити, чи достатньо інструктивного промптингу, чи необхідне параметрично-ефективне дотренування QLoRA;
- виконати порівняльне оцінювання комерційних API LLM, локальних encoder-моделей, запропонованого пайплайну CSSent і MamayLM за загальними метриками, F1 класу mixed, калібруванням ймовірностей та LIME/SHAP-пояснюваністю.

Початковий COSMUS містить лише 608 mixed-прикладів, тому для перевірки стійкості моделей на ширшому розподілі Telegram-повідомлень у роботі формується розширення COSMUS+.

На основі цієї постановки в розділі 2 запропоновано комплексний підхід CSSent для encoder-моделей, а в експериментальному розділі перевірено окремі режими адаптації для API LLM, локальних encoder-моделей і MamayLM.

Висновки до розділу 1

У цьому розділі було розглянуто теоретичні передумови та основи проведення дослідження сентимент-аналізу для код-змішаних текстів в українському цифровому просторі. Встановлено, що класифікація

тональності пройшла шлях від ранніх словникових (лексичних) підходів, крізь класичні алгоритми машинного навчання (SVM) та глибокі рекурентні мережі (LSTM/CNN), до сучасних архітектур на базі трансформерів, що домінують у сучасній парадигмі NLP.

Незважаючи на високу результативність аналізу 2-класових англомовних текстів, мультикласова (зокрема, 4-класова: positive, negative, neutral, mixed) класифікація залишається відкритою проблемою для мов із обмеженими ресурсами. Головною перешкодою є наявність сильного дисбалансу класів (у COSMUS: mixed становить лише ~5% зразків), що вимагає специфічної обробки та зважених підходів до метрики (Macro F1-score замість Accuracy) і функції втрат.

Українсько-російський суржик або код-змішування ще більше ускладнює задачу для класифікаторів: нестандартна токенізація, фрагментація слів на велику кількість субтокенів, семантично-емоційна зміна напрямку (сентименту) на стику двох мов та загальний дефіцит спеціалізованих моделей.

Було проведено детальний огляд архітектур для поточної задачі. Моделі поділяються на мультилінгвальні encoder-моделі (mBERT, XLM-RoBERTa), які краще працюють із крос-лінгвальним переходом між мовами, україноцентричні encoder-моделі (UkrRoBERTa, Modern-LiBERTa), які натреновані для кращого розуміння саме українських контекстів і мають мінімальну фрагментацію, та decoder-only LLM-архітектури (наприклад, MamayLM 4B), розроблені для генеративних та інструктивних завдань.

Сформульовано постановку задачі подальшого дослідження: українсько-російський код-змішаний сентимент-аналіз потребує перевірки не лише окремих архітектур, а й способів їх адаптації до мовної неоднорідності, дисбалансу класів і категорії mixed. З цієї постановки випливає потреба у комплексному підході для encoder-моделей, а також в окремому експериментальному порівнянні промптингу та QLoRA для відкритої decoder-only моделі MamayLM , а також у розширенні COSMUS до

COSMUS+ для перевірки стійкості моделей на ширшому розподілі Telegram-повідомлень.

Актуальні дослідження [1, 11, 21, 22] підкреслюють потребу у систематичному порівнянні encoder- та decoder-архітектур на специфічних текстах соціальних мереж та Telegram для досягнення стабільної та якісної системи аналізу. Саме це порівняння різнорідних архітектур і лежить в основі подальших розділів цієї роботи.

РОЗДІЛ 2 МАТЕМАТИЧНИЙ АПАРАТ, КОРПУСНА БАЗА ТА ЗАПРОПОНОВАНИЙ КОМПЛЕКСНИЙ ПІДХІД ДО АНАЛІЗУ КОД-ЗМІШАНИХ ТЕКСТІВ

2.1 Математичні основи архітектури трансформера

Архітектура Transformer, запропонована у революційній роботі «Attention Is All You Need» [34], докорінно змінила підхід до обробки природної мови і стала фундаментальною основою сучасних мовних моделей. На відміну від рекурентних нейронних мереж (таких як RNN, LSTM або GRU), які обробляють текст послідовно токен за токеном, трансформер працює на основі механізму само-уваги (Self-Attention). Це дозволяє йому паралельно обробляти всі позиції вхідної послідовності та ефективно моделювати довгі контекстні залежності.

Оригінальний Transformer складається з двох основних блоків.

1. Encoder (кодувальник): приймає вхідну послідовність токенів та формує їхні глибоко контекстуалізовані представлення. Кожний шар encoder містить два головні елементи (Multi-Head Self-Attention та Feed-Forward Network (FFN)), які з'єднані через залишкові зв'язки (residual connections) та нормалізацію по шарах (Layer Normalization) для забезпечення стабільного проходження градієнтів під час навчання.
2. Decoder (декодувальник): застосовується для генеративних задач і формує вихідну послідовність авторегресійно. Окрім стандартних self-attention та FFN, decoder містить Masked Self-Attention (маскування майбутніх позицій, щоб прогнозувати наступний токен) та Cross-Attention для зв'язку з представленнями від encoder.

У задачах класифікації тональності (сентимент-аналіз) часто використовується лише Encoder завдяки його здатності формувати двонаправлені контекстні зв'язки (наприклад, моделі BERT, XLM-RoBERTa, Modern-LiBERTa).

Механізм Self-Attention. Основною інновацією моделі є механізм Self-Attention. Він дозволяє моделі під час обробки певного токена «зважувати» вплив усіх інших токенів у послідовності на цей конкретний токен.

Технічно, вхідний вектор кожного токена x_i за допомогою трьох матриць ваг W^Q, W^K та W^V перетворюється у три нові вектори:

- Query (q_i): що поточний токен "запитує" або шукає в інших;
- Key (k_i): що інші токени репрезентують або "пропонують";
- Value (v_i): фактичне змістове представлення токена, яке буде агреговано.

Вагові коефіцієнти уваги та фінальне представлення розраховуються як нормалізований скалярний добуток:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V,$$

де d_k – розмірність векторів ключів, а поділ на $\sqrt{d_k}$ запобігає насиченню функції softmax, яке може призвести до зникнення градієнтів.

Multi-Head Attention. Замість єдиного розрахунку уваги (single-head), трансформер використовує механізм багатоголової уваги – Multi-Head Attention. Модель виконує h незалежних паралельних проєкцій запитів, ключів і значень (голів), кожна з яких навчається фокусуватися на різних типах взаємозв'язків між словами: синтаксичних, семантичних, локальних чи глобальних:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O,$$
$$\text{head}_i = \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right),$$

де W^O – фінальна матриця лінійної проєкції. Наприклад, класичний BERT-base використовує 12 голів уваги.

Позиційне кодування (Positional Encoding). Оскільки механізм Self-Attention сам по собі обробляє слова як "мішок слів" без розуміння порядку їх слідування (permutation-invariant), моделі необхідно явно передавати інформацію про позицію кожного токена.

В оригінальному трансформері застосовувались синусоїдальні та косинусоїдальні функції:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d}}\right),$$
$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d}}\right).$$

Проте, найсучасніші архітектури (наприклад, Modern-LiBERTa та MamayLM), що досліджуються в цій роботі, використовують більш вдосконалений підхід: Rotary Position Embeddings (RoPE). Цей метод кодує відносні позиції токенів шляхом обертання їхніх векторів Query та Key, що дозволяє значно краще екстраполювати увагу на послідовності довільної довжини.

2.2 Функції втрат та метрики оцінки

2.2.1 Функції втрат (Loss Functions)

Для навчання моделей-класифікаторів критичним є вибір функції втрат (loss function), що мінімізується під час алгоритму зворотного поширення помилки (backpropagation).

Cross-Entropy Loss (крос-ентропія). Стандартною функцією для задачі мультикласової класифікації є крос-ентропія:

$$\mathcal{L}_{\mathcal{CE}} = -\sum_{c=1}^{C} y_c \log(\widehat{p_c}),$$

де y_c – істинна мітка класу у форматі one-hot, а $\widehat{p_c}$ – передбачена моделлю ймовірність приналежності до класу c. Проте в умовах сильного дисбалансу класів (такого як у датасеті COSMUS, де клас mixed становить ~5%), стандартна крос-ентропія призводить до ігнорування міноритарних класів, адже глобальний мінімум втрат легко досягається шляхом точного передбачення мажоритарного класу.

Weighted Cross-Entropy (Зважена крос-ентропія). Щоб подолати вплив дисбалансу класів під час дотренування трансформерів, у даному дослідженні застосовується зважена крос-ентропійна функція втрат. Кожному класу c присвоюється вага w_c, яка є обернено пропорційною до його частоти у тренувальній вибірці:

$$\mathcal{L}_{WCE} = -\sum_{c=1}^{C} w_c \cdot y_c \log(\widehat{p_c}), \quad w_c = \frac{N}{C \cdot N_c},$$

де N – загальний обсяг тренувальної вибірки, C – кількість класів (4), N_c – кількість зразків класу c. Використання w_c "штрафує" модель сильніше за помилки на менш чисельних класах (наприклад, mixed або positive), змушуючи її вчити їхні відмінні ознаки.

Weighted Focal loss. У CSSent також використовується weighted focal loss, що є модифікацією крос-ентропії для задач із дисбалансом і великою кількістю легких прикладів [5]. Для одного прикладу позначимо через (p_y) імовірність, яку модель надала правильному класу (y), а через (w_y) – вагу цього класу. Тоді weighted focal loss має вигляд:

$$\mathcal{L}_{\mathrm{F}} = -w_y\left(1 - p_y\right)^{\gamma} \log\left(p_y\right),$$

де ($\gamma \geq 0$) є параметром фокусування. Якщо ($\gamma = 0$), focal loss зводиться до зваженої крос-ентропії. Якщо ($\gamma > 0$), множник ($\left(1 - p_y\right)^{\gamma}$) зменшує внесок

легких прикладів, для яких модель уже надає правильному класу високу ймовірність. Натомість складні приклади з малим (p_y) зберігають більший внесок у функцію втрат. У контексті COSMUS це корисно насамперед для класу mixed, оскільки він рідкісний і часто програє мажоритарним класам під час оптимізації стандартної крос-ентропії.

Для мультикласового випадку, якщо (y_c) є one-hot індикатором належності до класу (c), формулу можна записати так:

$$\mathcal{L}\text{FL} = -\sum c = 1^{C} w_c y_c (1 - \widehat{p_c})^{\gamma} \log(\widehat{p_c}),$$

де ($C = 4$) для класів positive, negative, neutral і mixed, (w_c) - вага класу, а ($\widehat{p_c}$) – прогнозована ймовірність класу (c). У роботі focal loss використовується не як самостійний універсальний розв'язок проблеми дисбалансу, а як один із елементів CSSent, ефект якого перевіряється через абляційний аналіз (ablation study).

2.2.2 Метрики оцінки продуктивності

Для оцінювання результативності моделей на етапі тестування використовуються кілька ключових метрик:

Accuracy (загальна точність). Accuracy показує частку правильних прогнозів серед усіх прикладів і є корисною для загального уявлення про роботу класифікатора. Проте для COSMUS вона не може бути основною метрикою, оскільки класи розподілені нерівномірно: у тестовій вибірці клас mixed має лише 60 прикладів із 1 223. Модель може отримати прийнятну accuracy, правильно класифікуючи переважно negative і neutral, але при цьому майже не розпізнавати mixed. Тому основною метрикою ранжування

моделей у роботі є Macro F1, яка надає однакову вагу кожному класу незалежно від представленості цього класу у вибірці.

Precision, Recall та F1-Score (Per-Class). Ці метрики обчислюються ізольовано для кожного класу:

- Precision показує, яка частка зразків, віднесених алгоритмом до класу c, дійсно йому належить:

$$P_c = \frac{TP_c}{TP_c + FP_c};$$

- Recall (повнота) ілюструє, яку частину зразків класу c модель змогла успішно знайти:

$$R_c = \frac{TP_c}{TP_c + FN_c};$$

- F1-score є гармонійним середнім між Precision і Recall, що дає компромісну збалансовану оцінку:

$$F1_c = \frac{2P_cR_c}{P_c + R_c}.$$

Macro F1-Score (Основна метрика)

З огляду на дисбаланс, для об'єктивного порівняння загальної якості моделей застосовано метрику Macro F1. Вона дорівнює простому середньому арифметичному всіх per-class F1 оцінок:

$$F1_{macro} = \frac{1}{C}\sum_{c=1}^{C} F1_c$$

Оскільки кожен клас (навіть найменший) вносить рівну частку до $F1_{macro}$, ця метрика безпосередньо висвітлює ефективність моделі у розпізнаванні міноритарних категорій, таких як mixed.

2.3 Датасет COSMUS

В рамках даного дослідження використовується відкритий датасет COSMUS (COde-Switched MUltilingual Sentiment for Ukrainian Social media). Він є першим спеціалізованим публічним корпусом для мультикласового сентимент-аналізу українськомовного та російськомовного контенту, а також українсько-російського код-змішування (суржику) у цифровому просторі.

Датасет складається із 12 224 текстових зразків, які збиралися з Telegram-каналів, сайтів з відгуками на товари та інших відкритих джерел. Кожен текст у датасеті анотований за двома ключовими параметрами.

Тональність за 4-класовою шкалою:

- positive: повідомлення виражають позитивні емоції, задоволення, підтримку, радість;
- negative: тексти виражають незадоволення, критику, розчарування або тривогу;
- neutral: інформативно-фактологічні повідомлення без жодного відчутного емоційного забарвлення;
- mixed: висловлювання, що містять водночас і позитивні, і негативні оцінки стосовно одного або кількох об'єктів (наприклад: "Сам фільм непоганий, але акторська гра дуже слабка").

Мовна належність:

- uk: переважно українська мова;
- ru: переважно російська мова;

- mixed (або код-змішування): суміш української та російської, тобто наявність суржику або перемикання кодів всередині тексту.

Статистика та проблема дисбалансу класів. Розподіл класів тональності у COSMUS є нерівномірним: neutral і negative разом становлять понад три чверті корпусу, тоді як mixed охоплює лише 608 текстів, тобто 5,0% вибірки. Через такий розподіл Accuracy не відображає однаково якість розпізнавання всіх класів, тому основною метрикою обрано Macro F1, а результати для mixed аналізуються окремо (рис. 2.1).

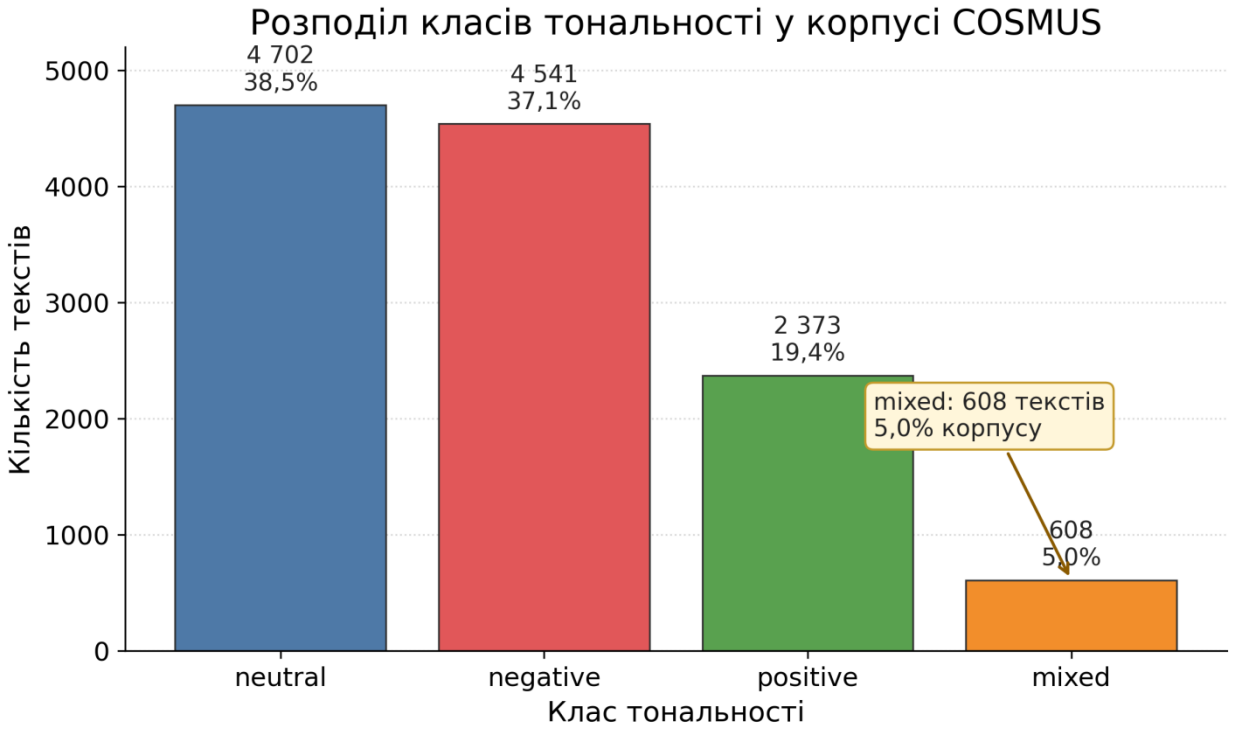

Рисунок 2.1 – Розподіл класів тональності у корпусі COSMUS

Коефіцієнт дисбалансу, тобто відношення найбільшого класу до найменшого, становить приблизно 7,7:1. Без компенсаційних механізмів нейронні мережі можуть зміщуватися до мажоритарних класів і суттєво недооцінювати категорію mixed. Для врахування цієї особливості під час навчання використовується функція втрат із ваговими коефіцієнтами класів.

Поділ датасету (Stratified Splitting). Для коректного проведення експериментів і об'єктивного оцінювання здатності моделей до узагальнення використано готове статичне розбиття df_set, наявне в COSMUS. Воно зберігає порівнюваність з оригінальним бенчмарком і не формувалося випадково заново.

1. Тренувальна вибірка (Train): 9 779 зразків, приблизно 80,0% корпусу. Використовується для навчання ваг encoder-класифікаторів і для добору few-shot прикладів генеративним моделям.
2. Валідаційна вибірка (Validation): 1 222 зразки, приблизно 10,0%. Застосовується після кожної епохи навчання для моніторингу процесу, ранньої зупинки та вибору гіперпараметрів.
3. Тестова вибірка (Test): 1 223 зразки, приблизно 10,0%. Тримається невидимою під час навчання та використовується для фінальної оцінки усіх моделей.

2.4 Збір та попередня обробка даних

Месенджер Telegram обрано як основне джерело збирання додаткових зразків для розширення оригінального датасету COSMUS. Основні аргументи на користь Telegram представлено нижче.

