Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Метод мультимодального прогнозування ефективності трансляції мРНК за повною послідовністю на основі глибинного навчання
(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-22-1
Данило КОСОВЦОВ
(власне ім'я, прізвище)
Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект
(повна назва освітньої програми)
Керівник проф. Олександр ШЕВЧЕНКО
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Лариса ЧАЛА
(підпис) (власне ім'я, прізвище)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Косовцову Данилу Івановичу
(прізвище, ім'я, по батькові)

1. Тема роботи Метод мультимодального прогнозування ефективності трансляції мРНК за повною послідовністю на основі глибинного навчання

затверджена наказом університету від 5 червня 2026 р. № 488Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 8 липня 20 26 р.

3. Вихідні дані до роботи відкритий набір даних з послідовностями мРНК та кількісними показниками ефективності трансляції; повні нуклеотидні послідовності транскриптів; інформація про функціональні ділянки 5′UTR, CDS і 3′UTR; структурно-статистичні характеристики мРНК; засоби мови Python для обробки даних, формування ознак, навчання та оцінювання моделей машинного і глибинного навчання; метрики MAE, RMSE, Pearson і Spearman для оцінювання якості прогнозування.

4. Перелік питань, що потрібно опрацювати в роботі

1) Аналіз теоретичних та методологічних основ предметної галузі

2) Розроблення методу мультимодального прогнозування ефективності трансляції мРНК

3) Програмна реалізація методу мультимодального прогнозування ефективності трансляції мРНК

4) Експериментальне дослідження та аналіз результатів

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / термін виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	08.06.2026	виконано
2	Аналіз предметної галузі та сучасних підходів	09.06.2026 – 10.06.2026	виконано
3	Формування постановки задачі та обґрунтування	11.06.2026 – 12.06.2026	виконано
	мультимодального підходу		
4	Завантаження та попередня обробка даних	13.06.2026 – 14.06.2026	виконано
5	Формування послідовнісних, структурних і	15.06.2026 – 16.06.2026	виконано
	статистичних ознак мРНК		
6	Програмна реалізація моделей машинного та	17.06.2026 – 18.06.2026	виконано
	глибинного навчання		
7	Навчання моделей, підбір гіперпараметрів та	19.06.2026	виконано
	експериментальне оцінювання результатів		
8	Оформлення пояснювальної записки та	20.06.2026	виконано
9	Перевірка на академічний плагіат	22.06.2026	виконано
10	Нормоконтроль	30.06.2026	виконано
11	Підготовка презентації та доповіді	01.07.2026	виконано
12	Попередній захист	03.07.2026	виконано
13	Рецензування	04.07.2026	виконано
14	Захист перед ЕК	08.07.2026	виконано

Дата видачі завдання 8 червня 2026 р.

Здобувач _____
(підпис)

Керівник роботи _____ проф. Олександр ШЕВЧЕНКО
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 171 с., 20 рис., 22 табл., 2 дод., 28 джерел.

ГЛИБИННЕ НАВЧАННЯ, ЕФЕКТИВНІСТЬ ТРАНСЛЯЦІЇ, ЗГОРТКОВА НЕЙРОННА МЕРЕЖА, МАШИННЕ НАВЧАННЯ, МІНІМАЛЬНА ВІЛЬНА ЕНЕРГІЯ, мРНК, МУЛЬТИМОДАЛЬНЕ ПРОГНОЗУВАННЯ, ПОВНА ПОСЛІДОВНІСТЬ, РЕГРЕСІЯ, СТРУКТУРНО-СТАТИСТИЧНІ ОЗНАКИ, 3′UTR, 5′UTR, CDS.

Об’єкт дослідження – процес прогнозування ефективності трансляції мРНК на основі послідовнісних, структурних і статистичних характеристик транскрипту.

Предмет дослідження – методи мультимодального подання повної послідовності мРНК, її функціональних сфер і структурно-статистичних ознак для регресійного прогнозування ефективності трансляції.

Мета роботи – розробити, програмно реалізувати та експериментально перевірити метод мультимодального прогнозування ефективності трансляції мРНК за повною послідовністю на основі глибинного навчання.

Методи дослідження – аналіз наукових джерел, біоінформатичний аналіз послідовностей, формування структурно-статистичних ознак, методи машинного та глибинного навчання, регресійне моделювання, підбір гіперпараметрів, оцінювання якості за метриками MAE, RMSE, Pearson і Spearman.

У роботі розроблено відтворюваний Python-проєкт для прогнозування ефективності трансляції мРНК. Запропонований метод поєднує повну нуклеотидну послідовність, позиційну інформацію про 5′UTR, CDS і 3′UTR та 83 структурно-статистичні ознаки.

ABSTRACT

Bachelor’s thesis contains: 171 pp., 20 fig., 22 tabl., 2 ann., 28 references.

CDS, CONVOLUTIONAL NEURAL NETWORK, DEEP LEARNING, FULL SEQUENCE, MACHINE LEARNING, MINIMUM FREE ENERGY, mRNA, MULTIMODAL PREDICTION, REGRESSION, STRUCTURAL AND STATISTICAL FEATURES, TRANSLATION EFFICIENCY, 3′UTR, 5′UTR.

The object of the study is the process of predicting mRNA translation efficiency based on sequence-based, structural, and statistical characteristics of a transcript.

The subject of the study is methods for multimodal representation of the full mRNA sequence, its functional regions, and structural-statistical features for regression-based prediction of translation efficiency.

The aim of the work is to develop, implement, and experimentally evaluate a multimodal method for predicting mRNA translation efficiency from the full sequence using deep learning.

Research methods include analysis of scientific sources, bioinformatic sequence analysis, structural-statistical feature engineering, machine learning and deep learning methods, regression modelling, hyperparameter tuning, and evaluation using MAE, RMSE, Pearson, and Spearman metrics.

The work develops a reproducible Python project for predicting mRNA translation efficiency. The proposed method combines the full nucleotide sequence, positional information about 5′UTR, CDS, and 3′UTR, and 83 structural-statistical features.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 8
Вступ 9
1 Аналіз теоретичних та методологічних основ предметної галузі 12
1.1 Біологічні основи трансляції мРНК 12
1.2 Структурна організація повної послідовності мРНК 15
1.3 Показники трансляційної активності мрнк та методи їх отримання . 18
1.4 Фактори, що впливають на ефективність трансляції мРНК 23
1.5 Огляд сучасних підходів до прогнозування ефективності трансляції мРНК 28
1.6 Проблеми та обмеження існуючих підходів 33
2 Розроблення методу мультимодального прогнозування ефективності трансляції мРНК 37
2.1 Формальна постановка задачі прогнозування ефективності трансляції мРНК 37
2.2 Обґрунтування вибору мультимодального підходу 41
2.3 Формування вхідного представлення мРНК 45
2.4 Формування структурних та статистичних ознак мРНК 50
2.5 Архітектура мультимодальної моделі глибинного навчання 55
2.6 Методика навчання та оцінювання якості моделі 63
3 Програмна реалізація методу мультимодального прогнозування ефективності трансляції мРНК 68
3.1 Загальна логіка програмної реалізації 68
3.2 Програмне середовище, бібліотеки та конфігурація проєкту 72
3.3 Структура файлів програмного проєкту 76
3.4 Реалізація завантаження та аудиту вихідного набору даних 81
3.5 Реалізація попередньої обробки даних 86
3.6 Реалізація формування структурно-статистичних ознак 91
3.7 Реалізація підготовки даних для pytorch 96