1. Широке охоплення і висока роль у новинному споживанні: Telegram є одним з основних цифрових майданчиків для оперативного поширення новин, реакцій і коментарів в Україні. Саме тому тексти з Telegram придатні для вивчення сучасного суспільного дискурсу, зокрема коротких емоційних повідомлень, коментарів, суржику та українсько-російського код-змішування.
2. Жанрове і мовне різноманіття: Дозволяє збирати новини, суспільно-політичну аналітику, розважальний контент. Мовна

практика Telegram-співтовариств характеризується органічним перемиканням кодів та вживанням суржику в межах одного повідомлення.

3. Емоційна насиченість: Порівняно з формальними джерелами (наприклад, Вікіпедією), Telegram-канали та чати містять широкий спектр тональностей, що критично важливо для навчання моделей сентимент-аналізу.

Розширення корпусу COSMUS мотивовано потребою перевірити стійкість моделей на ширшому розподілі Telegram-повідомлень, збільшити абсолютну кількість прикладів міноритарного класу mixed і наблизити експеримент до реального потоку соціальних текстів.

Збір даних здійснюється з використанням асинхронного Python-клієнта Telethon. Розроблений скрипт завантажує текстові повідомлення та коментарі під постами з відібраного списку Telegram-каналів. Процедура скрапінгу обходить системні повідомлення (приєднання/від'єднання) та повідомлення без текстового супроводу (лише посилання чи медіа). Отримані дані зберігаються у Parquet-файлах.

Для автоматичної попередньої розмітки розширеного датасету використовуються API великих мовних моделей (DeepSeek V3.2, Gemini 3.0 Flash, Kimi K2.5).

2.4.1 Формування розширеного корпусу COSMUS+

Розширення COSMUS виконано для перевірки стійкості моделей на новіших Telegram-даних і для збільшення абсолютної кількості прикладів міноритарного класу mixed. Початковий масив нових повідомлень і коментарів містив 23 006 рядків. Після вилучення некоректних міток, спаму, надто коротких повідомлень, точних і наближених дублікатів, а також

перетинів із початковим COSMUS залишено 21 045 нових очищених прикладів. Після об'єднання з 12 222 рядками початкового COSMUS підсумковий COSMUS+ містить 33 267 текстів.

Формування COSMUS+ включало кілька рівнів фільтрації. На першому рівні вилучалися рядки з некоректними sentiment або language label, оскільки такі записи не можуть бути використані для контрольованого навчання. На другому рівні видалялися повідомлення, які Gemini позначила як spam або непридатні для розмітки. На третьому рівні застосовувалися rule-based фільтри: вилучення порожніх або надто коротких повідомлень, URL-only рядків, текстів із низькою часткою алфавітних символів, а також очевидного рекламного, казино, betting, crypto або giveaway шуму. На четвертому рівні контролювалися точні дублікати, наближені дублікати й перетин із початковим COSMUS.

У sentiment-розподілі COSMUS+ найбільшим класом є negative (16 981 приклад), далі йдуть neutral (8 813), positive (5 926) і mixed (1 547). Частка mixed лишається малою, близько 4,65%, тому розширення корпусу не усуває проблему міноритарного класу, але збільшує абсолютну кількість відповідних прикладів.

Якість автоматичної розмітки перевірено на ручній вибірці зі 100 прикладів. Для sentiment agreement отримано accuracy 0,7400 і Cohen's kappa 0,5775, що є прийнятним для шумного соціального корпусу, але не означає безпомилкової розмітки. Для language agreement результат вищий: accuracy 0,9200 і Cohen's kappa 0,8617. Отже, COSMUS+ використовується як розширений корпус для додаткової перевірки моделей, а не як повністю ручний золотий стандарт.

2.4.2 Етичні аспекти роботи з Telegram-даними

У межах цієї дисертації Telegram використовується як джерело відкритого соціального дискурсу, а не як джерело персональних профілів. Основною одиницею аналізу є текстове повідомлення або коментар, а не особа автора. Тому під час формування корпусу акцент робиться на вилученні або ігноруванні зайвих ідентифікаційних ознак, збереженні лише того текстового матеріалу, який потрібен для задачі sentiment classification.

З практичного погляду це означає, що модель не навчається передбачати політичні погляди конкретної людини, її соціальний статус або інші персональні характеристики. Завдання обмежене класифікацією тональності окремого тексту за чотирма класами: positive, negative, neutral і mixed. Така постановка звужує ризик надмірної інтерпретації: результат моделі не повинен трактуватися як психологічний портрет автора, а лише як оцінка тонального забарвлення конкретного повідомлення в межах заданого корпусу.

Окремим ризиком є тематична чутливість Telegram-даних. Український Telegram-дискурс часто містить повідомлення про війну, безпеку, політичні рішення, економічні труднощі, втрати й суспільні конфлікти. У таких темах тональність може бути різкою, а межа між factual negative content і negative sentiment є складною. Наприклад, повідомлення про негативну подію не завжди є негативною оцінкою автора. Саме тому в роботі розмежовано neutral як фактологічний клас і negative як клас із вираженою оцінкою, а якісний розбір показових випадків використовується для перевірки типових помилок моделей.

Важливим є також те, що зібрані дані проходять фільтрацію спаму, рекламного шуму, URL-only повідомлень, дублікатів і явно непридатних рядків. Це потрібно не лише для покращення метрик, а й для коректності самого дослідження. Якщо модель навчається на великій кількості рекламних

або технічно порожніх повідомлень, вона може почати розпізнавати не sentiment, а структурні ознаки шуму. Тому очищення даних у COSMUS+ є частиною наукового протоколу, а не допоміжною технічною операцією.

Ще один аспект пов'язаний з автоматичною розміткою. Використання LLM як анотатора дозволяє швидко збільшити корпус, але воно може відтворювати зміщення teacher-моделі. Якщо teacher гірше розпізнає сарказм, суржик або mixed, ці помилки можуть перейти в розширений корпус. Саме тому COSMUS+ не подається як повністю ручний золотий стандарт, а супроводжується ручною перевіркою, distribution analysis і порівнянням моделей, навчених лише на COSMUS та на COSMUS+.

У практичному застосуванні sentiment-модель для Telegram не повинна використовуватися як єдине джерело рішень у чутливих соціальних або політичних сценаріях. Її коректна роль полягає в попередньому агрегуванні, пошуку тенденцій, сортуванні великих масивів повідомлень або виділенні прикладів для подальшого ручного аналізу. Це особливо важливо для класу mixed і для саркастичних повідомлень, де автоматична помилка може змінити інтерпретацію суспільної реакції.

2.5 Додаткові етапи оцінювання: калібрування, пояснюваність і розбір показових випадків

Окрім агрегованих метрик якості, у роботі використано три додаткові напрями оцінювання: калібрування ймовірностей, пояснюваність прогнозів і розбір показових випадків. Вони потрібні тому, що високе значення Macro F1 не гарантує, що модель коректно оцінює власну невпевненість, спирається на інтерпретовані мовні маркери або стабільно працює на семантично неоднозначних прикладах.

Калібрування ймовірностей оцінювалося за Expected Calibration Error (ECE) [35]. Для кожного прикладу модель повертає прогнозований клас ($\widehat{y_i}$) і впевненість ($s_i = \max_c p_i\,(c)$). Далі інтервал впевненості ([0,1]) ділиться на (M) бінів ($B_1, \dots, B_M$). Для кожного біна обчислюються емпірична точність і середня впевненість:

$$acc(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathrm{b1}(\widehat{y_i} = y_i),$$

$$conf(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} s_i.$$

Тоді ECE визначається як зважена сума відхилень між точністю та впевненістю:

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{n} |acc(B_m) - conf(B_m)|.$$

Нижче значення ECE означає кращу узгодженість між прогнозованою впевненістю та фактичною точністю. Наприклад, якщо модель часто дає впевненість близько 0,9, але правильно відповідає лише у 0,7 випадків, вона є надмірно впевненою. У роботі ECE використовується як діагностична метрика, а не як основний критерій вибору моделі.

Для локальної пояснюваності використано LIME і SHAP. LIME пояснює окремий прогноз через локальну апроксимацію поведінки моделі простішою інтерпретованою моделлю. Для тексту це означає, що з початкового повідомлення (x) формуються збурені варіанти (z), у яких частина токенів або слів вилучається чи маскується. Для цих варіантів отримуються прогнози моделі ($f(z)$), після чого навчається локальна модель ($g(z)$), яка мінімізує:

$$\mathcal{L}(f, g, \pi_x) + \Omega(g),$$

де $(\pi_{x(z)})$ – вага близькості збуреного прикладу (z) до початкового тексту (x), а $\big(\Omega(g)\big)$ – штраф за складність інтерпретованої моделі. У результаті LIME оцінює, які слова найбільше вплинули на конкретний прогноз у локальному околі цього прикладу.

SHAP базується на ідеї значень Шеплі з теорії кооперативних ігор. Для ознаки (j) її внесок (ϕ_j) визначається як середній маржинальний внесок цієї ознаки в усіх можливих підмножинах ознак:

$$\phi_j = \sum_{S \subseteq F \setminus j} \frac{|S|!\,(|F| - |S| - 1)!}{|F|!} [f(S \cup j) - f(S)],$$

де (F) – множина всіх ознак, (S) – підмножина ознак без (j), а $(f(S))$ - прогноз моделі за наявності тільки ознак із (S). У текстовій задачі точне обчислення всіх підмножин є надто дорогим, тому застосовується наближена процедура з маскуванням токенів. SHAP дає змогу оцінити, які лексеми системно підсилюють або послаблюють прогноз певного класу.

LIME і SHAP не доводять причинність у строгому сенсі. Їхні результати слід інтерпретувати як локальні та наближені діагностичні свідчення того, на які токени модель була чутливою під час прогнозування. Тому в роботі ці методи використовуються не для самостійного доведення якості моделі, а для зіставлення з кількісними результатами й розбором помилок.

Якісний аналіз показових випадків використовувався для виявлення типових ситуацій, у яких агреговані метрики недостатні. До таких ситуацій належать короткі контекстно залежні репліки, сарказм, цитати без явного маркування позиції автора, тексти з одночасною критикою і підтримкою, а

також випадки, де всі або майже всі моделі дають однаково неправильний прогноз. Для таких прикладів у роботі аналізувалися істинна мітка, прогнози кількох моделей, рівень згоди моделей і можливі лінгвістичні причини помилки.

2.6 Обчислювальна інфраструктура та програмне середовище

Розподіл обчислювальних ресурсів. Експерименти різного рівня складності було розподілено між доступними апаратними платформами. Комерційні API-моделі (DeepSeek, Gemini, Kimi) тестувалися виключно на локальному комп'ютері, оскільки цей процес вимагає лише відправки мережевих HTTP-запитів без навантаження на локальний GPU.

Більшість обчислювально інтенсивних експериментів (дотренування encoder-моделей, інференс MamayLM) проводилась на хмарних GPU Tesla T4 16 GB (Google Colab, Kaggle). Для найбільших моделей використовувалися інстанси з NVIDIA A100 40 GB VRAM. Зокрема, QLoRA-дотренування MamayLM 4B виконувалося на A100 40 GB, а QLoRA-дотренування MamayLM 12B – на NVIDIA RTX PRO 6000 з 102 GB VRAM, оскільки 12B-модель навіть у 4-бітній квантизації з LoRA-адаптерами потребує суттєво більшого обсягу пам'яті, ніж 40 GB.

Програмний стек. Усі алгоритми працюють на базі мови програмування Python (версія 3.12.x) із закріпленими версіями пакетів у requirements.txt. Це підвищує відтворюваність експериментів і спрощує повторне виконання основних етапів, хоча не усуває повністю залежність від апаратного середовища, версій GPU-бібліотек і зовнішніх API-моделей. Код проєкту опубліковано у відкритому репозиторії GitHub[2].

[2] https://github.com/koihivilas/ua-cssent

Ядро Machine Learning (Глибоке навчання) базується на фреймворку PyTorch (≥2.2.0) як головному каркасі для градієнтних обчислень. Для зручної роботи з архітектурою моделей і токенізаторами задіяна екосистема HuggingFace, зокрема пакети transformers (≥4.46.0) та datasets. Бібліотека bitsandbytes застосовується для низькорівневих операцій із 4-bit квантуванням, а scikit-learn (≥1.5.0) для обчислення вагових коефіцієнтів класів та формування звітів класифікації.

Логування частини експериментів виконувалося за допомогою wandb (Weights & Biases), що використовувався для збереження кривих навчання, значень loss і валідаційних метрик. Для побудови графіків, матриць помилок і діаграм розподілів застосовувалися matplotlib, seaborn і plotly.

Клієнтські API та модулі включають telethon для асинхронної комунікації з Telegram (для збору даних) та офіційні SDK openai (для DeepSeek/Kimi через OpenAI-compatible API) і google-genai для викликів до API Gemini.

Обчислювальні компроміси та практичне розгортання. Обчислювальна інфраструктура в цій роботі має прямий зв'язок із практичними висновками. Якщо модель оцінюється лише за Macro F1, то різниця між API LLM, encoder-моделлю та QLoRA-дотренованою decoder-моделлю може здаватися невеликою. Проте в реальній системі sentiment monitoring важливими є також швидкість інференсу, вартість обробки одного повідомлення, контроль версії моделі, можливість повторення результату, приватність вхідних даних і вимоги до апаратного забезпечення.

Комерційні API LLM мають найнижчий поріг входу для дослідника: не потрібно зберігати модель локально, налаштовувати GPU або контролювати memory footprint. Водночас кожен прогноз є зовнішнім запитом, що створює залежність від тарифікації, лімітів, доступності сервісу та політики обробки даних. Для академічного baseline це прийнятно, але для постійного моніторингу великих потоків Telegram-коментарів така залежність може бути суттєвим практичним обмеженням.

Локальні encoder-моделі потребують GPU для навчання, але після дотренування їхній інференс значно простіший і передбачуваніший. Modern-LiBERTa або XLM-RoBERTa Large можна запускати в локальному середовищі, контролювати checkpoint, batch size, max length і preprocessing. Для системи, яка регулярно класифікує однотипні короткі повідомлення, encoder може бути економічно вигіднішим за API LLM, навіть якщо початкове навчання потребує хмарного GPU. Крім того, локальний запуск зменшує ризики, пов'язані з передаванням чутливих текстів зовнішнім сервісам.

MamayLM після QLoRA займає проміжну позицію. З одного боку, це локальна відкрита decoder-only модель, що зменшує залежність від комерційного API. З іншого боку, її інференс і навчання залишаються важчими за encoder-класифікатор, а якість за Macro F1 трохи нижча за Modern-LiBERTa і Gemini. Тому MamayLM є особливо цікавою не як найпростіший production-класифікатор, а як приклад відкритої україноцентричної LLM, яку можна адаптувати до спеціалізованої задачі без повного оновлення всіх параметрів.

У практичному застосуванні доцільно розрізняти три сценарії. Для швидкого прототипування або разового аналізу можна використовувати API LLM, оскільки вона дає сильний baseline без навчання. Для стабільного регулярного моніторингу доцільніше розгортати локальний encoder, особливо якщо якість близька до API-моделі. Для дослідницьких або відкритих україномовних LLM-сценаріїв варто розглядати MamayLM QLoRA, оскільки вона демонструє, що параметрично-ефективне дотренування може суттєво підвищити якість decoder-only моделі.

Такий поділ важливий для інтерпретації результатів дисертації. Найкраща модель за однією метрикою не завжди є найкращою для впровадження. Якщо різниця між Gemini 3.0 Flash і Modern-LiBERTa становить лише кілька тисячних Macro F1, то питання вартості, приватності й повторюваності можуть переважити невелику метрикальну перевагу API

LLM. Саме тому результати в четвертому розділі подаються разом із практичним аналізом режимів використання, а не тільки як leaderboard.

2.7 Запропонований комплексний підхід CSSent

CSSent є комплексним підходом до адаптації encoder-моделей для сентимент-аналізу код-змішаних текстів. Він не є окремою новою мовною моделлю; це схема навчання й оцінювання, у якій базова encoder-модель доповнюється складовими, спрямованими на три проблеми: мовну неоднорідність повідомлень, дисбаланс класів і складність категорії mixed. Усі конфігурації CSSent оцінюються на тому самому тестовому протоколі, що й відповідні базові encoder-моделі, тому результат інтерпретується як ефект запропонованого способу адаптації, а не як наслідок зміни датасету чи метрики.

CSSent складається з п'яти складових, що оцінюються окремо та в комбінаціях. Перша складова, `lh`, додає мовно-зумовлену класифікаційну голову: до текстового представлення encoder-моделі передається мовна ознака повідомлення, після чого модель прогнозує один із класів тональності. Таке подання мовної структури є релевантним для задачі, де українські, російські та змішані фрагменти можуть мати різну прагматичну роль. Друга складова, `fl`, використовує зважену фокусну функцію втрат [5], яка зменшує внесок легких прикладів і сильніше фокусується на складних або міноритарних класах, насамперед mixed. Третя складова, `mt`, вводить допоміжну задачу визначення мови за схемою багатозадачного навчання [36]: спільний encoder одночасно оптимізується для прогнозу тональності та мовної мітки, що сприяє збереженню інформації про мовну структуру тексту в проміжних представленнях.

Четверта складова, `csa`, відповідає за аугментацію код-змішаних прикладів. Для цього використано стратегії CSPI (Code-Switch Point Injection) і SPP (Sentiment-Preserving Paraphrase): CSPI моделює контрольовані вставки іншої мови в межах повідомлення, а SPP створює перефразування зі збереженням тональності. Їхня мета полягає не в механічному збільшенні корпусу, а в посиленні представленості ситуацій, де тональність формується через поєднання мовних і оцінних маркерів. П'ята складова, `kd`, використовує дистиляцію знань [37] від Gemini Flash через м'які мітки: модель-учень навчається не лише на жорсткій мітці класу, а й на розподілі ймовірностей моделі-вчителя. Це особливо важливо для межових прикладів, де модель-вчитель не є повністю впевненою між, наприклад, negative і mixed.

Ці стратегії аугментації мають дещо спільну ідею із дослідженням [26]: код-змішаний текст розглядається як окремий тип даних, а не як монолінгвальний приклад із додатковим шумом. У [26] мовна структура враховується через упорядкування код-змішаних даних за часткою ресурсно багатої мови та поступове навчання. У CSSent цей принцип реалізовано через аугментацію код-змішаних прикладів, допоміжну задачу визначення мови та мовно-зумовлену класифікаційну голову.

Схему роботи CSSent подано на рис. 3.1. Вхідний текст спочатку проходить токенізацію та encoder-основу, після чого формується текстове представлення. Далі це представлення використовується звичайною sentiment head і, для конфігурацій із lh, мовно-зумовленою класифікаційною головою. Мовна ознака також може використовуватися в допоміжній задачі mt. Під час навчання до основної функції втрат додаються weighted focal loss, distillation loss від soft labels і, для конфігурацій із csa, аугментовані приклади CSPI/SPP. Виходом пайплайну є один із чотирьох класів: positive, negative, neutral або mixed.

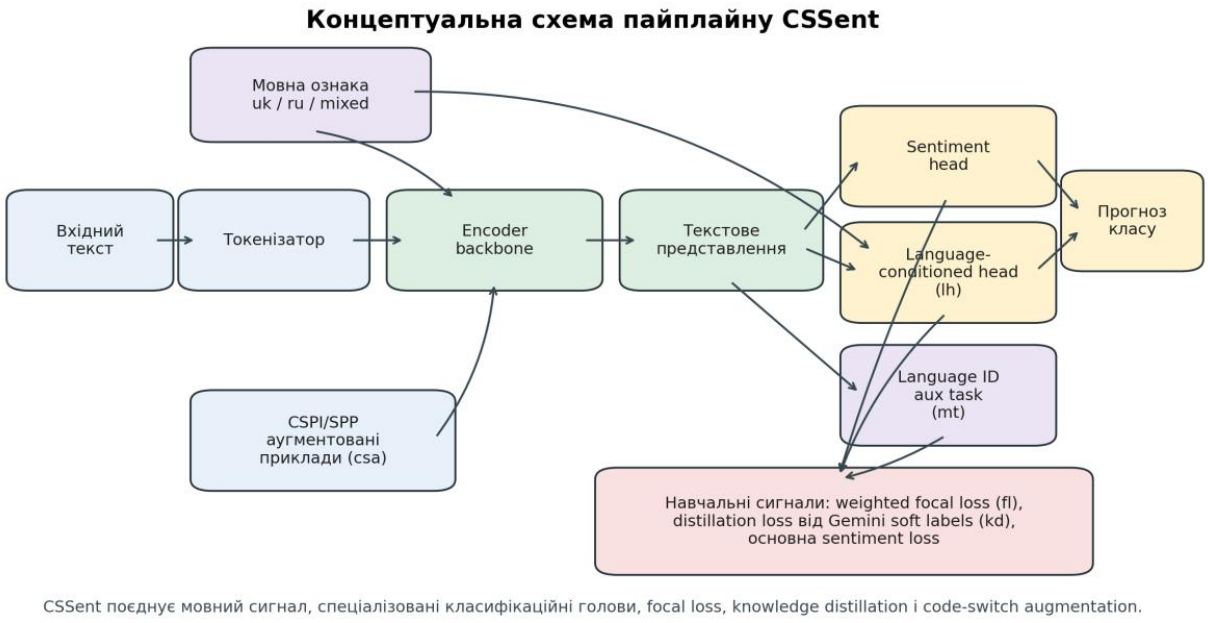

Рисунок 2.2 – Концептуальна схема пайплайну CSSent: токенізатор, encoder, language-conditioned head, sentiment head, допоміжна language-ID задача, focal loss, distillation loss та CSPI/SPP-аугментація

Висновки до розділу 2

У цьому розділі сформовано методологічну основу дослідження мультикласового сентимент-аналізу українсько-російських код-змішаних текстів. Розділ охоплює математичні основи трансформерної архітектури, функції втрат і метрики оцінювання, опис корпусів COSMUS і COSMUS+, додаткові критерії аналізу моделей, обчислювальну інфраструктуру та запропонований комплексний підхід CSSent.

Початковий корпус COSMUS визначає основні умови задачі: чотири класи тональності, наявність українських, російських і змішаних повідомлень, а також виражений дисбаланс класів. Найменш представленим є клас mixed, що обґрунтовує використання Macro F1 як основної метрики та

застосування зважених функцій втрат під час навчання. Розширення COSMUS+ сформовано для перевірки стійкості моделей на ширшому розподілі Telegram-повідомлень і збільшення абсолютної кількості прикладів міноритарного класу. Водночас COSMUS+ не розглядається як повністю ручний еталонний корпус, оскільки частина його міток отримана автоматично.

Крім агрегованих метрик якості, у методології передбачено аналіз калібрування ймовірностей, пояснюваності прогнозів і показових випадків. Expected Calibration Error використовується для перевірки узгодженості між упевненістю моделі та фактичною точністю, LIME і SHAP – для локального аналізу внеску токенів у прогноз, а розбір показових випадків – для пояснення помилок, які не видно з одних лише агрегованих метрик. Описана обчислювальна інфраструктура дає змогу зіставити API-моделі, локальні encoder-моделі та decoder-only моделі з урахуванням не лише якості, а й практичних обмежень розгортання.

На основі сформульованої у розділі 1 постановки задачі запропоновано комплексний підхід CSSent для адаптації encoder-моделей. CSSent інтегрує в єдиній схемі навчання п'яти складових, кожна з яких спрямована на конкретний аспект задачі: мовно-зумовлена класифікаційна голова й допоміжна задача визначення мови враховують мовну структуру повідомлення, зважена фокусна функція втрат – дисбаланс класів і складність категорії mixed, аугментація CSPI/SPP – код-змішану природу даних, а дистиляція знань від Gemini Flash – невпевненість моделі-вчителя на межових прикладах.

РОЗДІЛ 3 ДИЗАЙН ЕКСПЕРИМЕНТІВ ДЛЯ ОЦІНЮВАННЯ МОВНИХ МОДЕЛЕЙ НА COSMUS I COSMUS+

3.1 Базовий конвеєр препроцесингу тексту та контроль якості розмітки

Препроцесинг зібраних текстових даних є невід'ємним етапом підготовки перед поданням текстів у трансформерні моделі. Конвеєр препроцесингу охоплює кроки, представлені нижче.

1. Базове очищення (Clean-Up):
   - видалення URL-адрес (http://...);
   - видалення згадок користувачів (@username);
   - нормалізація пробілів (множинні пробіли і табуляції зводяться до одного).
2. Обробка емодзі та спецсимволів. У сучасних підходах емодзі не видаляються. Більшість трансформерів здатні оперувати емодзі як незалежними семантичними токенами, оскільки вони містять значну частку емоційної полярності повідомлення.
3. Збереження мовної специфіки. Суржикові форми та граматичні помилки (код-змішування) навмисно зберігаються. Жодних виправлень до нормативної форми (орфографічної корекції) не проводиться, оскільки метою дослідження є аналіз соціального тексту "as-is". Також не застосовуються ні стоп-листи, ні лематизація, оскільки трансформер "розуміє" контекст за допомогою всього речення, а втрата прийменників та займенників може спотворити оригінальну тональність.
4. Токенізація (Model-specific). Тексти розбиваються на суб-токени (subword tokenization) відповідно до вимог конкретної архітектури моделі за допомогою HuggingFace-токенізаторів. Токен (token) є мінімальною одиницею тексту, яку обробляє модель; залежно від

алгоритму токенізації, це може бути ціле слово, його частина (субслово) або окремий символ:

- mBERT використовує WordPiece (119K словник);
- XLM-RoBERTa і UkrRoBERTa/Modern-LiBERTa використовують SentencePiece алгоритм (з різними розмірами вокабуляру);
- MamayLM (на базі Gemma 3) використовує SentencePiece-токенізатор з алгоритмом BPE (Byte-Pair Encoding).

Максимальна довжина послідовності (max_length) встановлена на 512 токенів для encoder-моделей. Для текстів, що перевищують цю довжину, застосовується segmented inference з перекриттям 128 токенів та усередненням softmax-ймовірностей. Коротші тексти доповнюються до єдиної довжини батчу динамічним padding-ом. Медіанна довжина тексту у COSMUS становить 96 символів, тому обрізання застосовується рідко.

Контроль якості та обмеження автоматичної розмітки. Автоматична розмітка соціальних текстів за допомогою LLM є практичним способом швидко збільшити корпус, але вона не є повною заміною ручної анотації. У цій роботі COSMUS+ використовується як розширення для додаткових експериментів, а не як безумовно «ідеальна» версія датасету. Модель може покращуватися на більшому корпусі, але частина приросту або помилок може бути пов'язана з шумом автоматичних label, зміною тематичного розподілу або специфікою Telegram-каналів.

Найскладнішою для автоматичної розмітки є межа між neutral, negative і mixed. Повідомлення про негативну подію може бути нейтральним, якщо автор просто передає факт, і негативним, якщо автор явно оцінює ситуацію. Повідомлення з критикою може бути negative, якщо воно має одну домінантну негативну позицію, і mixed, якщо в ньому є самостійний позитивний компонент. Саме такі межі складно стабільно відтворити без ширшого контексту, особливо в коротких Telegram-коментарях.

Мовна розмітка загалом є стабільнішою, що підтверджується manual agreement. Однак і тут є обмеження: суржик може бути записаний нестандартно, російські та українські форми можуть збігатися графічно або бути змішаними на рівні морфології, а окремі цитати іншою мовою не завжди перетворюють увесь текст на mixed. Тому language label в роботі використовується як корисний додатковий сигнал, але не як абсолютно точний опис кожного токена.

Для зменшення ризику помилок у фінальних висновках COSMUS+ аналізується окремо від початкового COSMUS. Найважливіші порівняння на оригінальному benchmark виконуються на COSMUS test, а COSMUS+ використовується для перевірки, чи допомагає розширення даних на новому test set. Такий поділ дозволяє не змішувати два питання: якість моделей на відомому відкритому benchmark і користь додаткових автоматично розмічених Telegram-прикладів.

3.2 Дизайн та конфігурація експериментів

Дослідження побудовано як серію систематичних зіставлень різних архітектурних концепцій у межах одного завдання: 4-класового мультикласового сентимент-аналізу текстів з код-змішуванням. Експериментальний шлях розділено на кілька взаємопов'язаних етапів: розвідувальний аналіз COSMUS; API baselines для комерційних LLM у режимах zero-shot і few-shot; повне дотренування encoder-моделей; prompting MamayLM 4B і 12B; QLoRA-дотренування MamayLM 4B і 12B; розробку та ablation study пайплайну CSSent; побудову COSMUS+; оцінювання калібрування, пояснюваності та якісний розбір показових випадків.

Для уникнення стохастичних відхилень у генерації та випадкових ініціалізаціях у відповідних експериментах фіксувався random seed. Для

фінальної звітності використовувалися локальні CSV, JSON, Parquet-артефакти та збережені графіки, що дає змогу перевірити числові результати незалежно від зовнішніх сервісів логування.

3.2.1 Конфігурація emany API Baselines

Тестування можливостей комерційних Large Language Models без їх донавчання встановлює мірило для решти дослідження. Експерименти проводяться з трьома моделями, обраними за критерієм cost-efficiency:

- DeepSeek V3.2: відкрита Mixture of Experts (MoE) архітектура, доступна через OpenAI-сумісне API;
- Kimi K2.5 (Moonshot AI): модель з оптимальним балансом великого контексту та якісних інференсів;
- Gemini 3.0 Flash Preview (Google): модель, оптимізована для швидкого інференсу.

Тестування проходить у чотирьох конфігураціях (2 мови промпту × 2 режими):

- Zero-shot: моделі надається лише системна інструкція та перелік з чотирьох класів;
- Few-shot: моделям додатково подаються 7 реальних прикладів з тренувальної вибірки COSMUS із анотаціями різних класів тональності.

3.2.2 Конфігурація етапу Encoder Fine-tuning

Для навчання локальних encoder-моделей визначено п'ять кандидатів від базового до великого розміру:

- mBERT (bert-base-multilingual-cased): ~178M параметрів, навчена на 104 мовах. WordPiece токенізатор; універсальний мультилінгвальний baseline;
- XLM-RoBERTa Base (xlm-roberta-base): 278M параметрів; еталон мультілінгвального підходу; використовує SentencePiece-токенізатор; для класифікації застосовується представлення початкового токену <s>, аналогічно до [CLS] у BERT;
- XLM-RoBERTa Large (xlm-roberta-large): ~560M параметрів. Збільшена з місткістю до 24 шарів;
- UkrRoBERTa (youscan/ukr-roberta-base): україноцентрична модель (~125M параметрів). Навчена виключно на ~60GB українських текстів; її особливістю є майже ідеальна нефрагментована токенізація української мови, що може виявитись вирішальним на коротких текстах;
- Modern-LiBERTa Large (Goader/modern-liberta-large): найсучасніша (~410M парам.) україноцентрична варіація ModernBERT; включає архітектурні оновлення: RoPE, чергування локальної/глобальної уваги, GeGLU та Flash Attention 2; підтримує послідовності до 8192 токенів.

Всі моделі використовують однаковий оптимізатор AdamW зі спадом ваг (weight decay) 0.01. Розмір batches становить 16 (або 8 для Large моделей з акумуляцією градієнтів, щоб імітувати effective batch 16).

Learning Rate (швидкість навчання) ~$1{,}17\times10^{-5}$ з 10% розігрівом (лінійний warmup_ratio=0.1) і подальшим лінійним (linear) спадом. Епохи

тренування сягають максимум 10, але з вбудованим механізмом Early Stopping (patience=5), якщо метрика Macro F1 на валідаційному датасеті перестає зростати. Застосовується функція втрат Weighted Cross-Entropy. Змішана точність fp16 (mixed precision) увімкнена для всіх encoder-моделей для двократного зменшення споживання VRAM.

Додатково для Modern-LiBERTa було виконано Optuna-оптимізацію гіперпараметрів на валідаційній вибірці. Перевірялися learning rate, weight decay, dropout, warmup ratio, scheduler, batch size, gradient accumulation, max length, кількість епох і label smoothing. Наявні артефакти Optuna не містять підтвердженого test-set покращення порівняно з базовою конфігурацією, узгодженою з попередніми COSMUS-експериментами. Тому в підсумковому порівнянні Optuna розглядається як перевірка простору гіперпараметрів, а не як окрема фінальна модель із заявленим приростом якості.

Для зменшення ризику перенавчання та стабілізації навчання використовувалася рання зупинка за валідаційною Macro F1, warmup на початку навчання, лінійний спад learning rate і mixed precision для encoder-моделей. У задачі з дисбалансом класів ці рішення мають різні ролі: early stopping обмежує перенавчання на мажоритарних класах, warmup зменшує нестабільність перших кроків fine-tuning, а mixed precision дозволяє збільшити ефективність обчислень без зміни цільової функції. Для порівняння моделей важливо, що основні encoder-baselines навчалися за спільним протоколом, тому різниця в результатах інтерпретується насамперед через архітектуру, токенізацію та мовну спеціалізацію моделі, а не через несумісні режими тренування.

3.2.3 Конфігурація етапів Decoder-only (MamayLM)

Для україноцентричної decoder-only моделі MamayLM перевірено два режими адаптації. Перший режим, prompting, не оновлює ваги моделі й оцінює здатність MamayLM виконувати класифікацію лише за інструкцією. Другий режим, QLoRA, є параметрично-ефективним дотренуванням на COSMUS із 4-бітною квантизацією базової моделі та LoRA-адаптерами.

Prompting оцінювався для MamayLM 4B і MamayLM 12B. Для інференсу модель завантажувалася в 4-бітній квантизації NF4 з greedy decoding, щоб забезпечити детермінованість відповіді. Кожен текст оформлювався у форматі instruction/chat prompt із вимогою повернути одну з чотирьох міток. Для стабільності результатів відповіді додатково нормалізувалися до допустимого label set.

QLoRA-дотренування було виконано для MamayLM 4B і MamayLM 12B. Для обох моделей застосовано LoRA r=32, alpha=64, effective batch size 16, learning rate 0.0002 і 3 епохи. Використання QLoRA дозволяє істотно зменшити обсяг оновлюваних параметрів порівняно з повним дотренуванням LLM.

3.3 Відтворюваність експериментів

Відтворюваність забезпечувалася через фіксацію тестових протоколів, збереження конфігурацій запуску та окреме збереження прогнозів моделей для подальшої перевірки метрик. Для кожної групи експериментів використовувався той самий набір класів positive, negative, neutral, mixed, а також єдина схема нормалізації відповідей моделей до цих класів.

Оцінювання на COSMUS і COSMUS+ не змішувалося в одній метриці. Результати на початковому COSMUS test використовувалися для порівняння API-моделей, encoder-моделей, CSSent і MamayLM у спільному контрольному сценарії. Результати на COSMUS+ test використовувалися окремо для перевірки впливу розширення корпусу й стійкості моделей на новішому розподілі даних.

Для зменшення ризику випадкових розбіжностей фінальні таблиці формувалися тільки з експериментів, для яких були доступні прогнозовані мітки, агреговані метрики та per-class-звіти. Це дало змогу перевірити, що Accuracy, Macro F1, Weighted F1 і F1 за класами відповідають одному й тому самому тестовому набору.

3.4 Prompt-протокол і нормалізація відповідей LLM

Оцінювання LLM через prompting має свою специфіку, оскільки модель не повертає клас із фіксованого softmax-шару, а генерує текстову відповідь. Тому для коректного порівняння необхідно не лише сформулювати інструкцію, а й визначити правила нормалізації відповіді. У цій роботі допустимими виходами є лише чотири label: positive, negative, neutral і mixed. Якщо модель генерує розгорнуте речення, наприклад пояснення або фразу з кількома словами, відповідь приводиться до допустимого класу лише тоді, коли з неї однозначно можна вилучити label.

Zero-shot prompt перевіряє, наскільки модель здатна виконати задачу за інструкцією без прикладів. Такий режим корисний як чистий тест instruction-following і внутрішніх знань моделі. Однак у чотирикласовому сентимеит-аналізі він має суттєві обмеження. Модель може по-різному інтерпретувати mixed, плутати neutral із відсутністю емоційної лексики або трактувати

негативний зміст новини як negative sentiment навіть без авторської оцінки. Тому zero-shot результати важливо зіставляти з few-shot режимом.

Few-shot prompt додає приклади з навчальної вибірки, які задають не лише формат відповіді, а й межі класів. Для класу mixed це особливо важливо: модель має побачити, що mixed означає співіснування позитивного й негативного сигналів, а не просто невпевненість. Водночас few-shot режим може бути чутливим до конкретного набору прикладів. Якщо приклади надто прості, модель може погано працювати з іронією; якщо приклади надто довгі, вона може гірше переносити правило на короткі реакції.

Мова промпту також є експериментальним фактором. Україномовний prompt може краще відповідати задачі й корпусу, особливо для моделей із сильною українською підтримкою. Англомовний prompt інколи може бути стабільнішим для моделей, які проходили значну instruction-tuning оптимізацію англійською. Саме тому в роботі перевіряються eng_zero_shot, eng_few_shot, ukr_zero_shot і ukr_few_shot, а висновки формулюються за фактичними результатами, а не за припущенням, що українська інструкція завжди найкраща.