3.8 Реалізація архітектур моделей глибинного навчання 100
3.9 Реалізація навчання, підбору гіперпараметрів та збереження моделей 105
3.10 Реалізація оцінювання та збереження артефактів 110
4 Експериментальне дослідження та аналіз результатів..... 116
4.1 Характеристика вихідного набору даних 116
4.2 Результати попередньої обробки даних..... 122
4.3 Аналіз сформованих структурно-статистичних ознак..... 126
4.4 Результати базових моделей машинного навчання..... 131
4.5 Результати навчання моделей глибинного навчання 134
4.6 Підбір гіперпараметрів мультимодальної моделі..... 139
4.7 Фінальне тестове оцінювання мультимодальної моделі 144
4.8 Порівняння моделей та аналіз похибок 148
4.9 Обмеження проведеного експерименту..... 155
4.10 Узагальнення результатів експериментального дослідження..... 159
Висновки 163
Перелік джерел посилання..... 165
Додаток А Ілюстрації роботи демонстраційного інтерфейсу 169
Додаток Б Відомість кваліфікаційної роботи..... 171

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

мРНК – матрична рибонуклеїнова кислота;

РНК – рибонуклеїнова кислота;

3′UTR – 3′ untranslated region – 3′-нетрансльована ділянка мРНК;

5′UTR – 5′ untranslated region – 5′-нетрансльована ділянка мРНК;

AI – Artificial Intelligence – штучний інтелект;

AUG – старт-кодон, що зазвичай відповідає метіоніну та визначає початок трансляції;

CDS – Coding Sequence – кодуюча послідовність мРНК;

CNN – Convolutional Neural Network – згорткова нейронна мережа;

GC-вміст – частка нуклеотидів гуаніну та цитозину в послідовності;

MAE – Mean Absolute Error – середня абсолютна похибка;

MFE – Minimum Free Energy – мінімальна вільна енергія;

ML – Machine Learning – машинне навчання;

ORF – Open Reading Frame – відкрита рамка зчитування;

Pearson – коефіцієнт кореляції Пірсона;

Ribo-seq – Ribosome Profiling Sequencing – секвенування рибосомних профілів;

RMSE – Root Mean Squared Error – корінь із середньоквадратичної похибки;

RNA-seq – RNA sequencing – секвенування РНК;

Spearman – коефіцієнт рангової кореляції Спірмена;

TE – Translation Efficiency – ефективність трансляції;

uAUG – upstream AUG – старт-кодон AUG, розташований у 5′UTR перед основним старт-кодоном;

uORF – upstream Open Reading Frame – відкрита рамка зчитування, розташована перед основною кодуючою послідовністю.

ВСТУП

Ефективність трансляції мРНК є однією з ключових характеристик, що визначає фактичний рівень синтезу білка за наявності певного транскрипту [1]. Для прикладних задач біоінформатики, синтетичної біології та розроблення терапевтичних мРНК важливим є не лише аналіз механізмів трансляційного контролю, а й прогнозування рівня трансляції на основі властивостей самої молекули. Сучасні високопродуктивні експериментальні методи, зокрема polysome profiling та ribosome profiling, дали змогу перейти від локального аналізу окремих транскриптів до побудови кількісних моделей, здатних пов’язувати послідовність мРНК з її трансляційною активністю [2]. Translation efficiency зазвичай оцінюють шляхом поєднання даних Ribo-seq та RNA-seq, а mean ribosome loading використовується для оцінювання інтенсивності трансляції в експериментах полісомного профілювання.

Актуальність роботи зумовлена тим, що ефективність трансляції залежить від багатьох факторів, серед яких важливе місце займають 5′UTR, кодуюча послідовність, вторинна структура РНК, локальні структурні особливості поблизу старт-кодону, кодонний склад ранньої частини CDS, довжина окремих ділянок транскрипту та клітинний контекст. Через складність взаємодії цих факторів прогнозування ефективності трансляції залишається нетривіальною задачею. Крім того, сучасні моделі глибокого навчання, навчені на великих експериментальних наборах даних, можуть демонструвати зниження точності під час застосування до інших типів даних, зокрема до ендогенних мРНК [3].

Існуючі методи трансляційного прогнозування основуються на аналізі часткових ділянок транскрипту або спеціально обраних ознак. Зокрема, деякі методи фокусуються лише на 5′UTR, а інші зафіксовані послідовнісні або структурні особливості. Однак молекула мРНК складається з різних ділянок, які можуть мати зворотне вплив на процесс трансляції різним

способом. Це обґрунтовує надобність створення методу, який бере до уваги всі аспекти молекулярних речовин.

Метою роботи є розроблення та дослідження мультимодального підходу до прогнозування ефективності трансляції мРНК на основі повної послідовності транскрипту, структурно-статистичних ознак та методів машинного навчання.

Об’єктом дослідження є процес прогнозування ефективності трансляції мРНК на основі властивостей транскрипту.

Предметом дослідження є моделі та методи машинного навчання для прогнозування ефективності трансляції мРНК з використанням послідовнісних, структурних і числових ознак.

Для досягнення поставленої мети необхідно вирішити такі задачі:

– проаналізувати сучасні підходи до прогнозування ефективності трансляції мРНК;

– дослідити структуру вихідного набору даних та характеристики повних послідовностей мРНК;

– виконати попередню обробку даних і підготувати їх до подальшого моделювання;

– сформувати числові, структурні та послідовнісні ознаки для опису транскриптів;

– розробити моделі машинного навчання для прогнозування ефективності трансляції;

– провести навчання та оцінювання моделей за регресійними метриками;

– порівняти результати моделей, що використовують лише послідовність, із мультимодальним підходом;

– визначити переваги, обмеження та можливі напрями подальшого розвитку запропонованого підходу.

В даній роботі застосовано методи аналізу біологічних послідовностей, передпроцессування даних, формування ознак,

регресійного моделювання, машинного та глибокого навчання. Як показники якості прогнозування використовуються регресійні метрики, які дають можливість порівнювати точність різних моделей та встановлювати можливості мультимодального представництва даних.

Практичний значущий даний метод в тому, що його можна буде використовувати для задач біоінформатики, виявлення терапевтичних мРНК, оптимізації синтетичних генетичних конструкцій та вивчення механизмів трансляційного контролю. Використання моделей прогнозування буде допомагати у випередній оцінці трансляційної активності транскрипту та у суттєвому скороченні кількості експериментальної роботи на ранніх етапах дослідження.

1 АНАЛІЗ ТЕОРЕТИЧНИХ ТА МЕТОДОЛОГІЧНИХ ОСНОВ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Біологічні основи трансляції мРНК

Одним із центральних процесів реалізації генетичної інформації в клітині є синтез білка за матрицею мРНК. У загальному вигляді передавання генетичної інформації можна подати як послідовність процесів: ДНК використовується для синтезу РНК, а мРНК, у свою чергу, використовується рибосомою як матриця для синтезу поліпептидного ланцюга. Саме цей етап, на якому нуклеотидна послідовність мРНК перетворюється на амінокислотну послідовність білка, називається трансляцією [4]. Загальну схему процесу трансляції мРНК наведено на рисунку 1.1.