Для MamayLM prompt-протокол виконує ще одну функцію: він діагностує, наскільки відкрита україноцентрична decoder-only модель здатна поводитися як класифікатор без навчання. Результати показують, що prompting 4B є нестабільним і суттєво поступається encoder-підходам, тоді як 12B prompting помітно сильніший. Проте навіть 12B prompting не досягає рівня QLoRA. Це підтверджує, що для спеціалізованої класифікаційної задачі instruction-following не замінює адаптацію до цільового корпусу.

Нормалізація відповідей потрібна також для чесного порівняння з encoder-моделями. Encoder завжди повертає один із класів, тоді як LLM може відповідати природною мовою. Якщо не нормалізувати формат, частина помилок буде пов'язана не з sentiment understanding, а з невиконанням інструкції. Тому в роботі відокремлюється змістова якість прогнозу від

технічної стабільності формату, наскільки це можливо в межах prompt-based оцінювання.

Висновки до розділу 3

У цьому розділі сформовано дизайн експериментального дослідження для оцінювання мовних моделей на корпусах COSMUS і COSMUS+. Експериментальна схема охоплює три групи режимів використання: комерційні API-моделі без дотренування, локальні encoder-моделі з повним дотренуванням, а також decoder-only модель MamayLM у режимах інструктивного промптингу та QLoRA-дотренування.

Для початкового COSMUS використано статичне розбиття `df_set`, що зберігає порівнюваність результатів з іншими експериментами на цьому корпусі. Для COSMUS+ сформовано окремі train, validation і test підвибірки з контролем витоку даних між навчальною та тестовою частинами. Препроцесинг залишено мінімальним: він усуває технічний шум, але зберігає суржик, емодзі, нестандартну орфографію та мовне перемикання, оскільки ці ознаки можуть бути пов'язані з тональністю й комунікативним контекстом повідомлення.

Для encoder-моделей визначено єдиний протокол повного дотренування (AdamW, weight decay 0.01, лінійний warmup, рання зупинка за валідаційною Macro F1, mixed precision), що дозволяє інтерпретувати різницю результатів насамперед через архітектуру, токенізацію та мовну спеціалізацію базової моделі. Для decoder-only моделей MamayLM перевірено два режими: промптинг без оновлення ваг і параметрично-ефективне дотренування QLoRA. Це дає змогу зіставити інструктивну здатність відкритої україноцентричної LLM із її якістю після адаптації до цільового корпусу. Для API-моделей визначено prompt-протокол і правила

нормалізації відповідей до чотирьох допустимих класів: positive, negative, neutral і mixed; розглянуто чотири стратегії – eng_zero_shot, eng_few_shot, ukr_zero_shot та ukr_few_shot – що дозволяє оцінити роль мови інструкції та наявності прикладів.

Відтворюваність експериментів забезпечується єдиним набором класів, фіксованими тестовими протоколами, збереженням прогнозів моделей і розділенням оцінювання на COSMUS та COSMUS+. Така організація дозволяє порівнювати не лише окремі архітектури, а й режими їх адаптації до задачі: зовнішній API-запит, повне дотренування encoder-моделі, промптинг і QLoRA-дотренування decoder-only моделі.

РОЗДІЛ 4 РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТІВ ТА ЇХ АНАЛІЗ

4.1 Результати розвідувального аналізу даних (EDA)

4.1.1 Загальна статистика та розподіли датасету

Розвідувальний аналіз (Exploratory Data Analysis) дозволив оцінити структуру та характеристики датасету COSMUS. Датасет містить 12 224 текстових зразки з платформи Telegram. Кожен запис супроводжується текстом повідомлення, міткою тональності (positive, negative, neutral, mixed) та мовною категорією.

Розподіл класів тональності (табл. 4.1) підтверджує суттєвий дисбаланс, притаманний реальним даним із соціальних мереж.

Таблиця 4.1 – Розподіл класів тональності в COSMUS

Клас тональності	*Кількість*	*Відсоток від загального обсягу*
neutral	4702	38.5%
negative	4541	37.1%
positive	2373	19.4%
mixed	608	5.0%

Коефіцієнт дисбалансу (max/min) становить 7.7:1. Клас mixed є найменш представленим і становить найбільший виклик для машинного навчання.

Аналіз мовних категорій виявив три мовні групи: українська (uk), російська (ru) та змішані тексти з код-змішуванням (mixed).

Крос-табуляція (перехресний аналіз) між тональністю та мовою показала наявність певних відмінностей у розподілі тональності між мовними групами. Ці спостереження вказують на те, що мовні патерни можуть бути додатковим сигналом для алгоритмів класифікації тональності.

У таблиці 4.2 показано статистику після аналізу довжин текстів.

Таблиця 4.2 – Аналіз довжин текстів

Показник	*Символи*	*Слова*
Середнє	170.6	25.0
Медіана	96	15
75-й перцентиль	204	30
Максимум	5749	774

Абсолютна більшість зразків датасету вільно вписується у стандартне обмеження 512 токенів, використане для encoder-моделей. Для поодиноких довших текстів застосовується segmented inference з перекриттям 128 токенів.

4.1.2 Розподіли COSMUS+ та зіставлення з початковим корпусом

Окрім початкового COSMUS, у роботі сформовано розширений корпус COSMUS+. Його побудова була потрібна не лише для збільшення кількості навчальних прикладів, а й для перевірки того, чи зберігаються основні властивості задачі на новіших Telegram-даних. Після очищення нових повідомлень, вилучення спаму, надто коротких текстів, дублікатів і контролю витоку даних фінальний COSMUS+ містить 33 267 рядків. До нього увійшли 21 045 нових очищених прикладів і 12 222 рядки з початкового COSMUS після вилучення двох рядків, що створювали ризик leakage для фінального test set (табл. 4.3, рис. 4.1).

Таблиця 4.3 – Порівняння розподілів класів тональності у COSMUS і COSMUS+

Корпус	*Усього*	*positive*	*negative*	*neutral*	*mixed*
COSMUS original	12 224	2 373	4 541	4 702	608
COSMUS+ new cleaned	21 045	3 554	12 440	4 112	939
COSMUS+ combined	33 267	5 926	16 981	8 813	1 547

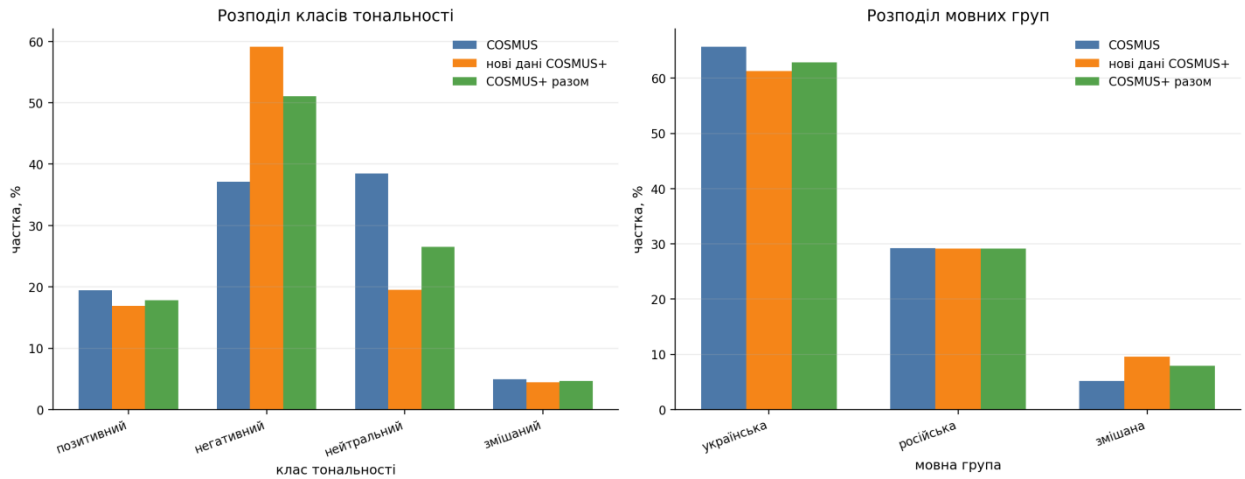

Рисунок 4.1 – Порівняння розподілів класів тональності та мовних груп у COSMUS і COSMUS+

Розширення корпусу не усуває дисбаланс класів, але збільшує абсолютну кількість прикладів класу mixed з 608 до 1 547. У combined-версії COSMUS+ найбільшим класом стає negative, що становить близько 51,0% корпусу, тоді як mixed залишається міноритарним із часткою близько 4,7%. Отже, COSMUS+ не спрощує задачу штучно, а зберігає її ключову складність: модель має працювати з нерівномірним розподілом класів і рідкісними амбівалентними повідомленнями.

За мовною ознакою COSMUS+ також зберігає тригрупову структуру: uk, ru і mixed. У combined-корпусі наявно 20 917 українськомовних, 9 702 російськомовні та 2 648 змішаних текстів. Частка мовно змішаних прикладів у COSMUS+ є більшою, ніж у початковому COSMUS, що робить розширений корпус корисним для перевірки стійкості моделей до код-змішування (рис. 4.2).

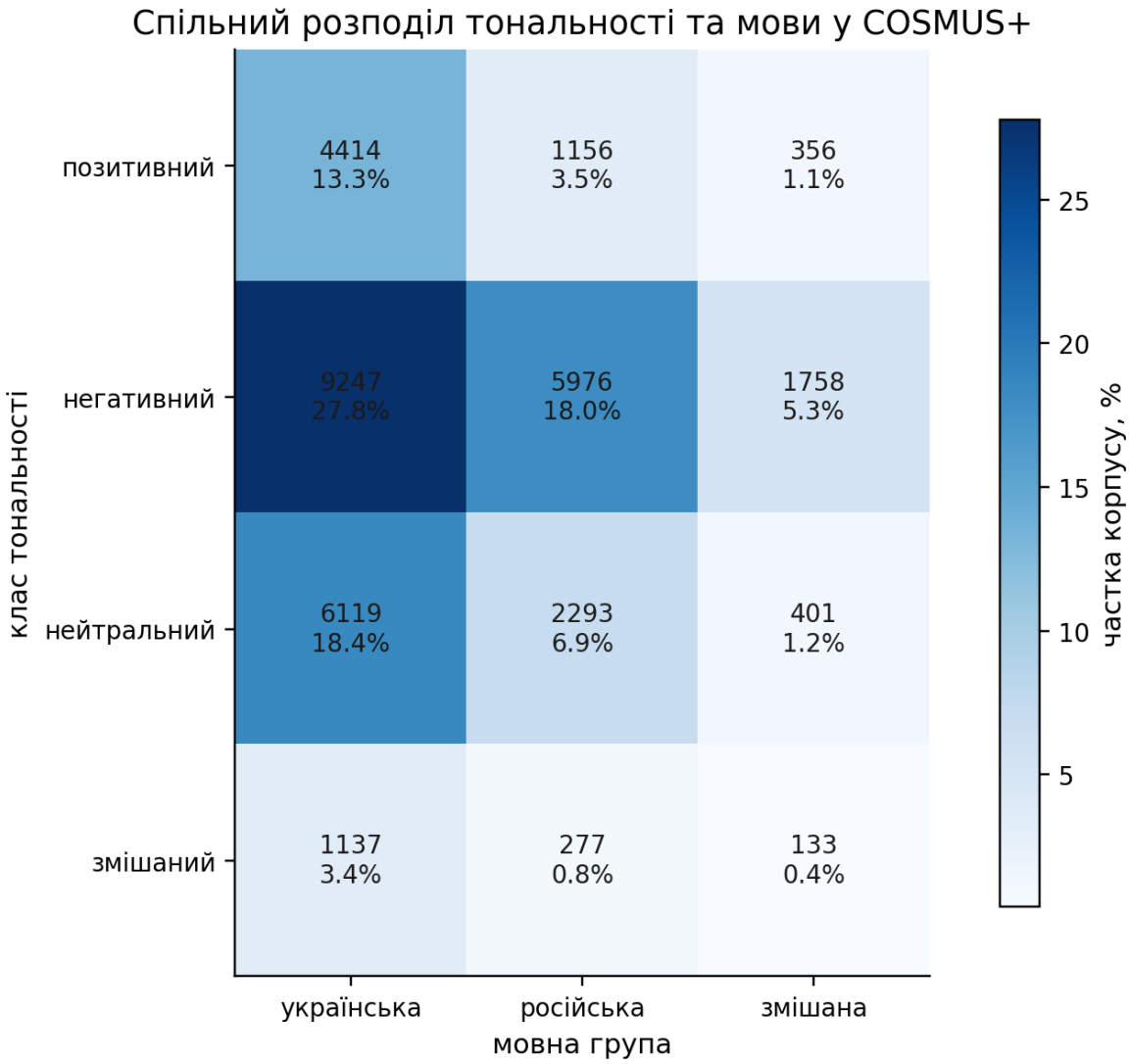

Рисунок 4.2 – Спільний розподіл тональності та мовних груп у корпусі COSMUS+

4.1.3 Приклади текстів і типові анотаційні ситуації

Для коректної інтерпретації результатів важливо розуміти, що приклади з тестових і діагностичних вибірок COSMUS/COSMUS+ відрізняються не лише набором ключових слів, а й прагматикою висловлювання. Positive зазвичай містить явну підтримку, схвалення, вдячність або позитивну оцінку події. Negative охоплює критику, обурення, розчарування, страх або недовіру. Neutral не означає «позитивно-негативний баланс», а радше відсутність вираженої авторської оцінки. Mixed використовується тоді, коли в межах одного повідомлення співіснують самостійні позитивні й негативні компоненти.

У соціальних медіа межа між класами часто залежить від контексту. Наприклад, коротка реакція «Так и есть» може бути позначена як positive, якщо вона виражає погодження з позитивною оцінкою в попередньому повідомленні. Однак без зовнішнього контексту така фраза майже не містить явних сентиментних маркерів, тому моделі схильні відносити її до neutral. Це демонструє одну з фундаментальних проблем корпусів коротких коментарів: анотація може спиратися на контекст дискусії, тоді як модель отримує лише сам текст.

Інший тип складності пов'язаний із непрямою негативністю. Повідомлення «В Украине цена не уменьшается если нефть падает, она просто не увеличивается» не містить грубо негативної лексики, але виражає критичну оцінку економічної ситуації. Моделі часто класифікують такі тексти як neutral, оскільки поверхнева форма є майже фактологічною. Для людини негативність відновлюється через знання сценарію: якщо ціна не зменшується за сприятливих умов, це подається як проблема або критика.

Окрема група прикладів стосується іронії та мовної гри. Фраза «Русский диалект укрмовы сойдёт?» є негативно забарвленою, хоча не містить прямого слова на кшталт «погано» або «жахливо». Негативний

сигнал виникає через саркастичну форму питання, мовну деформацію та контекст дискусії про мову. Такі приклади особливо важливі для код-змішування, бо оцінка може бути закладена не в окремому sentiment-token, а в самому способі змішування мов.

Для mixed типовими є повідомлення з протиставленням або багатокомпонентною оцінкою. У довших Telegram-коментарях автор може одночасно підтримувати загальну мету дії та критикувати її виконання; дякувати конкретним людям, але висловлювати недовіру до інституцій; визнавати позитивний результат, але наголошувати на негативних наслідках. Саме такі повідомлення мають високу цінність для аналізу суспільних настроїв, тому що вони фіксують амбівалентну позицію, яку не можна коректно звести до positive або negative.

З погляду EDA ці приклади пояснюють, чому простий підрахунок класів недостатній. Дисбаланс показує, що mixed рідкісний, але не пояснює, чому він складний. Якісний аналіз показує, що складність виникає з поєднання малої представленості класу у датасеті, довшої структури текстів, прагматичної залежності від контексту, іронії та мовного перемикання. Тому подальше оцінювання моделей у розділі 3 розглядає не лише загальні метрики, а й per-class F1, калібрування та показові випадки.

4.2 Оцінка комерційних baseline-показників

Оцінювання якості комерційних мовних моделей закритого доступу у форматах zero-shot та few-shot встановлює baseline-показники для подальшого порівняння. Моделі оцінюються через API без доступу до навчальної вибірки COSMUS і спираються виключно на свої внутрішні репрезентації.

У рамках етапу протестовано три моделі: DeepSeek V3.2, Gemini 3.0 Flash Preview та Kimi K2.5. Кожна модель тестувалась із чотирма промпт-стратегіями: eng_zero_shot, eng_few_shot, ukr_zero_shot, ukr_few_shot. Тестова вибірка: 1 223 тексти.

4.2.1 Результати за метрикою Macro F1

У таблиці 4.4 показано зведені результати комерційних API-моделей.

Таблиця 4.4 – Результати комерційних API-моделей

Модель	*Промпт-стратегія*	*Accuracy*	*F1 macro*	*F1 weighted*
Gemini 3.0 Flash	ukr_few_shot	0.7441	**0.6768**	**0.7421**
Gemini 3.0 Flash	ukr_zero_shot	**0.7465**	0.6746	0.7412
Gemini 3.0 Flash	eng_few_shot	0.7326	0.6728	0.7280
Gemini 3.0 Flash	eng_zero_shot	0.7237	0.6421	0.7191
Kimi K2.5	eng_few_shot	0.7081	0.6244	0.7027
Kimi K2.5	ukr_zero_shot	0.7065	0.6243	0.7047
DeepSeek V3.2	ukr_few_shot	0.7155	0.6202	0.7101
Kimi K2.5	ukr_few_shot	0.6958	0.6195	0.6921
Kimi K2.5	eng_zero_shot	0.7343	0.6142	0.7256
DeepSeek V3.2	ukr_zero_shot	0.6893	0.5836	0.6797
DeepSeek V3.2	eng_few_shot	0.6860	0.5595	0.6703
DeepSeek V3.2	eng_zero_shot	0.6819	0.5220	0.6621

Найвищу ефективність продемонструвала модель Gemini 3.0 Flash з українським few-shot промптом (Macro F1 = 0.6768). Gemini стабільно

показувала найкращі результати у всіх конфігураціях. Kimi K2.5 зайняла другу позицію (найкращий Macro F1 = 0.6244 з eng_few_shot). DeepSeek V3.2 показала найнижчі результати серед трьох моделей (найкращий Macro F1 = 0.6202 з ukr_few_shot).

Вплив промпт-стратегії. Аналіз впливу мови промпту та наявності прикладів виявив, що:

- для Gemini та DeepSeek найкращою стратегією виявився українськомовний few-shot промпт, тоді як для Kimi найефективнішим був англомовний few-shot;
- Few-shot промпти загалом покращували результати порівняно з zero-shot, особливо для DeepSeek (приріст до +10 відсоткових пунктів F1 macro);
- українськомовні промпти мали перевагу для моделей з кращою підтримкою української (Gemini, DeepSeek).

Per-class F1 аналіз (найкраща конфігурація). У таблиці 4.5 показано F1 аналіз за класами для найкращих конфігурацій комерційних API-моделей.

Таблиця 4.5 – Per-class F1 аналіз для найкращих конфігурацій комерційних API-моделей

Клас	*DeepSeek (ukr_few_shot)*	*Gemini (ukr_few_shot)*	*Kimi (eng_few_shot)*
negative	0.76	**0.80**	0.77
neutral	**0.73**	0.72	0.68
positive	0.69	**0.75**	0.70
mixed	0.31	**0.44**	0.34

Клас mixed залишається найскладнішим для всіх API-моделей, але навіть без дотренування Gemini досягає F1 = 0.44 для цього класу.

4.2.2 Інтерпретація API-baseline у контексті локальних моделей

API-baseline у цій роботі виконує роль зовнішньої верхньої планки для prompt-based класифікації без навчання на COSMUS. Gemini 3.0 Flash демонструє найкращу якість серед API-моделей, що очікувано для сильної комерційної LLM з потужною instruction-following поведінкою. Водночас результати показують, що перевага Gemini над найкращим локальним encoder не є великою: Modern-LiBERTa baseline має дуже близький Macro F1 і трохи вищу Accuracy у фінальному порівнянні. Це змінює інтерпретацію API-результату: Gemini є сильним baseline, але не робить локальне дотренування непотрібним.

Різниця між API-моделями також показує, що сам факт «велика мовна модель» не гарантує однаково високої якості для українсько-російського code-mixed sentiment analysis. Kimi K2.5 і DeepSeek V3.2 мають прийнятні результати, але помітно поступаються Gemini. Це може бути пов'язано з різною підтримкою української мови, різною instruction-tuning оптимізацією, різним рівнем стійкості до коротких соціальних текстів і різною здатністю розрізняти mixed. Тому для прикладного використання LLM у low-resource або code-mixed задачах необхідно проводити власне тестування, а не переносити загальні враження про модель з англомовних benchmark.

Промпт-стратегія впливає не однаково для всіх моделей. Для Gemini український few-shot prompt дає найкращий Macro F1, але український zero-shot також залишається дуже сильним. Це означає, що Gemini вже має достатньо якісне внутрішнє представлення задачі й добре реагує на українську інструкцію. Для інших API-моделей few-shot приклади можуть мати більший стабілізаційний ефект, оскільки вони задають формат відповіді та межі класів. Водночас few-shot не є універсальною гарантією: якщо приклади не репрезентують складні випадки mixed або сарказм, модель може покращити загальну метрику, але не вирішити головну проблему задачі.

Per-class результати API-моделей показують, що навіть найсильніша LLM без дотренування не розв'язує mixed повністю. Gemini досягає F1 mixed близько 0,44, що є найкращим API-результатом, але все ще значно нижче за F1 для negative або positive. Це важливе спостереження: широкі знання LLM і здатність працювати з інструкцією не компенсують повністю нестачу прикладів і складність амбівалентних повідомлень. Отже, mixed потребує або спеціалізованих навчальних стратегій, або додаткової ручної перевірки у практичних системах.

API-baseline також корисний як teacher-сигнал для distillation, але тут потрібна обережність. Якщо Gemini найкраще серед API-моделей розпізнає загальну тональність, її soft labels можуть містити корисну інформацію про невпевненість між класами. Проте teacher-модель теж має слабкі місця, зокрема mixed і непряму негативність. Тому distillation у CSSent не слід сприймати як автоматичне перенесення істини; це спосіб передати додаткову інформацію, ефект якої перевіряється експериментально.

Загалом API-результати підтверджують, що комерційні LLM є сильним інструментом для швидкого sentiment baseline, але вони не скасовують потребу в локальних моделях. У задачах, де важлива контрольованість, повторюваність і вартість масового інференсу, локальний encoder може бути практично привабливішим. У задачах, де важлива швидкість прототипування або відсутній розмічений корпус, API LLM може бути кращим стартовим рішенням. Саме таке розмежування використовується в подальшому порівняльному аналізі.

4.3 Результати дотренування Encoder-моделей

4.3.1 Оцінка загальної ефективності

Для локальних encoder-моделей було перевірено кілька рівнів складності: базові мультилінгвальні моделі, україноцентричні encoder-моделі та CSSent-конфігурації для XLM-RoBERTa Large та Modern-LiBERTa. Усі результати в цьому підрозділі подано для тестової вибірки COSMUS з чотирма класами тональності: positive, negative, neutral і mixed. Основною метрикою інтерпретації є Macro F1, оскільки вона однаково враховує всі класи і є чутливою до помилок на міноритарному класі mixed.

Таблиця 4.6 подає зведені результати для encoder-моделей. Найсильнішим локальним encoder baseline стала Modern-LiBERTa з Accuracy = 0,7482 і Macro F1 = 0,6742. XLM-RoBERTa Large baseline поступається Modern-LiBERTa за Macro F1, але після застосування CSSent-конфігурацій наближається до її рівня: найкраща CSSent-конфігурація для XLM-RoBERTa Large має Macro F1 = 0,6691. Базові mBERT, XLM-RoBERTa Base і UkrRoBERTa демонструють нижчі значення, що вказує на важливість як розміру/якості базової моделі, так і її мовної релевантності для українсько-російського код-змішаного контенту (рис. 4.3).

Отримані результати показують, що Modern-LiBERTa є найкращим локальним encoder baseline, тоді як CSSent найпомітніше допомагає XLM-RoBERTa Large. Це важливе спостереження: додаткові мовно-структуровані модулі не варто розглядати як універсальний спосіб покращення будь-якої моделі. Їхній ефект залежить від того, наскільки базова модель уже має сильні представлення для української, російської та змішаних фрагментів.

Таблиця 4.6 – Результати encoder-моделей і CSSent-конфігурацій на тестовій вибірці COSMUS

Модель / конфігурація	*Accuracy*	*Macro F1*	*Weighted F1*	*F1 positive*	*F1 negative*	*F1 neutral*	*F1 mixed*
Modern-LiBERTa	**0,7482**	**0,6742**	0,7449	0,72	0,81	0,73	**0,43**
XLM-RoBERTa Large CSSent lh_fl_mt_csa_kd	0,7400	0,6691	0,7389	0,74	0,80	0,72	0,42
XLM-RoBERTa Large CSSent lh_fl_csa_kd	0,7457	0,6684	0,7447	0,76	0,80	0,73	0,39
XLM-RoBERTa Large	0,7326	0,6500	0,7322	0,72	0,79	0,73	0,36
XLM-RoBERTa Base	0,6566	0,5932	0,6766	0,68	0,70	0,70	0,29
UkrRoBERTa	0,6721	0,5923	0,6716	0,65	0,73	0,67	0,32
mBERT	0,6419	0,5594	0,6533	0,65	0,66	0,70	0,22

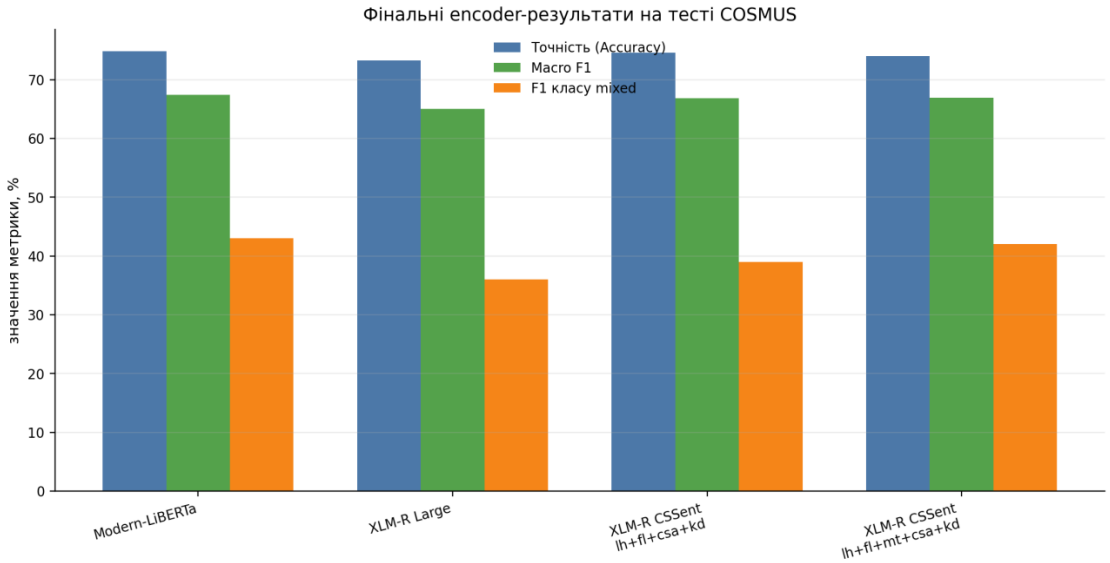

Рисунок 4.3 – Фінальні результати encoder-моделей на тестовій вибірці COSMUS

Окремо слід інтерпретувати результат UkrRoBERTa. Нижчий показник у цьому експерименті не означає, що україномовна спеціалізація загалом є неефективною. Для цієї моделі суттєвим чинником може бути конкретний режим навчання та відсутність додаткових аугментацій, які використовувалися в початкових бенчмарках COSMUS. Тому результат UkrRoBERTa в цій роботі доцільно трактувати як результат базового повного дотренування в межах єдиного протоколу, а не як остаточну оцінку всіх можливих стратегій адаптації цієї архітектури.

4.3.2 Аналіз F1 за класами

За F1 за класами видно, що головна складність задачі не зводиться до загальної точності. Для всіх encoder-моделей найстабільнішими є класи negative, neutral і positive, тоді як mixed залишається найпроблемнішим. Це очікувано, оскільки mixed має найменшу кількість прикладів і потребує розпізнавання одночасної присутності різноспрямованих оцінок.

Modern-LiBERTa має найкращий F1 mixed серед базових encoder-моделей – 0,43. XLM-RoBERTa Large baseline має F1 mixed = 0,36, але CSSent-конфігурація lh_fl_mt_csa_kd підвищує його до 0,42. Отже, CSSent не обов’язково дає найвищий загальний результат, але для XLM-RoBERTa Large покращує саме ту частину задачі, яка є найскладнішою для звичайного дотренування (табл. 4.7).

Таблиця 4.7 – F1 за класами для encoder-моделей і CSSent-конфігурацій

Модель / конфігурація	*F1 positive*	*F1 negative*	*F1 neutral*	*F1 mixed*	*Основне спостереження*
Modern-LiBERTa	0,72	0,81	0,73	0,43	найсильніший локальний encoder baseline
XLM-RoBERTa Large	0,72	0,79	0,73	0,36	стабільний на мажоритарних класах, слабший на mixed
XLM-R CSSent lh_fl_csa_kd	0,76	0,80	0,73	0,39	покращує positive і частково mixed
XLM-R CSSent lh_fl_mt_csa_kd	0,74	0,80	0,72	0,42	найкращий CSSent-зріз для mixed
UkrRoBERTa	0,65	0,73	0,67	0,32	нижча загальна якість у базовому режимі
XLM-RoBERTa Base	0,68	0,70	0,70	0,29	обмеження меншої encoder-основи
mBERT	0,65	0,66	0,70	0,22	найслабший результат на mixed

Найважливіший висновок із цього зрізу полягає в тому, що mixed потрібно аналізувати окремо. Модель може мати прийнятну Accuracy за рахунок negative і neutral, але все ще майже не розпізнавати mixed. Саме тому Macro F1 і per-class F1 мають більшу пояснювальну цінність для цієї задачі, ніж одна загальна точність.

4.3.3 Матриця помилок (Confusion Matrix Analysis)

Матриці помилок використано для уточнення того, які саме класи змішуються між собою під час класифікації. Для цієї задачі найбільш інформативними є помилки, у яких справжній mixed прогнозується як negative, neutral або positive.

Для порівняння обрано дві ключові encoder-моделі: Modern-LiBERTa baseline як найсильнішу локальну базову encoder-модель і XLM-RoBERTa Large CSSent як приклад запропонованого комплексного підходу. Матриці побудовано на однаковій тестовій вибірці COSMUS із 1 223 прикладів і чотирма класами: positive, negative, neutral і mixed (рис. 4.4).

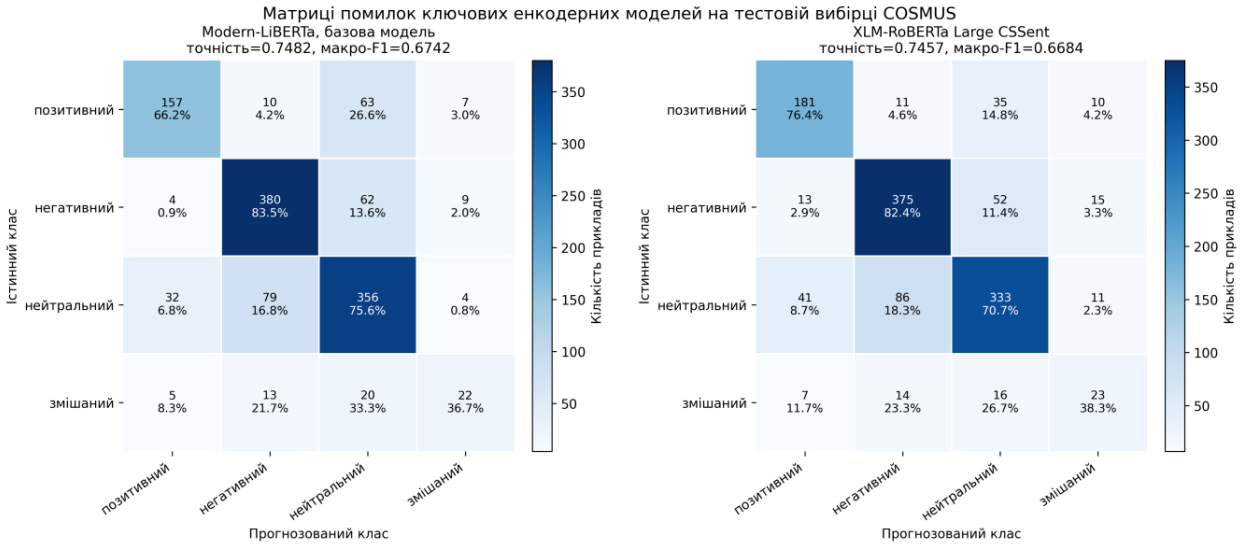

Рисунок 4.4 – Матриці помилок для Modern-LiBERTa baseline та XLM-RoBERTa Large CSSent на тестовій вибірці COSMUS

Матриці помилок показують, що основна складність обох моделей пов'язана з класом mixed. Для Modern-LiBERTa baseline правильно класифіковано 22 із 60 mixed-прикладів, тоді як 20 mixed-прикладів перейшли в neutral, 13 – у negative і 5 – у positive. Для XLM-RoBERTa Large

CSSent правильно класифіковано 23 із 60 mixed-прикладів; кількість переходів mixed у neutral зменшується до 16, але частина прикладів усе ще прогнозується як negative або positive. Це підтверджує, що mixed залишається найскладнішою категорією: модель часто розпізнає один домінантний оцінний компонент, але не завжди фіксує співіснування позитивної та негативної оцінки в одному повідомленні.

4.3.4 Оцінка запропонованого комплексного підходу CSSent

Результати CSSent потрібно інтерпретувати відносно базової моделі. Для XLM-RoBERTa Large базовий Macro F1 становить 0,6500, тоді як конфігурації lh_fl_csa_kd і lh_fl_mt_csa_kd підвищують результат до 0,6684 і 0,6691 відповідно. Найпомітніше це проявляється на mixed: F1 зростає з 0,36 до 0,39-0,42.

Схема впливу складових запропонованого підходу CSSent подана на рис. 4.5. На рисунку показано зміну Macro F1 у відсоткових пунктах відносно відповідного baseline для двох encoder-основ. Додатні значення означають покращення, від’ємні – погіршення. Для XLM-RoBERTa Large більшість CSSent-комбінацій дає додатний ефект, тоді як для Modern-LiBERTa ті самі модулі частіше погіршують результат. Це підтверджує, що CSSent є не готовою “надбудовою для всього”, а інструментом, ефективність якого залежить від базової моделі.

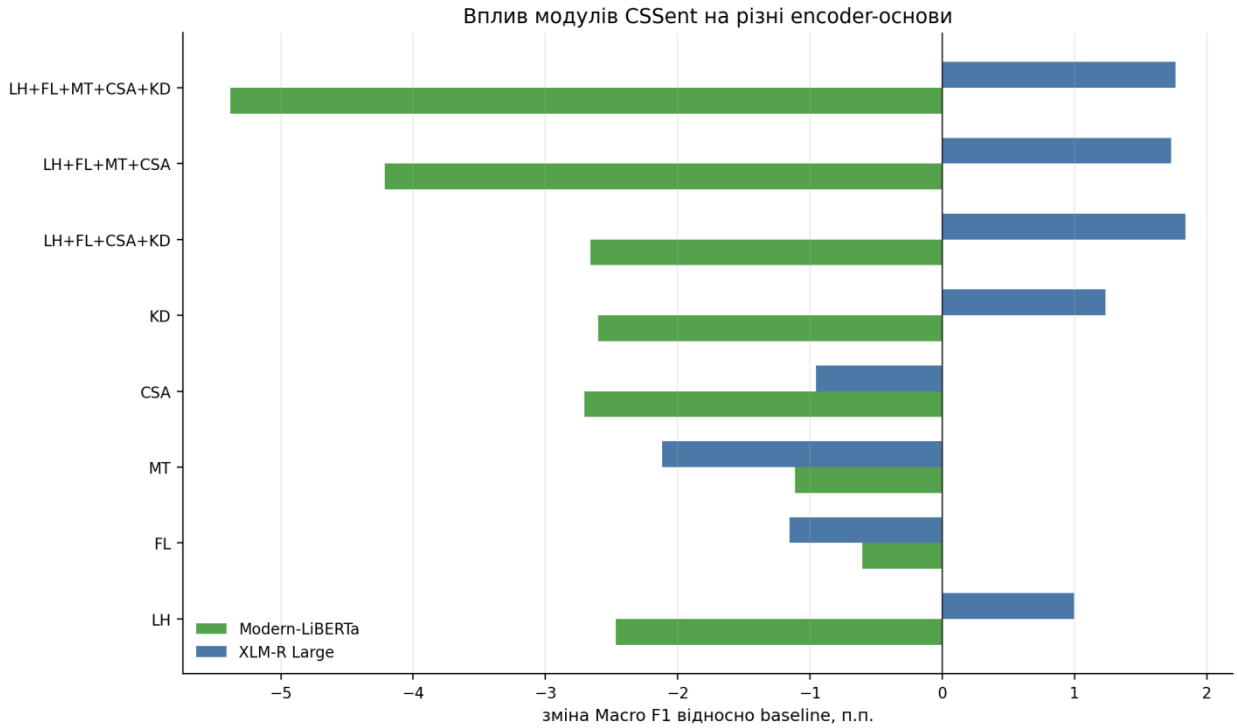

Рисунок 4.5 – Зміна Macro F1 відносно baseline для окремих складових CSSent

Аугментація csa має найбільшу потенційну користь для рідкісних і складних ситуацій, але вона також є ризикованою. Якщо аугментовані приклади занадто відрізняються від природного Telegram-дискурсу, модель може навчитися штучних шаблонів. Саме тому ефект csa оцінювався через абляційний аналіз, а не приймався як гарантоване покращення. У фінальних XLM-RoBERTa результатах комбінації з csa і kd дали корисний ефект, але це не означає, що будь-яка аугментація код-змішаних прикладів автоматично покращує якість.