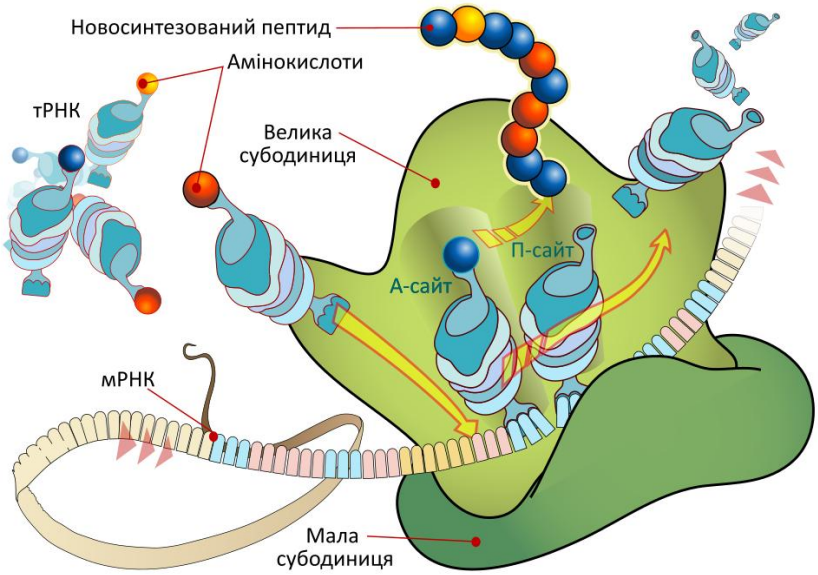


Рисунок 1.1 – Загальна схема трансляції мРНК

Як видно з рисунка 1.1, під час трансляції мРНК використовується рибосомою як матриця для синтезу поліпептидного ланцюга. Рибосома послідовно зчитує кодони мРНК, а транспортні РНК доставляють відповідні

амінокислоти. У результаті між амінокислотами формуються пептидні зв’язки, що приводить до утворення білкового продукту.

Такий процес є основою для подальшого розуміння задачі прогнозування ефективності трансляції. Якщо певна мРНК активніше залучається рибосомами до синтезу білка, то її translation efficiency буде вищою. Отже, модель прогнозування повинна враховувати ті властивості мРНК, які можуть впливати на залучення рибосоми, вибір старт-кодону та подальший перебіг елонгації.

Молекула мРНК містить інформацію у вигляді послідовності нуклеотидів. Воно зчитується рибосомою не нуклеотидами в окремості, а кодонами – триплетами нуклеотідів, що отримали цей прізвище через те, що кожен з них відповідає деякій амінокислоті або сигналу завершення. Наприклад, кодон AUG зазвичай є старт-кодоном, кодуючи старт кодуючої ділянка, а стоп-кодони UAA, UAG та UGA – кодують стоп. Однозначно розпізнавання стартового кодона і зчитування кодонів є необхідними умовами правильного синтезу білкового продукту.

Трансляція є процесом, що включає в себе зазначені вище рибосому, транспортну РНК, амінокислоту, трансляційні фактори та енергетичні молекули. Способом дії рибосоми є зчитування мРНК з 5′-кінця до 3′-кінця, координація засипки відповідних транспортних РНК і каталитична формулювання пептидних зв'язків між амінокислотами. В результаті формується поліпептидний ланцюг, який може перетворитися у білок.

У еукаріотичних клітинах трансляцію зазвичай поділяють на кілька основних етапів: ініціацію, елонгацію, термінацію та рециклінг рибосоми [5]. На етапі ініціації відбувається розпізнавання мРНК рибосомним комплексом, сканування початкової ділянки мРНК та вибір старт-кодону. Саме цей етап є одним із найбільш регульованих, оскільки від ефективності залучення рибосоми та правильного вибору стартової позиції значною мірою залежить загальна інтенсивність синтезу білка.

На елонгаційному етапі рибосома просуватиметься з 5′-кінця до 3′-кінця кодуючої ділянки та засипиться амінокислотами за кодонами. На цьому етапі важливими є самі кодони, локальна друга структура мРНК, швидкість розпізнавання кодонів транспортними РНК а також можливі ділянки зниження темпу розпізнавання транспортною РНК. Тобто кодуюча послідовність не тільки задає структуру білка, але і його ефективність синтезу.

Процес трансляції зупиняється на стоп-кодоні. На цій стадії в рибосомальний комплекс залучаються спеціалізовані фактори термінації, що сприяє вивільненню синтезованого поліпептидного ланцюга. Після завершення цього процесу компоненти комплексу трансляції можуть бути розділені та використані в наступних циклах синтезу білка.

Ефективність трансляції окремої мРНК не є постійною величиною, яка визначається виключно наявністю певного транскрипту в клітині. Два транскрипти з подібним або майже подібним рівнем експресії можуть забезпечувати різні рівні синтезу білка. Це пов’язано з тим, що на процес трансляції впливають структурні характеристики мРНК, особливості її некодуючих і кодуючих ділянок, контекст стартового кодону, структура кодонів, регуляторні мотиви, стабільність молекули та стан клітинного середовища. Тому кількісне прогнозування трансляційної активності вимагає врахування не лише наявності мРНК, а й її послідовності, структурних і контекстуальних особливостей.

З точки зору проблем машинного навчання, процес трансляції можна розглядати як біологічний процес, результат якого частково відображається в вимірюваних кількісних характеристиках, таких як ефективність трансляції або середнє навантаження на рибосоми. Ці параметри описують ступінь використання певної мРНК для синтезу білка рибосомами. Відповідно, прогноз ефективності трансляції на основі властивостей мРНК є актуальною проблемою біоінформатики, оскільки дозволяє перейти від

опису окремих регуляторних елементів до розробки обчислювальних моделей, здатних оцінити трансляційний потенціал.

1.2 Структурна організація повної послідовності мРНК

Для створення прогностичної моделі ефективності трансляції вкрай важливо розглядати молекулу мРНК як структуровану, що складається з функціонально різних частин, а не як однорідний ланцюг нуклеотидів. Зріла молекула мРНК зазвичай складається з 5'-нетрансльованої ділянки (5'UTR), кодуючої послідовності (CDS) і 3'-нетрансльованої ділянки (3'UTR). Таке структурування визначає не лише амінокислотний склад майбутнього білка, але й регуляторні властивості самого транскрипту, його стабільність, доступ до рибосомного комплексу та можливий рівень трансляції. Всю молекулу мРНК можна представити у вигляді трьох основних частин: 5'UTR, CDS і 3'UTR. Структурну організацію зрілої еукаріотичної мРНК наведено на рисунку 1.2.

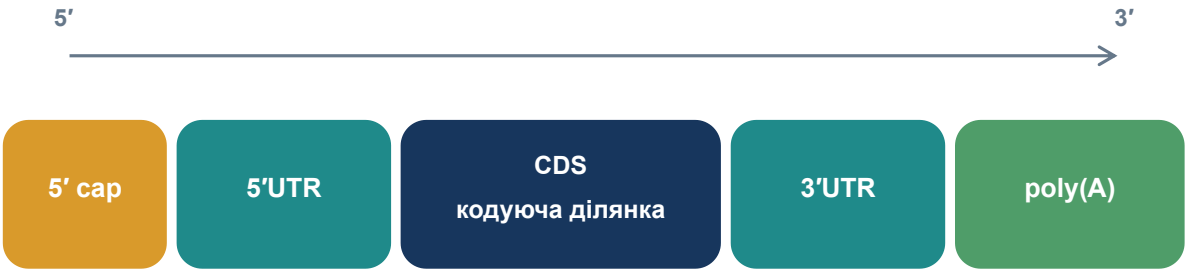


Рисунок 1.2 – Структурна організація зрілої еукаріотичної мРНК

Як видно з рисунка 1.2, зрілі молекули мРНК мають організовану структуру, де кожна частина виконує певні функції. 5'UTR розташований вище за кодуючу послідовність і бере участь у регулюванні процесу ініціації трансляції. CDS несе інформацію про амінокислотний склад білка і розпізнається безпосередньо рибосомою. 3'UTR розташований нижче за кодуючу послідовність і може впливати на стабільність, локалізацію та

регуляцію самої мРНК. Таке структурування є критичним для задачі прогнозування ефективності перекладу. Один нуклеотидний мотив може виконувати різні біологічні функції залежно від його положення в молекулі мРНК. Таким чином, у цьому дослідженні вся мРНК розглядається як набір функціонально різних частин, а не однорідний ланцюг символів.