4.4 Результати адаптації Decoder-моделей (MamayLM)

Окремим етапом дослідження стало тестування україноцентричної decoder-only моделі MamayLM v1.0. Було перевірено два режими адаптації: prompting без оновлення ваг і QLoRA-дотренування. Prompting оцінювався для MamayLM 4B і 12B, тоді як QLoRA також було виконано для обох розмірів моделі. Це дозволило відокремити два питання: наскільки добре MamayLM класифікує тональність лише за інструкцією та наскільки якість змінюється після параметрично-ефективного навчання на COSMUS.

4.4.1 Prompting MamayLM як діагностика інструктивної поведінки

MamayLM оцінювалася у двох різних режимах: інструктивна класифікація через промптинг і параметрично-ефективне дотренування QLoRA. Промптинг у цьому підрозділі розглядається не як фінальний найсильніший підхід, а як діагностика того, наскільки локальна decoder-only LLM здатна виконувати класифікацію без зміни ваг моделі. Для 4B і 12B перевірялися українські та англійські zero-shot/few-shot промпти (табл. 4.8).

Таблиця 4.8 – Результати промптингу MamayLM 4B і 12B на тестовій вибірці COSMUS

Модель	*Режим*	*Accuracy*	*Macro F1*
MamayLM 4B	eng zero-shot	0,5151	0,3521
MamayLM 4B	ukr zero-shot	0,5020	0,3591
MamayLM 4B	eng few-shot	0,4612	0,4161
MamayLM 4B	ukr few-shot	0,4383	0,4157
MamayLM 12B	eng zero-shot	**0,6942**	**0,5870**
MamayLM 12B	ukr zero-shot	0,6680	0,5674
MamayLM 12B	eng few-shot	0,6541	0,5446
MamayLM 12B	ukr few-shot	0,6460	0,5563

У цій таблиці наведено лише ті метрики, які є повністю зіставними для всіх prompting-конфігурацій.

Промптинг MamayLM 4B показав низьку якість: найкращий Macro F1 становив 0,4161 для англійського few-shot промпту, а повний per-class-звіт для українського few-shot режиму показує F1 mixed = 0,14. Це означає, що модель 4B без дотренування не формує достатньо стабільного правила для відокремлення mixed від інших класів.

MamayLM 12B суттєво краща в prompting-сценарії: найвищий Macro F1 становить 0,5870 для англійського zero-shot промпту. Водночас 12B не перетворюється на повноцінний класифікатор лише завдяки більшому розміру: F1 mixed лишається в діапазоні 0,1818-0,2439. Отже, збільшення кількості параметрів допомагає загальній класифікації, але не розв’язує проблему амбівалентних повідомлень.

4.4.2 QLoRA як перехід від генеративної моделі до класифікатора

QLoRA-дотренування змінює роль MamayLM: замість інструктивного генератора, який має з текстової інструкції самостійно вивести назву класу, модель навчається як спеціалізований класифікатор для фіксованого набору міток. Це суттєво підвищує якість і для 4B, і для 12B (табл. 4.9).

Таблиця 4.9 – Результати MamayLM 4B і 12B після QLoRA-дотренування

Модель	*Accuracy*	*Macro F1*	*Weighted F1*	*F1 positive*	*F1 negative*	*F1 neutral*	*F1 mixed*
MamayLM 4B QLoRA	0,7588	0,6555	0,7494	0,7273	0,8188	0,7469	0,3291
MamayLM 12B QLoRA	**0,7604**	**0,6654**	0,7516	0,7299	0,8218	0,7439	**0,3659**

Після QLoRA MamayLM 12B досягає Accuracy = 0,7604 і Macro F1 = 0,6654, а MamayLM 4B – Accuracy = 0,7588 і Macro F1 = 0,6555. Різниця між 4B і 12B після дотренування є меншою, ніж у prompting-сценарії. Це свідчить, що спеціалізоване навчання на задачі може частково компенсувати менший розмір моделі.

Клас mixed залишається найслабшим і після QLoRA: F1 mixed становить 0,3291 для 4B і 0,3659 для 12B. Це значно краще, ніж prompting, але нижче за Modern-LiBERTa baseline і найкращі XLM-R CSSent-конфігурації. Отже, QLoRA перетворює MamayLM на конкурентний локальний класифікатор за загальною точністю, проте не усуває головну проблему задачі – надійне розпізнавання амбівалентного сентименту.

4.4.3 Порівняння промптингу та QLoRA для MamayLM

Окреме порівняння промптингу й QLoRA показує, що вирішальним чинником для MamayLM є не лише розмір моделі, а саме адаптація до задачі. Для 4B перехід від найкращого prompting-режиму до QLoRA підвищує Macro F1 приблизно з 0,4161 до 0,6555. Для 12B приріст менший, але все одно суттєвий: від 0,5870 до 0,6654 (табл. 4.10).

Таблиця 4.10 – Порівняння промптингу та QLoRA для MamayLM

Модель	*Найкращий prompting Macro F1*	*QLoRA Macro F1*	*Приріст Macro F1*	*Доступний prompting F1 mixed*	*QLoRA F1 mixed*
MamayLM 4B	0,4161	0,6555	+0,2394	0,14	0,3291
MamayLM 12B	0,5870	0,6654	+0,0784	0,2439	0,3659

Для MamayLM 4B значення F1 mixed у таблиці 4.9 подано за українським few-shot режимом; за Macro F1 він практично збігається з найкращим англійським few-shot режимом. Для MamayLM 12B F1 mixed подано для англійського zero-shot режиму, який має найвищий Macro F1 серед prompting-конфігурацій 12B.

З рисунка 4.6 видно, що після QLoRA обидві MamayLM-моделі мають сильні результати на negative, neutral і positive, але mixed лишається нижчим за інші класи. Різниця між 4B і 12B на mixed є помітною, але не радикальною: 12B краще виявляє mixed, проте все ще часто зводить амбівалентні приклади до одного домінантного полюса.

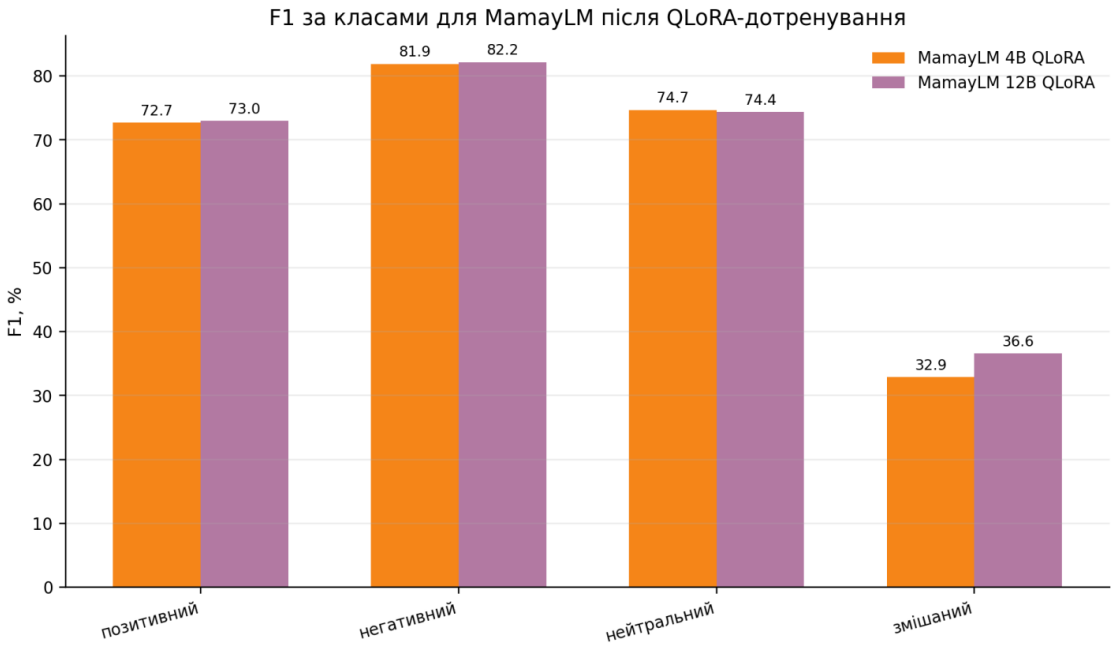

Рисунок 4.6 – F1 за класами для MamayLM 4B і MamayLM 12B після QLoRA-дотренування

Практично це означає, що MamayLM після QLoRA можна розглядати як локальну альтернативу encoder-класифікаторам у сценаріях, де важливі автономність і контроль над розгортанням. Водночас для задач, де критичним є саме mixed, краще додатково використовувати аналіз невпевненості, повторну перевірку складних прикладів або ансамблювання з encoder-моделлю.

4.5 Порівняльний аналіз

4.5.1 Зведене порівняння усіх підходів

Таблиця 4.11 подає фінальні релевантні конфігурації, які оцінювалися на тестовій вибірці COSMUS. До таблиці включено найкращі API-конфігурації для кожного провайдера, основні локальні encoder-моделі, CSSent-конфігурації, MamayLM після QLoRA та показові prompting-режими

MamayLM 4B/12B. Окремі проміжні або дубльовані результати не подаються, щоб зведена таблиця містила лише повністю зіставні метрики.

Таблиця 4.11 – Зведене порівняння основних підходів на тестовій вибірці COSMUS

Підхід	*Модель / конфігурація*	*Accuracy*	*Macro F1*	*Weighted F1*	*F1 mixed*
API LLM	Gemini 3.0 Flash, ukr few-shot	0,7441	**0,6768**	0,7421	**0,44**
Encoder	Modern-LiBERTa	0,7482	0,6742	0,7449	0,43
CSSent	XLM-R Large lh_fl_mt_csa_kd	0,7400	0,6691	0,7389	0,42
CSSent	XLM-R Large lh_fl_csa_kd	0,7457	0,6684	0,7447	0,39
QLoRA	MamayLM 12B QLoRA	**0,7604**	0,6654	0,7516	0,3659
QLoRA	MamayLM 4B QLoRA	0,7588	0,6555	0,7494	0,3291
Encoder	XLM-RoBERTa Large	0,7326	0,6500	0,7322	0,36
API LLM	Kimi K2.5, eng few-shot	0,7081	0,6244	0,7027	0,34
API LLM	DeepSeek V3.2, ukr few-shot	0,7155	0,6202	0,7101	0,31
Encoder	XLM-RoBERTa Base	0,6566	0,5932	0,6766	0,29
Encoder	UkrRoBERTa	0,6721	0,5923	0,6716	0,32
Prompting	MamayLM 12B, eng zero-shot	0,6942	0,5870	0,6848	0,2353
Encoder	mBERT	0,6419	0,5594	0,6533	0,22
Prompting	MamayLM 4B, ukr few-shot	0,4383	0,4157	0,49	0,14

Зі зведеного порівняння видно, що найкращі підходи розташовані дуже близько за Macro F1. Gemini 3.0 Flash має найвищий Macro F1 = 0,6768, але Modern-LiBERTa відстає мінімально й досягає 0,6742 без звернення до зовнішнього API. CSSent для XLM-RoBERTa Large наближає мультилінгвальну encoder-модель до цього рівня, а MamayLM після QLoRA демонструє конкурентну Accuracy, хоча поступається на mixed.

Для MamayLM 4B у зведеній таблиці подано український few-shot режим, а не англійський few-shot, хоча останній має трохи вищий Macro F1 (0,4161 проти 0,4157). Різниця за Macro F1 між цими двома prompting-режимами є мінімальною, тому такий вибір не змінює змістовного висновку: MamayLM 4B у режимі промптингу суттєво поступається QLoRA-адаптації та encoder-моделям.

Таке порівняння показує, що для цієї задачі немає однозначно найкращої моделі за всіма критеріями. Якщо важлива максимальна Macro F1 без локального розгортання, найсильнішою є Gemini 3.0 Flash. Якщо важливі локальність, контроль і відтворюваність, Modern-LiBERTa є найкращим baseline. Якщо метою є перевірка мовно-структурованих модулів, найінформативнішим є CSSent на XLM-RoBERTa Large. Якщо потрібна локальна decoder-only LLM, MamayLM після QLoRA є значно сильнішою за prompting.

На рисунку 4.7 порівнюються основні підходи за Macro F1.

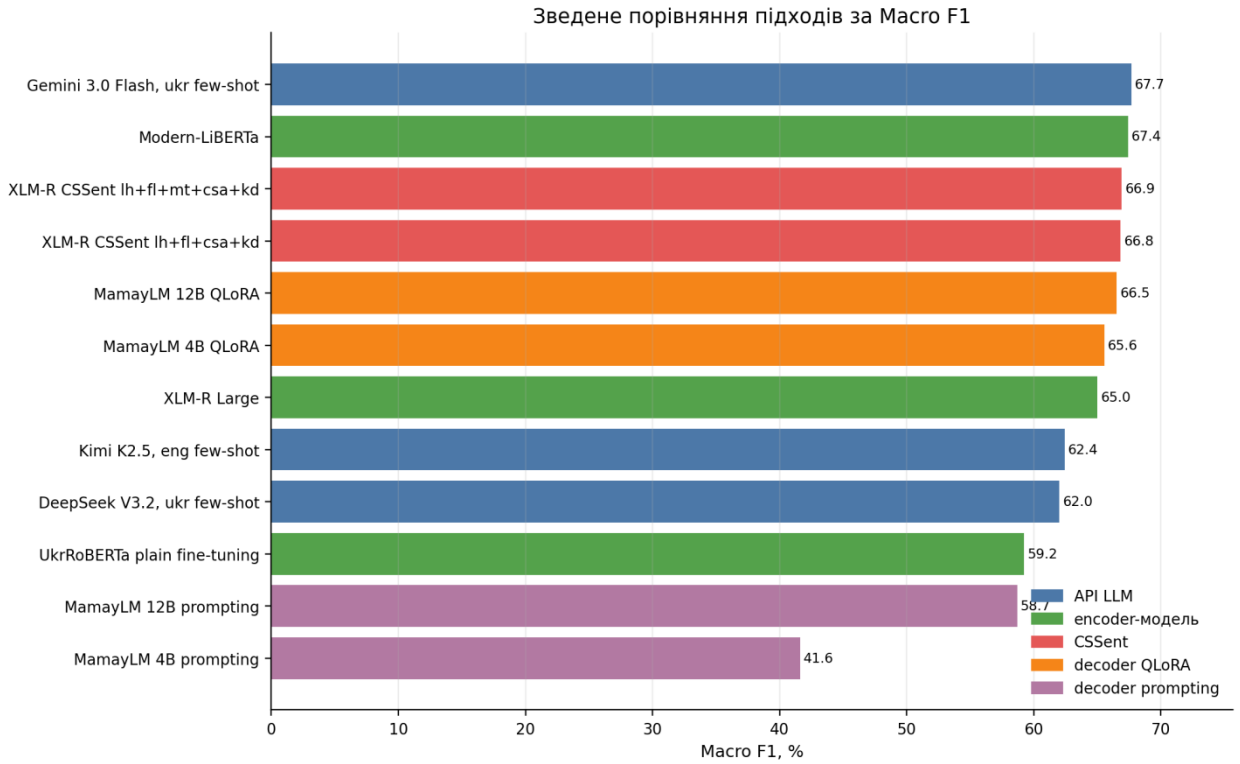

Рисунок 4.7 – Зведене порівняння найкращих досліджених підходів за Macro F1

4.5.2 Головні спостереження

Перший важливий результат полягає в тому, що локальна Modern-LiBERTa майже досягає рівня найкращої API-моделі. Різниця між Gemini 3.0 Flash і Modern-LiBERTa за Macro F1 становить приблизно 0,0026, що є малою різницею порівняно з перевагами локального розгортання: контроль над даними, відтворюваність і незалежність від зовнішнього сервісу.

Подібний компроміс між компактною локальною моделлю та великою LLM для української мови було продемонстровано і в суміжній задачі розпізнавання іменованих сутностей: у роботі [38] модифікована GLiNER-архітектура зі стиснутим кодувальником і крос-увагою досягає конкурентної

з LLM якості у zero-shot і few-shot режимах при суттєво менших обчислювальних вимогах, що додатково підтверджує доцільність орієнтації на локальні спеціалізовані моделі для українськомовних NLP-задач.

Другий результат стосується CSSent. Для XLM-RoBERTa Large спеціалізовані модулі справді покращують Macro F1 відносно baseline, але цей ефект не слід переносити на всі encoder-моделі. Висновок роботи полягає не в тому, що CSSent завжди кращий за звичайне дотренування, а в тому, що мовно-структуровані сигнали можуть бути корисними для певних базових моделей і мають перевірятися абляційно.

Третій результат пов’язаний із MamayLM. Промптинг локальної decoder-only моделі суттєво поступається повному дотренуванню encoder-моделей, але QLoRA різко зменшує цей розрив. Після QLoRA MamayLM 12B досягає Macro F1 = 0,6654 і стає конкурентною з XLM-RoBERTa Large baseline за загальними метриками. Водночас її F1 mixed залишається нижчим, ніж у Modern-LiBERTa та найкращих CSSent-конфігурацій.

Окремо варто підкреслити, що Accuracy не є достатньою метрикою для цієї задачі. MamayLM 12B QLoRA має найвищу Accuracy серед наведених підходів, але не є найкращою за Macro F1 і mixed. Це типовий наслідок дисбалансу: модель може добре працювати на мажоритарних класах і водночас гірше розпізнавати найскладнішу міноритарну категорію.

4.5.3 Порівняння F1 для класу mixed

Клас mixed є центральним діагностичним класом у роботі, оскільки він поєднує два джерела складності: низьку представленість у даних і семантичну амбівалентність. Таблиця 4.12 узагальнює F1 mixed для основних підходів.

Таблиця 4.12 – Порівняння F1 для класу mixed

Модель / конфігурація	*F1 mixed*
Gemini 3.0 Flash, ukr few-shot	0,44
Modern-LiBERTa	0,43
XLM-R CSSent lh_fl_mt_csa_kd	0,42
XLM-R CSSent lh_fl_csa_kd	0,39
MamayLM 12B QLoRA	0,3659
XLM-RoBERTa Large	0,36
Kimi K2.5, eng few-shot	0,34
MamayLM 4B QLoRA	0,3291
UkrRoBERTa	0,32
DeepSeek V3.2, ukr few-shot	0,31
XLM-RoBERTa Base	0,29
MamayLM 12B, ukr zero-shot	0,2439
mBERT	0,22
MamayLM 4B, ukr few-shot	0,14

Найкращі значення mixed мають Gemini 3.0 Flash і Modern-LiBERTa. Це означає, що сильна загальна модель і сильний локальний україноцентричний encoder можуть подібно працювати з амбівалентними повідомленнями. CSSent для XLM-RoBERTa Large також істотно покращує mixed відносно базового XLM-RoBERTa Large. Натомість prompting MamayLM, особливо у 4B-версії, майже не розв’язує проблему mixed.

Додатковий аналіз помилок за довжиною тексту. Довжина тексту впливає на складність класифікації неоднозначно. Дуже короткі повідомлення часто містять недостатньо контексту для розрізнення neutral, positive або sarcastic negative, наприклад короткі відповіді на попереднє повідомлення. Довгі тексти, навпаки, можуть містити кілька аспектів, цитати,

перелік фактів і різні емоційні фрагменти, через що модель має вирішити, чи є повідомлення mixed, чи один полюс домінує над іншим.

Тому аналіз довжини слід використовувати як допоміжну діагностику. Помилка на короткому повідомленні часто означає нестачу контексту, а помилка на довгому повідомленні частіше пов’язана з багатотемністю й аспектною структурою. Це добре узгоджується з case-study аналізом, де серед складних прикладів є як дуже короткі репліки на кшталт “Так і є”, так і довгі тексти з одночасною критикою та підтримкою.

4.5.4 Калібрування ймовірностей моделей

Калібрування оцінює не те, скільки прикладів модель класифікувала правильно, а наскільки її прогнозована впевненість відповідає фактичній точності. Якщо модель часто прогнозує клас із впевненістю 0,9, але правильно відповідає лише в 0,7 випадків, вона є надмірно впевненою. У таблиці 4.13 подано Expected Calibration Error для фінальних encoder-моделей.

Таблиця 4.13 – Expected Calibration Error для фінальних encoder-моделей

Модель	*ECE@10*
XLM-RoBERTa Large	0,1744
Modern-LiBERTa	0,1803
XLM-R CSSent lh_fl_mt_csa_kd	0,2278
XLM-R CSSent lh_fl_csa_kd	0,2294

На рис. 4.8 подано діаграми надійності для фінальних encoder-моделей. Вісь X відповідає передбаченій впевненості моделі, вісь Y – емпіричній

точності в кожному інтервалі впевненості. Пунктирна діагональ означає ідеальне калібрування: якщо стовпчик лежить нижче діагоналі, модель є надмірно впевненою, а якщо вище – недостатньо впевненою. Позначення ECE@10 означає, що інтервал впевненості поділено на 10 бінів.

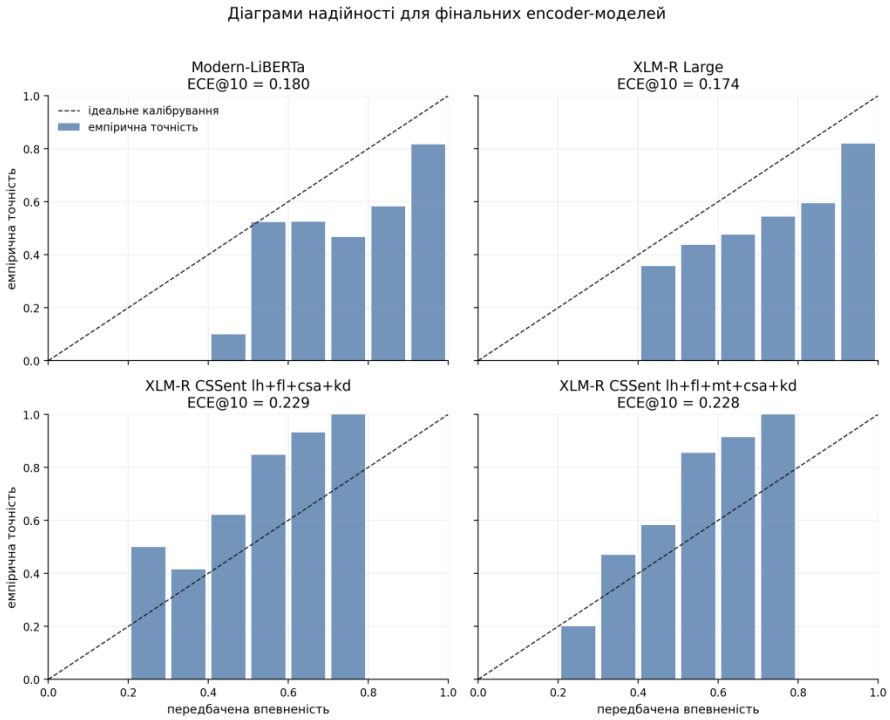

Рисунок 4.8 – Діаграми надійності для фінальних encoder-моделей на тестовій вибірці COSMUS

За ECE видно, що CSSent-конфігурації для XLM-RoBERTa Large мають гірше калібрування, ніж базові encoder-моделі, попри кращий Macro F1. Це важливий практичний висновок: модель може краще класифікувати приклади, але менш надійно оцінювати власну впевненість. Тому для сценаріїв, де ймовірність використовується як сигнал для подальшої ручної перевірки або порогового відбору, потрібно додатково застосовувати калібрування або обережніше інтерпретувати confidence.

4.5.5 Пояснюваність прогнозів: LIME та SHAP

LIME і SHAP використано як локальні методи пояснюваності для аналізу того, які токени мали найбільший внесок у прогнози моделей. Оскільки обидва методи мають різні припущення, корисним є не лише список top-термінів кожного методу окремо, а й їхній перетин. Якщо LIME і SHAP незалежно виділяють однакові слова для певного класу, це підсилює довіру до того, що модель справді чутлива до цих лексичних маркерів.

На рис. 4.9 показано кількість спільних top-термінів LIME і SHAP для Modern-LiBERTa та XLM-R CSSent. Для XLM-R CSSent перетин є більшим у всіх класах, особливо для positive і mixed. Це означає, що два методи пояснюваності частіше погоджуються щодо важливих токенів цієї моделі. Водночас більший перетин не означає автоматично кращу якість: він показує узгодженість пояснень, а не правильність прогнозу (табл. 4.14).

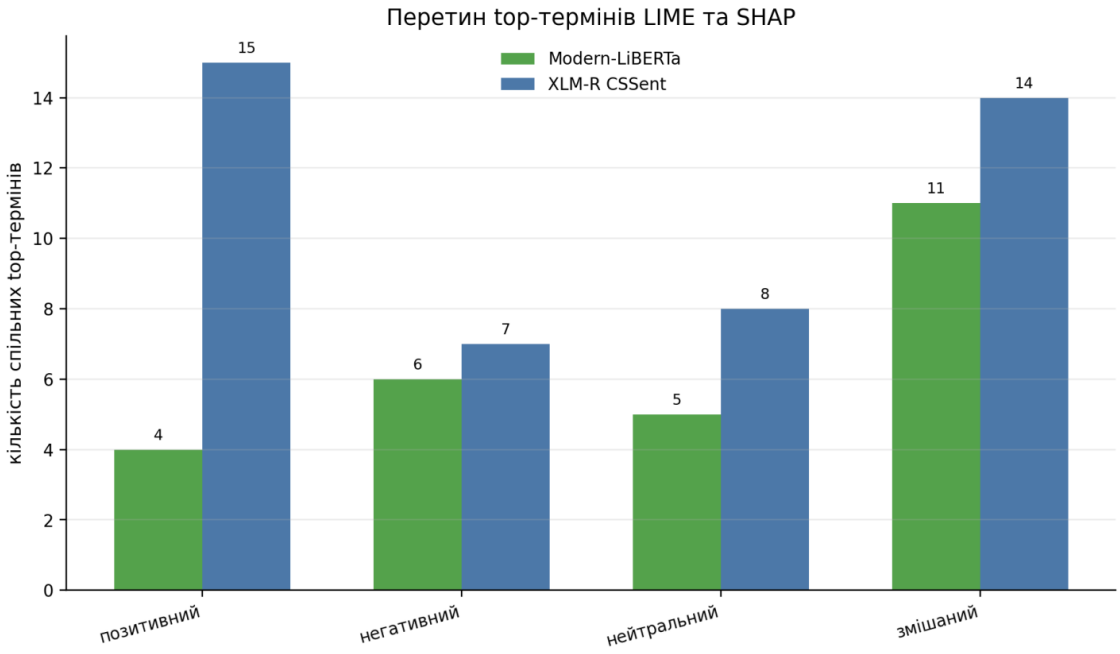

Рисунок 4.9 – Перетин top-термінів LIME та SHAP для Modern-LiBERTa і XLM-R CSSent

Таблиця 4.14 – Кількість спільних top-термінів LIME та SHAP за класами

Модель	*positive*	*negative*	*neutral*	*mixed*
Modern-LiBERTa	4	6	5	11
XLM-R CSSent	15	7	8	14

Для класу mixed серед спільних термінів Modern-LiBERTa трапляються «дякую», «круто», «але», «нажаль», «нах», «евакуації», «попередили», «довго». Частина цих слів є очевидними емоційними маркерами, а частина стає інформативною лише в контексті. Наприклад, «але» часто позначає зміну полярності всередині повідомлення, а поєднання позитивних слів на кшталт «дякую» або «круто» з негативними маркерами може сигналізувати mixed.

Для XLM-R CSSent у mixed перетинаються «дякую», «супер», «круто», «мінус», «але», «не», «добрий», «дія». Це добре узгоджується з ідеєю mixed як класу, де позитивні й негативні сигнали співіснують. Водночас наявність таких слів не є достатньою умовою mixed: слово «супер» може бути частиною чисто позитивного повідомлення, а «не» – нейтральної конструкції. Тому LIME/SHAP слід трактувати як підказку для аналізу, а не як правило класифікації.

Для positive XLM-R CSSent виділяє «хорошо», «супер», «ідеально», «спасибо», «норм», «приємний», «шедевр», «дуже». Для negative у Modern-LiBERTa з’являються «хватит», «трудом», «долго», «не». Ці приклади показують, що моделі спираються не лише на українські, а й на російські та змішані маркери, що є очікуваним для корпусу COSMUS.

4.5.6 Вплив COSMUS+ на якість моделей

Окремий блок експериментів перевіряв, чи покращує навчання на розширеному корпусі COSMUS+ якість на тестовій вибірці COSMUS+. Це порівняння не слід змішувати з основним COSMUS test, оскільки COSMUS+ має інший розподіл і використовується для перевірки стійкості моделей на розширених даних.

Рисунок 4.10 порівнює дві умови навчання для Modern-LiBERTa та XLM-RoBERTa Large: навчання лише на COSMUS і навчання на COSMUS+. Візуально рисунок підкреслює, що розширення корпусу покращує не тільки загальну точність, а й Macro F1 на новому тестовому розподілі (табл. 4.15).

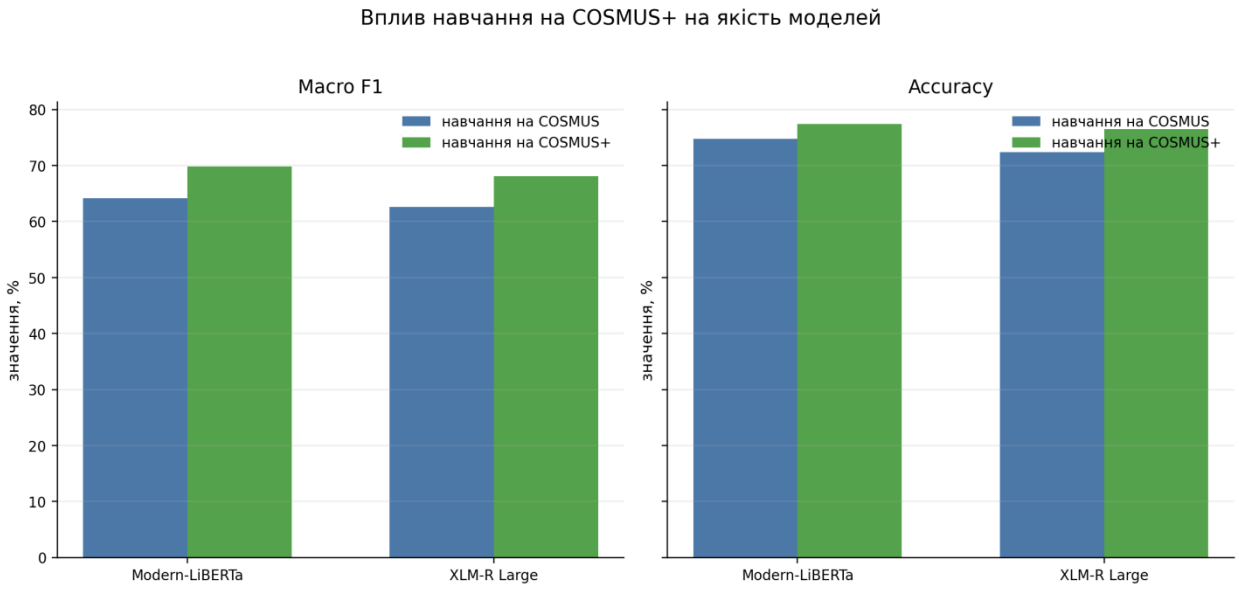

Рисунок 4.10 – Вплив навчання на COSMUS+ на якість Modern-LiBERTa та XLM-R Large

Таблиця 4.15 – Вплив навчання на COSMUS+ на якість моделей

Модель	*Навчальний корпус*	*Тестовий корпус*	*Accuracy*	*Macro F1*	*Weighted F1*
Modern-LiBERTa	COSMUS	COSMUS+ test	0,7471	0,6419	0,7411
Modern-LiBERTa	COSMUS+	COSMUS+ test	**0,7745**	**0,6981**	0,7723
XLM-RoBERTa Large	COSMUS	COSMUS+ test	0,7234	0,6263	0,7243
XLM-RoBERTa Large	COSMUS+	COSMUS+ test	0,7646	0,6808	0,7713

Навчання на COSMUS+ покращує обидві моделі. Для Modern-LiBERTa Macro F1 зростає з 0,6419 до 0,6981, а для XLM-RoBERTa Large – з 0,6263 до 0,6808. Найважливішим є те, що покращення стосується не лише мажоритарних класів. На підмножині справжніх mixed-прикладів Accuracy Modern-LiBERTa зростає з 0,3442 до 0,4805, а XLM-RoBERTa Large – з 0,4221 до 0,5390.

Таким чином, COSMUS+ виконує не лише роль збільшеного корпусу, а й роль перевірки стійкості. Результати показують, що для код-змішаного сентимент-аналізу додаткові релевантні дані можуть бути не менш важливими за ускладнення архітектури моделі.

4.5.7 Якісний аналіз показових випадків

Розбір показових випадків використано для пояснення тих помилок, які не видно з агрегованих метрик. У цьому аналізі приклади групувалися за рівнем згоди моделей: випадки, де всі моделі помилилися; випадки, де правильною була лише одна модель; випадки з розбіжністю прогнозів; і приклади mixed, де перевірялася здатність моделей розпізнавати амбівалентну тональність.

На рис. 4.11 показано структуру відібраних case studies за часткою коректних моделей. Це не розподіл усього датасету, а саме діагностична вибірка складних прикладів. Категорія “усі моделі помилилися” має 0% правильних прогнозів; “одна модель коректна” зазвичай відповідає приблизно 25% для чотирьох моделей або близько 4% для випадків із ширшим набором моделей; “розбіжність моделей” має широкий діапазон; а приклади mixed часто мають високу, але не ідеальну частку правильних прогнозів (табл. 4.16).

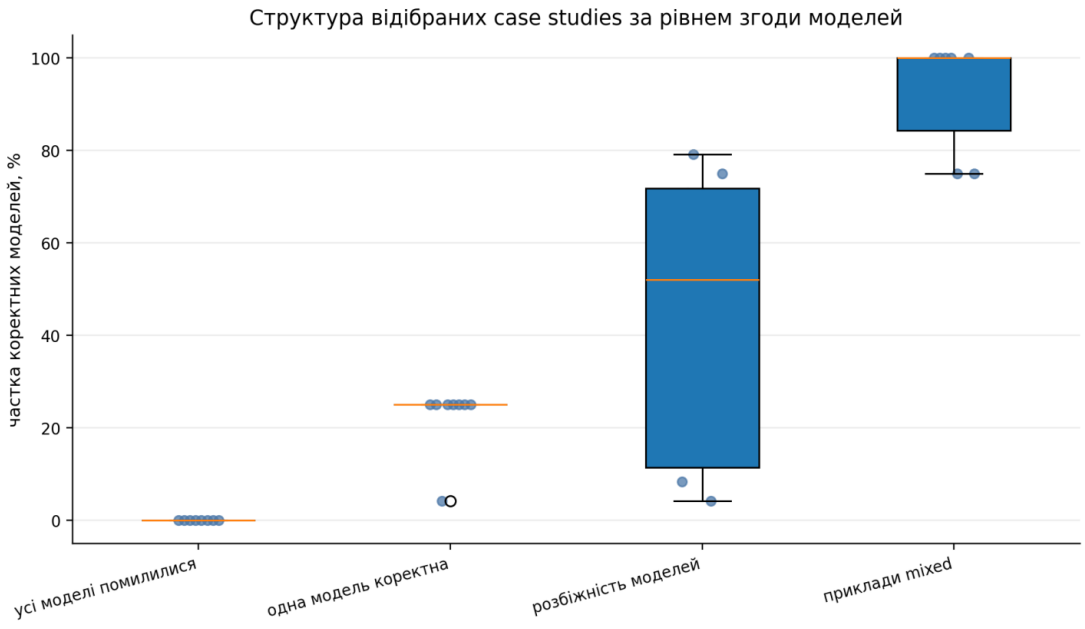

Рисунок 4.11 – Структура відібраних показових випадків за рівнем згоди моделей

Таблиця 4.16 – Приклади показових випадків і якісна інтерпретація помилок

Тип випадку	*Текстовий фрагмент*	*Істинний клас*	*Поведінка моделей*	*Інтерпретація*
Усі моделі помилилися	В Украине цена не уменьшается если нефть падает, она просто не увеличивается.	negative	усі прогнозують neutral	Негативність виражена непрямо через іронічну економічну оцінку, без явної лайки або емоційного прикметника.
Усі моделі помилилися	Так и есть	positive	усі прогнозують neutral	Коротка репліка залежить від попереднього контексту; без нього модель бачить майже нейтральне підтвердження.
Усі моделі помилилися	Русский диалект укрмовы сойдёт?	negative	усі прогнозують neutral	Негативність походить із саркастичного мовного коментаря, а не з очевидного sentiment-слова.
Усі моделі помилилися	У нас могут быть жертвы ... но мы делаем это ради будущего	mixed	моделі зводять до neutral або positive	Є негативний компонент про жертви та позитивно/виправдувальний компонент про майбутнє; моделі часто вибирають один домінантний аспект.
Mixed, переважно правильно	Коли введуть эповістки ... Але зараз все круто.	mixed	21 із 24 прогнозів правильні	Сигнал mixed явний: негативна умова до але і позитивна оцінка після нього.
Mixed, правильно	Ситуація з дискваліфікацією ... це ганьба ... Хочу висловити підтримку...	mixed	усі основні encoder-моделі прогнозують mixed	Текст містить чітку негативну оцінку рішення і позитивну підтримку людини, тому mixed розпізнається стабільніше.