Розташована вище за течією від кодуючої послідовності та не кодує жодних амінокислот – 5′UTR, або 5′ нетрансльована ділянка. Однак 5′UTR відіграє важливу регуляторну роль у процесі ініціації трансляції. Ініціація трансляції вимагає взаємодії рибосомного комплексу з молекулою мРНК і пошуку стартового кодону. З цієї причини довжина 5′UTR, його нуклеотидний склад, регуляторні елементи, вторинна структура, наявність AUG і uORF, що знаходяться вище, можуть значно впливати на рекрутинг рибосом і точне розпізнавання стартового кодону [6].

Особливу увагу варто звернути на стартовий кодон. Зазвичай стартовий кодон, який ініціює трансляцію в кодуючій послідовності, є AUG і служить рамкою зчитування для трансляції. Наявність додаткових кодонів AUG вище за течією від основного стартового кодону в 5′UTR може значно впливати на сканування рибосоми. Ці додаткові кодони AUG називаються вище за течією AUG, оскільки вони знаходяться вище за течією від основного сайту ініціації. Таким чином, присутність висхідного AUG може впливати на ефективність ініціації, оскільки рибосома може ідентифікувати інші сайти ініціації або витрачати зусилля на трансляцію uORF.

CDS, або кодуюча послідовність, становить основну частину мРНК, що кодує амінокислотну послідовність білка. Він починається зі стартового кодону і закінчується на одному зі стоп-кодонів. На відміну від 5′UTR і 3′UTR, CDS зчитується рибосомою у формі кодонів. Кожен кодон складається з трьох нуклеотидів і кодує певний сигнал припинення синтезу амінокислоти або білка. Іншими словами, CDS визначає послідовність амінокислот білка.

Однак роль CDS не обмежується кодуванням амінокислот. Існує кілька факторів, що впливають на швидкість і рівномірність подовження, які можна знайти в послідовності кодування. Фактори, що впливають на динаміку трансляції, можуть включати вибір кодону, локальну вторинну структуру мРНК, вміст GC, позиційні властивості на 5'-кінці CDS і можливі сайти паузи рибосом. З цієї причини при прогнозуванні ефективності трансляції необхідно окремо враховувати не тільки 5′UTR, але й послідовність кодування.

Розташована після кодуючої послідовності та стоп-кодону – 3′UTR, або 3′ нетрансльована ділянка. Він також не кодує амінокислотну послідовність білка, але виконує важливі регуляторні функції. 3′UTR може містити сайти зв’язування регуляторних білків і мікроРНК, елементів, що впливають на стабільність мРНК, її локалізацію всередині клітини та ефективність трансляції. У багатьох випадках саме 3′UTR бере участь у посттранскрипційній регуляції експресії генів, тобто регуляції, що відбувається після синтезу мРНК [7].

Важливо отримати 3′UTR при аналізі повного складу мРНК, після чого ця ділянка не є незначною частиною транскрипту. Якщо під час побудови моделі використовується лише 5′UTR або не використовується CDS, деяка інформація про поведінку мРНК може бути виключена з аналізу. тим самим забезпечується стабільність транскрипту, його можливі взаємодії з регуляторними молекулами та особливості функціонування в клітинному середовищі. Завдяки цій повній послідовності мРНК має сенс розглядати як набір взаємопов’язаних функціональних ділянок, а не просто послідовність нуклеотидів, записаних один за одним.

Окрім основних ділянок 5′UTR, CDS та 3′UTR, еукаріотична мРНК має додаткові структурні елементи, що впливають на її функціонування. До них відносяться 5'-кеп на початку молекули та полі(А)-хвіст на 3'-кінці молекули. 5′-кап бере участь у розпізнаванні мРНК трансляційним апаратом і захисті молекули від деградації. Полі(А)-хвіст впливає на стабільність

мРНК, її транспорт і ефективність трансляції. Хоча ці елементи не завжди представлені явно в термінах нуклеотидної послідовності в наборах даних, їх біологічна роль важлива для розуміння загальної структури мРНК.

Що стосується завдання машинного навчання, структурна організація мРНК має практичне значення, оскільки різні частини послідовності мають різну біологічну інтерпретацію. Той самий мотив нуклеотиду може мати різне значення залежно від частини послідовності, в якій він розташований: 5′UTR, CDS або 3′UTR. Наприклад, AUG у 5′UTR може бути AUG вище за течією та впливати на ініціацію трансляції, тоді як AUG у CDS є просто звичайним кодоном метіоніну в послідовності білка. Подібним чином, структурна стабільність фрагмента поблизу стартового кодону може мати інший вплив на ефективність трансляції, ніж структурна стабільність віддаленого фрагмента 3′UTR.

Таким чином, при розробці мультимодального методу прогнозування ефективності перекладу необхідно враховувати не тільки саму послідовність, але й позиційний контекст кожного фрагмента. Повна мРНК має бути представлена у вигляді послідовності з розділеними функціональними ділянками 5′UTR, CDS і 3′UTR. Такий підхід дозволяє поєднати аналіз регуляторних ділянок, інформації кодування та структурних характеристик транскрипту, що є необхідною умовою для подальшого розвитку мультимодальної моделі прогнозування.

1.3 Показники трансляційної активності мРНК та методи їх отримання

Для побудови моделей прогнозування ефективності трансляції необхідно визначити кількісний показник, який буде використовуватися як цільова змінна. У задачах аналізу трансляційної активності мРНК найчастіше використовують такі показники, як translation efficiency, mean ribosome loading та ribosome density. Вони пов’язані з активністю рибосом

на транскрипті, однак характеризують цей процес з різних боків і можуть бути отримані різними експериментальними методами.

Translation efficiency, або ефективність трансляції, характеризує, наскільки активно певна мРНК використовується рибосомами для синтезу білка відносно її кількості в клітині. Цей показник є важливим тому, що рівень мРНК не завжди прямо відповідає рівню білка. Один транскрипт може бути представлений у клітині у великій кількості, але транслюватися неефективно, тоді як інший транскрипт із нижчим рівнем експресії може активно залучатися до синтезу білка. Саме тому translation efficiency дозволяє оцінювати не лише наявність мРНК, а й інтенсивність її використання трансляційним апаратом.

У найпростішому вигляді translation efficiency для і-го транскрипту можна подати як відношення сигналу рибосомного профілювання до рівня мРНК:

$$\mathrm{TE_i} = \frac{\mathrm{Ribo_i}}{\mathrm{RNA_i}}, \tag{1.1}$$

де $\mathrm{TE_I}$ – ефективність трансляції і-го транскрипту;

$\mathrm{Ribo_i}$ – нормалізований сигнал ribosome profiling для і-го транскрипту;

$\mathrm{RNA_i}$ – нормалізований рівень RNA-seq для цього ж транскрипту.

У практичних задачах для стабілізації числових значень часто використовують логарифмічне подання цього співвідношення:

$$\log\mathrm{TE_i} = \log_2\left(\frac{(\mathrm{Ribo_i} + \varepsilon)}{(\mathrm{RNA_i} + \varepsilon)}\right), \tag{1.2}$$

де ε – мала додатна константа, яка запобігає діленню на нуль.

Логарифмічне перетворення зменшує вплив надто великих значень і робить розподіл цільової змінної зручнішим для подальшого статистичного аналізу та машинного навчання [8].