Ці приклади демонструють, що складність mixed не завжди полягає в наявності двох очевидних sentiment-слів. Іноді амбівалентність задається структурою повідомлення: протиставленням через але, переходом від критики до підтримки або поєднанням фактичного опису з авторською оцінкою. Для коротких реплік проблема інша: без попереднього контексту модель не може знати, чи фраза є схваленням, сарказмом або нейтральною відповіддю.

Окремо аналізувалися агреговані thinking-метадані для Gemini-запусків. Raw thinking-тексти не використовувалися як якісні пояснення; аналіз обмежувався довжиною та наявністю службових thinking-полів. У prediction-файлах Gemini покриття thinking-метаданими було майже повним, а неправильні прогнози мали більшу середню довжину thinking-поля, ніж правильні: приблизно 1599,7 проти 1374,6 символів. Це не означає, що довше “міркування” є причиною помилки. Коректніша інтерпретація така: складніші приклади частіше породжують довші службові метадані, але це не гарантує правильної класифікації.

За класами mixed мав найнижчу доступну accuracy у thinking-зрізі – приблизно 0,3667 – і найбільшу середню кількість thinking-токенів серед класів. Це узгоджується з основним висновком роботи: mixed є не лише статистично рідкісним, а й семантично складним класом, для якого навіть сильні LLM частіше демонструють невпевненість або помилки.

Обмеження експериментального порівняння. Результати роботи слід інтерпретувати в межах використаних корпусів і тестових протоколів. COSMUS і COSMUS+ охоплюють українсько-російський код-змішаний Telegram-дискурс, але не вичерпують усіх можливих жанрів, платформ і часових періодів. Тому результати не слід автоматично переносити на інші соціальні мережі або на тексти з іншою тематикою без додаткової перевірки.

Друге обмеження пов’язане з класом mixed. Через малу кількість прикладів навіть помірна зміна кількості правильних прогнозів може помітно

змінити F1 mixed. Саме тому в роботі mixed аналізується не лише чисельно, а й через case studies, LIME/SHAP і помилки моделей.

Третє обмеження стосується API-моделей. Вони мають сильні результати, але їхня поведінка залежить від зовнішнього сервісу, версії моделі, політик безпеки, формату відповіді та вартості інференсу. Локальні моделі краще контролюються, але потребують обчислювальних ресурсів і ретельної адаптації.

Висновки до розділу 4

У цьому розділі було експериментально підтверджено доцільність використання Macro F1 як основної метрики, оскільки саме вона найкраще відображає якість на незбалансованій чотирикласовій задачі з міноритарним класом mixed. Weighted loss, рання зупинка та контроль validation Macro F1 були важливими для стабільного навчання encoder-моделей, але самі по собі не усунули складність амбівалентних повідомлень.

Фінальні результати показали, що найкращим encoder baseline є Modern-LiBERTa з Macro F1 = 0,6742 і Accuracy = 0,7482. Найкращий одиничний результат серед усіх підходів на оригінальному COSMUS зберігає Gemini 3.0 Flash ukr_few_shot із Macro F1 = 0,6768, однак різниця між ним і Modern-LiBERTa є мінімальною. Це має практичне значення: спеціалізований локальний encoder майже досягає рівня великої комерційної LLM, залишаючись значно контрольованішим для розгортання й повторюваного інференсу.

QLoRA зробила MamayLM конкурентною з encoder-підходами: MamayLM 12B QLoRA отримала Macro F1 = 0,6654, MamayLM 4B QLoRA = 0,6555. CSSent показав залежність ефекту від backbone: для XLM-RoBERTa Large конфігурації CSSent покращили Macro F1 до 0,6684-0,6691, тоді як для

Modern-LiBERTa baseline залишився сильнішим за повний CSSent. Розширення COSMUS+ підвищило якість моделей на новому тесті: Modern-LiBERTa, навчена на COSMUS+, досягла Macro F1 = 0,6981, а XLM-RoBERTa Large = 0,6808.

Аналіз калібрування, LIME/SHAP та розбір показових випадків показав, що висока класифікаційна якість не вичерпує оцінювання sentiment-моделі. Модель може мати сильний Macro F1, але бути гірше відкаліброваною, або демонструвати системні помилки на непрямій негативності, сарказмі та коротких реакціях без контексту.

ВИСНОВКИ

Магістерська дисертація містить систематичне дослідження методів мультикласового сентимент-аналізу для українсько-російського код-змішаного контенту з соціальних мереж. Запропоновано комплексний підхід CSSent для адаптації encoder-моделей, побудовано розширення COSMUS+ для перевірки стійкості моделей на ширшому розподілі Telegram-повідомлень, а також порівняно комерційні API LLM, локальні encoder-моделі та MamayLM у режимах промптингу й QLoRA-дотренування.

У першому розділі узагальнено теоретичні засади сентимент-аналізу, мультикласової класифікації та код-змішування у задачах NLP. Показано, що українсько-російський суржик створює не лише технічні труднощі для токенізації, а й прагматичні труднощі для інтерпретації тональності. Особливу увагу приділено класу mixed, який потребує розпізнавання одночасної присутності позитивних і негативних сигналів в одному повідомленні.

У другому розділі сформовано методологічну основу дослідження: описано корпуси COSMUS і COSMUS+, вибір метрик оцінювання, додаткові критерії аналізу моделей та обчислювальну інфраструктуру. Основною метрикою обрано Macro F1, оскільки COSMUS має виражений дисбаланс класів: mixed становить лише 608 із 12 224 текстів, а в тестовій вибірці лише 60 із 1 223 прикладів.

У третьому розділі визначено експериментальний дизайн: правила розбиття COSMUS і COSMUS+, препроцесинг, конфігурації API-моделей, encoder-моделей і MamayLM, принципи відтворюваності та prompt-протокол для LLM, а також описано запропонований комплексний підхід CSSent,

У четвертому розділі подано розвідувальний аналіз даних, результати порівняння моделей, аналіз класу mixed, оцінювання калібрування, LIME/SHAP-пояснюваність і розбір показових випадків.

У межах роботи побудовано COSMUS+, який містить 33 267 текстів: 12 222 рядки з початкового COSMUS і 21 045 нових очищених Telegram-прикладів. У sentiment-розподілі COSMUS+ наявно 5 926 positive, 16 981 negative, 8 813 neutral і 1 547 mixed прикладів. Ручна перевірка на 100 прикладах показала sentiment agreement accuracy 0,7400 і Cohen's kappa 0,5775, language agreement accuracy 0,9200 і kappa 0,8617. Це підтверджує корисність корпусу для експериментів, але також вказує на шумність автоматично розмічених соціальних даних.

Експериментальне порівняння показало, що Gemini 3.0 Flash у конфігурації ukr_few_shot має найвищий результат на оригінальному COSMUS test: Macro F1 = 0,6768 і Accuracy = 0,7441. Водночас Modern-LiBERTa baseline досягає дуже близького результату: Macro F1 = 0,6742 і Accuracy = 0,7482. Це є одним із ключових практичних висновків: спеціалізований локальний encoder майже досягає рівня великої комерційної LLM, залишаючись контрольованішим для повторюваного інференсу, локального розгортання та роботи з чутливими даними.

Запропонований комплексний підхід CSSent поєднує мовно-зумовлену класифікаційну голову, зважену фокусну функцію втрат, допоміжну задачу визначення мови, аугментацію код-змішаних прикладів і дистиляцію знань від Gemini через м'які мітки. Абляційний аналіз показав, що CSSent не є універсальним покращенням для кожної базової моделі. Для XLM-RoBERTa Large CSSent підвищив Macro F1 з 0,6500 до 0,6684-0,6691, тоді як Modern-LiBERTa лишилася сильнішою як базова модель. Отже, ефективність запропонованого комбінованого підходу залежить від базової encoder-основи й має перевірятися експериментально.

MamayLM виявилася слабкою в режимі prompting для 4B, але суттєво покращилася після QLoRA. MamayLM 12B prompting досягла Macro F1 = 0,5870, MamayLM 4B QLoRA = 0,6555, а MamayLM 12B QLoRA = 0,6654. Це показує, що параметрично-ефективне дотренування може перетворити

decoder-only LLM на конкурентний локальний класифікатор, хоча клас mixed залишається проблемним навіть після адаптації.

Найскладнішим класом в усіх експериментах залишається mixed. Найкращі F1 для цього класу зосереджені приблизно в діапазоні 0,42-0,44 для Gemini, Modern-LiBERTa та найкращої XLM-RoBERTa CSSent-конфігурації. Додатковий аналіз COSMUS+ показав, що mixed у середньому довший за інші класи, а довгі повідомлення з кількома змістовими центрами частіше створюють труднощі для моделей. Це підтверджує, що mixed є складним не лише через малу представленість, а й через саму семантичну природу амбівалентних оцінок.

Оцінювання калібрування показало, що вища Macro F1 не гарантує кращої узгодженості впевненості моделі з фактичною правильністю. Modern-LiBERTa baseline має ECE@10 = 0,1803, XLM-RoBERTa Large baseline = 0,1744, тоді як CSSent-конфігурації XLM-RoBERTa Large мають ECE@10 близько 0,228-0,229. Отже, для практичних систем, де важливо ранжувати повідомлення за впевненістю, калібрування слід аналізувати окремо.

Аналіз LIME/SHAP і розбір показових випадків доповнив агреговані метрики. Моделі часто помиляються на непрямій негативності, коротких реакціях без контексту, сарказмі та мовній грі. Приклади на кшталт "В Украине цена не уменьшается если нефть падает, она просто не увеличивается", "Так и есть" і "Русский диалект укрмовы сойдёт?" демонструють, що нейтральні прогнози часто виникають не через технічний fallback, а через слабко виражений або прагматично непрямий сентиментний сигнал.

Розширення COSMUS+ підвищило якість моделей на новому test set. Modern-LiBERTa, навчена лише на COSMUS, отримала на COSMUS+ test Macro F1 = 0,6419, тоді як після навчання на COSMUS+ досягла 0,6981. Для XLM-RoBERTa Large приріст становить від 0,6263 до 0,6808. Це підтверджує доцільність розширення корпусу очищеними Telegram-даними, особливо для підвищення якості на новішому розподілі повідомлень.

Наукова цінність роботи полягає в систематичному порівнянні encoder, decoder- та API-підходів на спільному код-змішаному benchmark, а також у перевірці CSSent як конфігурованого пайплайну для задачі українсько-російського code-mixed sentiment analysis.

Подальші дослідження доцільно спрямувати не на повторення вже виконаних етапів, а на глибше моделювання класу mixed, перевірку стабільності CSSent на інших code-mixed корпусах, дослідження практичних компромісів між якістю, вартістю інференсу і вимогами до апаратного забезпечення. Також як окремий перспективний напрям можна розглядати ансамблювання encoder- та decoder-моделей.

Основні результати роботи планується апробувати на міжнародній науково-практичній конференції та представити у періодичному виданні «Штучний інтелект».

ПЕРЕЛІК ПОСИЛАНЬ

1. Shynkarov, Y., Solopova, V., & Schmitt, V. (2025). Improving sentiment analysis for Ukrainian social media code-switching data. In *Proceedings of the Fourth Ukrainian Natural Language Processing Workshop (UNLP 2025)*. https://doi.org/10.18653/v1/2025.unlp-1.18
2. INSAIT Institute. (2025). *MamayLM v1.0: Technical report*. Hugging Face. https://huggingface.co/INSAIT-Institute/MamayLM-Gemma-3-4B-IT-v1.0
3. Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/2939672.2939778
4. Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*. https://papers.neurips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions
5. Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. https://arxiv.org/abs/1708.02002
6. Conneau, A., et al. (2020). Unsupervised cross-lingual representation learning at scale. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. https://arxiv.org/abs/1911.02116
7. Haltiuk, M., & Smywiński-Pohl, A. (2025). On the path to make Ukrainian a high-resource language. In *Proceedings of the Fourth Ukrainian Natural Language Processing Workshop (UNLP 2025)*. https://doi.org/10.18653/v1/2025.unlp-1.14
8. Devlin, J., et al. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference*

of the North American Chapter of the Association for Computational Linguistics (NAACL). https://arxiv.org/abs/1810.04805

9. Radchenko, V. (2021). *youscan/ukr-roberta-base*. Hugging Face. https://huggingface.co/youscan/ukr-roberta-base
10. Mohammad, S. M., & Turney, P. D. (2013). Crowdsourcing a word-emotion association lexicon. *Computational Intelligence*. https://doi.org/10.48550/arXiv.1308.6297
11. Romanyshyn, M. (2013). Rule-based sentiment analysis of Ukrainian reviews. *International Journal of Artificial Intelligence & Applications*. https://doi.org/10.5121/ijaia.2013.4410
12. Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*. https://doi.org/10.1561/1500000011
13. Bobichev, V., Kanishcheva, O., & Cherednichenko, O. (2017). Sentiment analysis in the Ukrainian and Russian news. In *IEEE UKRCON*. https://doi.org/10.1109/UKRCON.2017.8100410
14. Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. https://arxiv.org/abs/1408.5882
15. Socher, R., et al. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. https://aclanthology.org/D13-1170/
16. Aryal, S. K., Prioleau, H., & Washington, G. (2022). Sentiment classification of code-switched text using pre-trained multilingual embeddings and segmentation. *arXiv*. https://arxiv.org/abs/2210.16461
17. Solorio, T., & Liu, Y. (2008). Learning to predict code-switching points. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. https://aclanthology.org/D08-1102/

18. Chen, L., Shang, S., & Wang, Y. (2025). Bridging resource gaps in cross-lingual sentiment analysis: Adaptive self-alignment with data augmentation and transfer learning. *PeerJ Computer Science*. https://doi.org/10.7717/peerj-cs.2851
19. Gladys, A. A., & Vetriselvi, V. (2024). Sentiment analysis on a low-resource language dataset using multimodal representation learning and cross-lingual transfer learning. *Applied Soft Computing*. https://doi.org/10.1016/j.asoc.2024.111553
20. Koto, F., et al. (2024). Zero-shot sentiment analysis in low-resource languages using a multilingual sentiment lexicon. *arXiv*. https://arxiv.org/abs/2402.02113
21. Prytula, M. (2024). Fine-tuning BERT, DistilBERT, XLM-RoBERTa and Ukr-RoBERTa models for sentiment analysis of Ukrainian language reviews. *Journal of Artificial Intelligence*. https://jai.in.ua/index.php/en/issues?paper_num=1623
22. Lashyn, Y., et al. (2025). Sentiment analysis of texts using recurrent neural networks of the transformer architecture. *Advanced Information Systems*. https://doi.org/10.20998/2522-9052.2025.3.11
23. Poplack, S. (1980). Sometimes I'll start a sentence in Spanish y termino en español: Toward a typology of code-switching. *Linguistics*. https://doi.org/10.1515/ling.1980.18.7-8.581
24. Sitaram, S., et al. (2019). A survey of code-switched speech and language processing. *arXiv*. https://arxiv.org/abs/1904.00784
25. Winata, G. I., et al. (2021). Are multilingual models effective in code-switching? In *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching (CALCS)*. https://doi.org/10.18653/v1/2021.calcs-1.20
26. Ranjan, S., Mekala, D., & Shang, J. (2022). Progressive sentiment analysis for code-switched text data. In *Findings of the Association for*

Computational Linguistics: EMNLP 2022. https://aclanthology.org/2022.findings-emnlp.82/

27. Hamed, I., Elmahdy, M., & Abdennadher, S. (2017). Building a first language model for code-switch Arabic-English. *Procedia Computer Science*. https://doi.org/10.1016/j.procs.2017.10.111
28. Khanuja, S., et al. (2020). GLUECoS: An evaluation benchmark for code-switched NLP. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. https://doi.org/10.18653/v1/2020.acl-main.329
29. Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. https://arxiv.org/abs/1801.06146
30. Brown, T. B., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*. https://arxiv.org/abs/2005.14165
31. Zhang, W., et al. (2024). Sentiment analysis in the era of large language models: A reality check. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. https://arxiv.org/abs/2305.15005
32. Hu, E. J., et al. (2022). LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/2106.09685
33. Dettmers, T., et al. (2023). QLoRA: Efficient finetuning of quantized LLMs. *Advances in Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2305.14314
34. Vaswani, A., et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*. https://arxiv.org/abs/1706.03762
35. Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In *Proceedings of the 34th International*

Conference on Machine Learning. https://proceedings.mlr.press/v70/guo17a.html

36. Caruana, R. (1997). Multitask learning. *Machine Learning*. https://doi.org/10.1023/A:1007379606734
37. Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv*. https://arxiv.org/abs/1503.02531
38. Kashperova, N., & Shapoval, N. (2025). Zero-shot and few-shot named entity recognition for the Ukrainian language based on a modified Gliner architecture. *Artificial Intelligence*, *30*(4), 69–77. https://doi.org/10.15407/jai2025.04.069