Основним експериментальним методом, який дозволяє отримувати інформацію про активність рибосом на мРНК, є ribosome profiling, або Ribo-seq. Цей метод ґрунтується на виділенні фрагментів мРНК, захищених рибосомами від нуклеазного розщеплення. Після секвенування таких фрагментів можна визначити, які ділянки транскриптів були зайняті рибосомами. На відміну від звичайного RNA-seq, який переважно характеризує кількість мРНК, ribosome profiling дає змогу оцінити трансляційний стан транскриптів і розподіл рибосом уздовж мРНК з високою роздільною здатністю [2].

Поєднання Ribo-seq та RNA-seq є методологічно важливим, оскільки ці два типи даних описують різні рівні експресії генетичної інформації. RNA-seq показує, скільки транскрипту наявно в клітині, а Ribo-seq показує, наскільки активно цей транскрипт взаємодіє з рибосомами. Саме співвідношення цих двох сигналів дозволяє перейти від аналізу кількості мРНК до оцінювання ефективності її трансляції. Принцип оцінювання translation efficiency на основі поєднання RNA-seq та ribosome profiling наведено на рисунку 1.3.

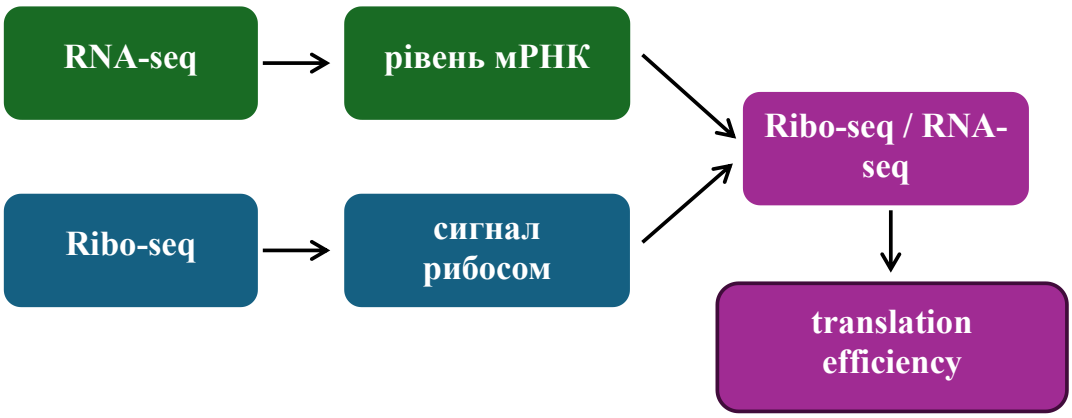


Рисунок 1.3 – Принцип оцінювання translation efficiency за даними RNA-seq і Ribo-seq

Як видно з рисунка 1.3, RNA-seq характеризує кількість транскрипту, тоді як ribosome profiling відображає активність рибосом на цьому

транскрипті. Поєднання цих двох типів даних дає змогу оцінити, наскільки активно наявна мРНК використовується для синтезу білка.

Саме тому translation efficiency є зручним цільовим показником для задачі машинного навчання. На вхід моделі можуть подаватися властивості мРНК, а на виході формується числовий прогноз її трансляційної активності. Така постановка узгоджує біологічний зміст задачі з регресійним моделюванням.

Іншим важливим показником трансляційної активності є mean ribosome loading, або середнє рибосомне навантаження. Цей показник відображає середню кількість рибосом, асоційованих з певною мРНК. Якщо з транскриптом пов’язана велика кількість рибосом, це зазвичай свідчить про активну трансляцію. Якщо ж мРНК переважно перебуває у фракціях з малою кількістю рибосом або без них, її трансляційна активність може бути нижчою.

У загальному вигляді mean ribosome loading можна подати як зважене середнє за фракціями, що відповідають різній кількості рибосом:

$$MRL_i = \sum_{\{k\}} k \cdot p_{\{i,k\}}, \tag{1.3}$$

де MRL_I – середнє рибосомне навантаження і-го транскрипту;

k – кількість рибосом, асоційованих з мРНК у відповідній фракції;

$p_{\{i,k\}}$ – частка або відносний сигнал і-го транскрипту у фракції з k рибосомами.

Mean ribosome loading часто отримують за допомогою polysome profiling. У цьому методі клітинний лізат розділяють у градієнті сахарози, унаслідок чого мРНК розподіляються за кількістю пов’язаних з ними рибосом. Нефункціонально або слабо трансльовані мРНК зазвичай потрапляють у легші фракції, тоді як активно трансльовані мРНК з кількома рибосомами потрапляють у важчі полісомні фракції [9]. Подальший аналіз

цих фракцій дозволяє оцінити, наскільки інтенсивно окремі транскрипти залучені до синтезу білка.

Ще одним пов’язаним показником є ribosome density, або щільність рибосом. На відміну від translation efficiency або mean ribosome loading, які можуть бути подані як одне інтегральне значення для транскрипту, ribosome density може описувати розподіл рибосом уздовж окремих ділянок мРНК. Такий показник є корисним для аналізу локальної трансляційної динаміки, зокрема ділянок можливого уповільнення рибосом, накопичення рибосом або нерівномірної елонгації в межах CDS.

Для задачі прогнозування ефективності трансляції важливо розуміти, що translation efficiency, mean ribosome loading та ribosome density не є повністю тотожними показниками. Translation efficiency найчастіше описує відносну ефективність використання мРНК рибосомами з урахуванням рівня транскрипту. Mean ribosome loading характеризує середню кількість рибосом, асоційованих з мРНК. Ribosome density, у свою чергу, може описувати локальний розподіл рибосом уздовж транскрипту. Усі ці показники пов’язані з трансляційною активністю, однак відрізняються способом отримання, біологічною інтерпретацією та рівнем деталізації. Урахування зазначених відмінностей дає змогу коректніше зіставляти результати, отримані за допомогою різних експериментальних підходів.

У межах даної роботи основною цільовою змінною доцільно розглядати translation efficiency, оскільки вона безпосередньо відповідає темі прогнозування ефективності трансляції мРНК. Mean ribosome loading може розглядатися як близький показник трансляційної активності, особливо в задачах, побудованих на даних polysome profiling. Такий підхід дозволяє узгодити біологічну постановку задачі з регресійною постановкою в машинному навчанні, де на вхід моделі подаються ознаки мРНК, а на виході формується числове передбачення трансляційної активності.

1.4 Фактори, що впливають на ефективність трансляції мРНК

Ефективність трансляції мРНК формується під впливом сукупності факторів, що належать до різних рівнів організації транскрипту. До них належать послідовнісні ознаки, структурні характеристики, позиційне розташування регуляторних елементів, властивості кодуючої ділянки та клітинний контекст. Саме багатофакторність процесу трансляції зумовлює необхідність використання не одного окремого показника, а комплексного набору ознак, здатних описати мРНК з різних боків. Основні групи факторів, що можуть впливати на ефективність трансляції мРНК, узагальнено на рисунку 1.4.



Рисунок 1.4 – Основні фактори, що впливають на ефективність трансляції мРНК

Як показано на рисунку 1.4, ефективність трансляції формується під впливом кількох груп факторів. Частина з них пов’язана з окремими функціональними ділянками мРНК, зокрема 5′UTR, CDS і 3′UTR. Інші фактори описують загальні властивості послідовності, такі як GC-вміст, наявність upstream AUG, k-мерний склад і локальна вторинна структура.

Така багатофакторність пояснює, чому прогнозування ефективності трансляції не доцільно зводити до аналізу лише однієї ділянки мРНК. Для повнішого опису транскрипту необхідно враховувати як повну нуклеотидну послідовність, так і позиційний контекст її функціональних ділянок та числові структурно-статистичні характеристики. Саме це обґрунтовує використання мультимодального підходу в подальших розділах роботи.

Одним із найважливіших факторів є 5′UTR, оскільки ця ділянка безпосередньо пов’язана з ініціацією трансляції. Під час канонічної еукаріотичної ініціації рибосомний комплекс взаємодіє з 5′-кінцем мРНК і сканує послідовність у напрямку до старт-кодону. Тому довжина 5′UTR, її нуклеотидний склад, наявність локальної вторинної структури, регуляторних мотивів та альтернативних стартових елементів можуть впливати на швидкість і правильність початку трансляції [6].

Важливим елементом 5′UTR є кодон AUG, розташований перед основним старт-кодоном CDS. Наявність таких елементів може змінювати поведінку рибосомного комплексу під час сканування. Якщо рибосома розпізнає upstream AUG як стартову позицію, це може призвести до трансляції короткої upstream open reading frame або до зниження ефективності трансляції основної кодуючої послідовності. Тому кількість upstream AUG та їх розташування відносно основного старт-кодону є важливими ознаками для опису регуляторного потенціалу 5′UTR.

Кількість upstream AUG у 5′UTR може бути подана у вигляді простої дискретної ознаки:

$$uAUG = \sum_{\{i=1\}}^{\{p_{\{start\}}-2\}} I\left(s_i s_{\{i+1\}} s_{\{i+2\}} = \backslash mathrm\{AUG\}\right), \quad (1.4)$$

де $uAUG$ – кількість upstream AUG;

$p_{\{start\}}$ – позиція основного старт-кодону CDS;

s_i – нуклеотид у позиції i;

$I(\cdot)$ – індикаторна функція, що дорівнює 1, якщо умова виконується, і 0 в іншому випадку; підсумовування виконується лише для позицій, розташованих перед основним старт-кодоном CDS.

Окрім самого факту наявності старт-кодону AUG, важливим є його локальний послідовнісний контекст. У еукаріотичних мРНК ефективність розпізнавання старт-кодону залежить від нуклеотидів, розташованих поруч із AUG. Такий контекст часто пов’язують із послідовністю Козак, яка описує сприятливе оточення старт-кодону для ініціації трансляції [10]. Якщо старт-кодон розташований у слабкому контексті, рибосомний комплекс може менш ефективно ініціювати трансляцію або пропускати цей старт, що впливає на рівень синтезу білка.

Активно використовується 5′UTR в експериментальних і обчислювальних дослідженнях трансляції. Зокрема, було показано, що послідовність 5′UTR може містити достатньо інформації для прогнозування ribosome loading, якщо використовується великий експериментальний набір даних і модель глибокого навчання [11]. Це підтверджує важливість 5′UTR як окремого джерела ознак, однак не означає, що прогнозування ефективності трансляції повинно обмежуватися лише цією ділянкою.

Наступною важливою частиною повної мРНК є CDS. Кодуюча послідовність визначає амінокислотний склад білка, але водночас впливає і на сам процес трансляції. Значення мають вибір синонімічних кодонів, частота окремих кодонів, локальна вторинна структура, GC-вміст, а також особливості початкової частини CDS. Оскільки рибосома рухається вздовж CDS під час елонгації, властивості цієї ділянки можуть впливати на швидкість синтезу білка та на можливі локальні затримки рибосом.

Особливу роль відіграють перші кодони CDS. Початкова частина кодуючої послідовності розташована поблизу старту трансляції, тому її структурні та послідовнісні властивості можуть впливати на перехід від ініціації до елонгації. Дослідження, у яких аналізувалася повна кодуюча послідовність із синонімічними варіантами, показують, що навіть

інформація про початкові кодони може бути суттєвою для прогнозування рівня білкової продукції [12]. Це обґрунтовує необхідність врахування CDS як самостійного компонента моделі.

Однією з базових статистичних характеристик послідовності є GC-вміст. Він показує частку гуаніну та цитозину в аналізованій ділянці мРНК:

$$GC = \frac{(N_G + N_C)}{L}, \tag{1.5}$$

де GC – частка гуаніну та цитозину;

N_G – кількість нуклеотидів G;

N_C – кількість нуклеотидів C;

L – довжина аналізованої послідовності.

GC-вміст є важливим тому, що він пов’язаний із потенційною стабільністю вторинної структури РНК. Пари G-C є енергетично стабільнішими, ніж пари A-U, тому ділянки з високим GC-вмістом можуть частіше формувати стабільні локальні структури. Водночас вплив GC-вмісту не можна розглядати ізольовано, оскільки значення має не лише загальна частка нуклеотидів, а й їх порядок у послідовності та розташування відносно функціональних елементів мРНК.

Вторинна структура РНК є ще одним важливим фактором, що може впливати на трансляцію. Молекула мРНК здатна формувати локальні спарювання між комплементарними нуклеотидами, унаслідок чого виникають шпильки, петлі та інші структурні елементи. Якщо стабільна структура формується поблизу 5′-кінця, у 5′UTR або біля старт-кодону, вона може ускладнювати сканування рибосомного комплексу та знижувати ефективність ініціації трансляції.

Для кількісного опису вторинної структури часто використовують мінімальну вільну енергію, або MFE. Цей показник характеризує енергетично найвигіднішу передбачену структуру РНК для заданої послідовності. Значення MFE зазвичай є від’ємним: чим воно нижче, тим

стабільнішою вважається відповідна структура. Одним із поширених інструментів для прогнозування вторинної структури РНК є RNAfold із пакета ViennaRNA, який обчислює MFE та оптимальну вторинну структуру для заданої послідовності [13].

Оскільки значення MFE залежить від довжини аналізованого фрагмента, для порівняння різних ділянок доцільно використовувати нормалізоване значення:

$$MFE_{norm} = \frac{MFE}{L_w}, \tag{1.6}$$

де MFE_{norm} – нормалізована мінімальна вільна енергія;

MFE – мінімальна вільна енергія фрагмента;

L_w – довжина вікна або аналізованої ділянки.

У задачі прогнозування ефективності трансляції доцільно розраховувати MFE не лише для всієї мРНК, а й для окремих локальних вікон. Особливо важливими можуть бути 5′UTR, ділянка навколо старт-кодону, початкова частина CDS та інші регіони, де вторинна структура може безпосередньо впливати на доступність мРНК для рибосоми. Такий підхід дозволяє врахувати не лише загальну структурну стабільність транскрипту, а й локальні структурні ефекти.

Окрему роль у регуляції трансляції відіграє 3′UTR. Ця ділянка не кодує білок, однак може містити регуляторні мотиви, ділянки зв’язування мікроРНК та білків, елементи, що впливають на стабільність, локалізацію та час життя мРНК. Через це 3′UTR може опосередковано впливати на кількість доступного транскрипту та на умови його трансляції [7]. У контексті прогнозування за повною послідовністю мРНК ця ділянка повинна розглядатися як частина загального вхідного представлення.

Клітинний контекст також може істотно змінювати ефективність трансляції. Одна і та сама мРНК може мати різну трансляційну активність у різних типах клітин, тканинах або експериментальних умовах. Це пов’язано

з відмінностями у концентрації факторів трансляції, активності регуляторних білків, наявності мікроРНК, метаболічному стані клітини та інших особливостях біологічного середовища. Тому клітинний контекст є важливою потенційною модальністю, якщо відповідна інформація присутня в наборі даних.

У задачах машинного навчання перелічені фактори можуть бути представлені у вигляді різних груп ознак. Послідовнісні ознаки описують порядок нуклеотидів у 5′UTR, CDS і 3′UTR. Позиційні ознаки визначають, до якої функціональної ділянки належить кожен фрагмент послідовності. Структурні ознаки характеризують вторинну структуру РНК і MFE. Статистичні ознаки включають довжини ділянок, GC-вміст, кількість upstream AUG, частоти k-мерів та інші числові характеристики. Контекстні ознаки, за наявності, описують тип клітини або умови експерименту.

Таким чином, ефективність трансляції мРНК не може бути повністю пояснена лише однією ділянкою або одним типом ознак. 5′UTR впливає переважно на ініціацію трансляції, CDS – на кодування білка та перебіг елонгації, 3′UTR – на посттранскрипційну регуляцію, вторинна структура – на доступність ділянок мРНК для рибосомного комплексу, а клітинний контекст – на умови функціонування транскрипту. Це обґрунтовує доцільність мультимодального підходу, у якому різні групи ознак розглядаються як взаємодоповнювальні джерела інформації для прогнозування translation efficiency або mean ribosome loading.

1.5 Огляд сучасних підходів до прогнозування ефективності трансляції мРНК

Сучасні підходи до прогнозування ефективності трансляції мРНК можна умовно поділити на декілька напрямів. Частина досліджень зосереджується на 5′UTR, оскільки ця ділянка безпосередньо пов’язана з ініціацією трансляції та містить регуляторні елементи, що впливають на

залучення рибосомного комплексу. Інші підходи враховують кодуючу послідовність, локальну вторинну структуру мРНК, розподіл рибосом уздовж транскрипту або відмінності між клітинними контекстами. Такий розвиток предметної галузі демонструє поступовий перехід від аналізу окремих ділянок мРНК до більш комплексного опису трансляційного процесу.

Для коректного аналізу існуючих підходів доцільно розрізняти три рівні роботи з даними: експериментальне отримання показників трансляційної активності, формування ознак мРНК та побудову прогнозної моделі. Експериментальні методи, такі як ribosome profiling і polysome profiling, формують кількісні цільові показники, зокрема translation efficiency або mean ribosome loading [2], [9]. Після цього послідовність мРНК перетворюється на набір ознак або безпосередньо подається до моделі глибинного навчання. Тому сучасні роботи доцільно оцінювати не лише за точністю прогнозування, а й за тим, які саме джерела інформації про мРНК вони використовують.

Одним із базових прикладів прогнозування трансляційної активності за 5′UTR є підхід, запропонований Sample та співавт. [11]. У цій роботі було використано масивний експериментальний набір даних, отриманий на основі рандомізованих людських 5′UTR, для яких за допомогою polysome profiling визначали mean ribosome loading. Така постановка дала змогу оцінити, наскільки сама послідовність 5′UTR здатна пояснювати рівень рибосомного навантаження.

На основі отриманих даних автори побудували модель глибинного навчання, яка пов’язувала нуклеотидну послідовність 5′UTR із прогнозованим рівнем MRL. Важливо, що модель використовувалася не лише для регресійного прогнозування, а й для аналізу впливу варіантів 5′UTR та проєктування нових регуляторних послідовностей із бажаними властивостями. Для даної роботи цей підхід є важливим, оскільки він підтверджує інформативність 5′UTR як окремого джерела ознак.

Водночас така постановка не охоплює повну послідовність мРНК і не враховує внесок CDS, 3′UTR, локальних структурних характеристик усієї молекули та клітинного контексту.

Роль кодуючої послідовності детальніше розглядається в роботі Nieuwkoop та співавт. [12]. На відміну від підходів, що зосереджені переважно на 5′UTR, у цій роботі аналізувалася повна CDS. Автори використовували синонімічно рандомізовані варіанти кодуючої послідовності червоного флуоресцентного білка та вимірювали рівні білкової продукції для різних варіантів. Далі ці дані застосовувалися для навчання моделей машинного навчання, зокрема Random Forest і LASSO.

Основний висновок цієї роботи полягає в тому, що для прогнозування рівня білкової продукції особливо важливою може бути початкова частина CDS. Зокрема, було показано, що інформація про перші кодони кодуючої послідовності вже містить суттєві ознаки для прогнозування рівня експресії білка. При цьому важливим фактором є не лише загальний рівень кодонної оптимізації, а й локальна вторинна структура мРНК у початковій частині CDS. Для теми даної роботи це важливо, оскільки підтверджує необхідність враховувати CDS не тільки як послідовність кодонів, але й як джерело локальних структурних та позиційних ознак.

Окреме значення для прогнозування ефективності трансляції мають підходи до обчислення вторинної структури РНК. Структурні характеристики не є безпосередньо заданими у послідовності, але можуть бути отримані за допомогою спеціалізованих алгоритмів прогнозування фолдингу. Зокрема, за допомогою інструментів пакета ViennaRNA можна розраховувати мінімальну вільну енергію, локальну стабільність структури та інші характеристики, пов’язані з потенційним згортанням РНК [13]. Такі ознаки особливо важливі для 5′UTR, ділянки навколо старт-кодону та початкової частини CDS, оскільки саме ці ділянки можуть впливати на ефективність розпізнавання старту трансляції та подальший перебіг синтезу білка.

Практичне застосування моделей прогнозування добре ілюструє підхід Castillo-Hair та співавт. [14]. У цій роботі 5′UTR розглядається не лише як об’єкт аналізу, а й як елемент, який можна цілеспрямовано оптимізувати для задач mRNA-delivered gene editing. Автори виконали polysome profiling бібліотек повністю або частково рандомізованих 5′UTR у декількох типах клітин і використали отримані дані для навчання моделей, здатних прогнозувати MRL та пропонувати нові регуляторні послідовності.

Особливість цього підходу полягає в переході від прогнозування до дизайну. За допомогою моделей глибинного навчання, gradient descent та генеративних нейронних мереж було створено нові 5′UTR, після чого їх експериментально перевірили у мРНК, що кодували компоненти систем редагування генів [14]. Для даної роботи цей приклад є важливим, оскільки він показує практичне значення прогнозних моделей трансляційної активності. Разом із цим, основний акцент у цій роботі зроблено саме на 5′UTR, тоді як повна послідовність транскрипту не є центральним об’єктом моделювання.

Показовим прикладом переходу до ширшої постановки задачі є модель RiboNN, запропонована Zheng та співавт. [15]. Цей підхід спрямований на прогнозування translation efficiency мРНК у клітинах ссавців на основі послідовнісних ознак. На відміну від багатьох попередніх моделей, які переважно аналізували лише 5′UTR, RiboNN враховує внесок повної послідовності мРНК, зокрема 5′UTR, CDS та 3′UTR. Це робить даний підхід особливо близьким до теми даної кваліфікаційної роботи.

RiboNN є багатозадачною згортковою нейронною мережею, яка прогнозує translation efficiency у різних клітинних контекстах [15]. Важливою особливістю моделі є врахування просторового розташування низькорівневих ди- та три-нуклеотидних ознак, зокрема кодонів. Це означає, що для прогнозування важлива не лише наявність певного мотиву, а й його позиція відносно інших елементів транскрипту. Для даної роботи це є принциповим, оскільки підтверджує доцільність поєднання повної

послідовності мРНК, позиційної інформації та клітинного контексту в межах єдиної моделі.

Окремий напрям становить моделювання не одного інтегрального показника, а розподілу рибосом уздовж транскрипту. Прикладом такого підходу є RiboMIMO, де ribosome density прогнозується вздовж CDS у постановці multi-input / multi-output [16]. На відміну від моделей, які повертають одне значення translation efficiency або mean ribosome loading, RiboMIMO дозволяє аналізувати локальну трансляційну динаміку та ділянки можливого накопичення рибосом.

Значення RiboMIMO для даної роботи полягає в тому, що ця модель підкреслює роль кодуючої послідовності у процесі трансляції. Якщо 5′UTR переважно впливає на ініціацію, то CDS може впливати на перебіг елонгації, позиційні ефекти та локальну щільність рибосом. Тому для прогнозування ефективності трансляції за повною послідовністю мРНК недоцільно обмежуватися лише регуляторною ділянкою перед старт-кодоном. Водночас RiboMIMO не є прямим розв’язанням задачі прогнозування translation efficiency, оскільки її основна мета полягає в побудові профілю ribosome density уздовж CDS.

Проведений огляд показує, що сучасні підходи поступово переходять від аналізу окремих фрагментів мРНК до комплекснішого моделювання трансляційної активності. Підходи, орієнтовані на 5′UTR, добре описують етап ініціації трансляції, але не враховують повністю внесок CDS і 3′UTR. Моделі, що аналізують CDS або ribosome density, краще відображають процес елонгації, але не завжди дають інтегральний прогноз translation efficiency. Контекстно-залежні моделі, такі як RiboNN, демонструють важливість клітинного середовища та повної послідовності мРНК, однак питання поєднання послідовнісних, структурних і контекстних ознак залишається актуальним.

1.6 Проблеми та обмеження існуючих підходів

Попри активний розвиток методів прогнозування ефективності трансляції мРНК, існуючі підходи мають низку обмежень, пов’язаних як із біологічною складністю процесу трансляції, так і з особливостями експериментальних даних. На відміну від задач, у яких цільовий показник визначається одним домінантним фактором, трансляційна ефективність формується під впливом багатьох взаємопов’язаних характеристик: послідовності 5′UTR, CDS і 3′UTR, локальної вторинної структури, контексту старт-кодону, кодонного складу, стабільності мРНК та клітинного середовища. Через це моделі, що враховують лише один тип ознак або одну ділянку транскрипту, не завжди здатні повністю описати реальний процес синтезу білка.

Одним із поширених обмежень є зосередженість багатьох моделей переважно на 5′UTR. Такий підхід є біологічно обґрунтованим, оскільки 5′UTR безпосередньо пов’язана з ініціацією трансляції та містить регуляторні елементи, що впливають на залучення рибосомного комплексу [6], [11]. Проте аналіз лише цієї ділянки не враховує впливу кодуючої послідовності, перших кодонів CDS, локальної структури в межах CDS, вибору синонімічних кодонів та можливих ділянок уповільнення елонгації. Унаслідок цього моделі, побудовані лише на 5′UTR, можуть добре описувати початковий етап трансляції, але не повністю відображати весь процес формування білкового продукту.

Іншим обмеженням є неповне використання повної послідовності мРНК. Тема даної роботи передбачає прогнозування за повною послідовністю транскрипту, тобто з урахуванням 5′UTR, CDS і 3′UTR. Однак у багатьох дослідженнях увага зосереджується лише на окремих фрагментах мРНК. Такий підхід спрощує постановку задачі, але може призводити до втрати частини регуляторної інформації. Зокрема, 3′UTR не кодує білок, але може впливати на стабільність мРНК, її локалізацію,

взаємодію з мікроРНК і регуляторними білками [7]. Тому ігнорування 3′UTR обмежує можливість комплексного опису трансляційної активності.

Суттєвою проблемою є використання синтетичних або репортерних бібліотек. Такі набори даних є зручними для контрольованих експериментів, оскільки дозволяють змінювати окремі елементи послідовності та вимірювати їхній вплив на трансляцію. Наприклад, рандомізовані бібліотеки 5′UTR дають змогу безпосередньо оцінювати внесок цієї ділянки у mean ribosome loading [11], [14]. Водночас синтетичні конструкції не завжди повністю відповідають властивостям ендогенних мРНК. Природні транскрипти мають складнішу організацію, різну довжину UTR і CDS, додаткові регуляторні елементи та функціонують у складному клітинному середовищі.

Особливо важливою є проблема переносимості моделей між різними наборами даних. Модель може демонструвати високу якість на даних, на яких вона була навчена, але втрачати точність при застосуванні до інших експериментальних умов, клітинних ліній або типів мРНК. У роботі Schlusser та співавт. зазначено, що сучасні моделі глибинного навчання для прогнозування трансляції часто погано узагальнюються на інші набори даних, зокрема на ендогенні мРНК, які відрізняються від репортерних конструкцій багатьма властивостями [3]. Це обмежує практичне використання таких моделей для прогнозування трансляційної активності нових або природних транскриптів.

Окремим викликом є врахування клітинного контексту. Ефективність трансляції не є сталою властивістю самої послідовності мРНК. Одна й та сама мРНК може транслюватися з різною ефективністю в різних типах клітин або за різних експериментальних умов. Це пов’язано з відмінностями у складі факторів трансляції, активності регуляторних білків, наявності мікроРНК, метаболічному стані клітини та інших характеристиках біологічного середовища. Підходи, які ігнорують клітинний контекст,

можуть бути недостатньо точними при перенесенні на інші типи клітин або тканин. Саме тому моделі, подібні до RiboNN, враховують контекстну залежність translation efficiency як важливу складову прогнозування [15].

Ще одним обмеженням є неоднорідність цільових показників. У різних роботах використовуються translation efficiency, mean ribosome loading, ribosome density або інші похідні характеристики трансляційної активності. Хоча ці показники пов’язані з роботою рибосом на мРНК, вони не є повністю тотожними. Translation efficiency зазвичай відображає співвідношення між сигналом ribosome profiling і рівнем мРНК, mean ribosome loading характеризує середню кількість рибосом на транскрипті, а ribosome density описує локальний розподіл рибосом уздовж послідовності [2], [8], [9], [6]. Через це результати моделей, навчених на різних цільових змінних, не завжди можна безпосередньо порівнювати між собою.

Складним залишається і коректне врахування вторинної структури РНК. У багатьох підходах структурний вплив подається спрощено, наприклад через одне значення мінімальної вільної енергії для всієї ділянки або для фіксованого вікна біля старт-кодону. Такий підхід є зручним для машинного навчання, але не повністю описує реальну структурну поведінку мРНК. MFE характеризує лише одну енергетично найвигіднішу передбачену структуру, тоді як молекула РНК може існувати в ансамблі альтернативних конформацій. Крім того, значення MFE залежить від довжини фрагмента, тому його складно безпосередньо порівнювати між короткими та довгими ділянками без нормалізації [13].

Також існує проблема роботи з довгими послідовностями. Повні мРНК можуть мати значну довжину, а це ускладнює їх безпосереднє використання в моделях глибинного навчання. Довгі послідовності потребують більшої обчислювальної пам’яті, збільшують час навчання та можуть містити як інформативні, так і слабко релевантні ділянки. Через це необхідно обирати спосіб подання повної мРНК: обрізання послідовності до

фіксованої довжини, використання локальних вікон, масок функціональних ділянок або виділення агрегованих числових ознак. Кожен із цих підходів має власні переваги та обмеження.

Важливою проблемою є інтеграція різнорідних вхідних даних у межах однієї прогнозної моделі. Послідовність мРНК є дискретною символьною інформацією, MFE та GC-вміст є числовими ознаками, маски функціональних ділянок задають позиційний контекст, а клітинний контекст може бути представлений категоріальними або числовими змінними. Тому модель повинна не лише обробляти кожну групу ознак окремо, а й коректно об’єднувати їх у єдине представлення. Саме ця проблема є однією з причин доцільності мультимодального підходу.

Окремо слід зазначити проблему інтерпретованості моделей глибинного навчання. Багато сучасних архітектур здатні знаходити складні залежності в біологічних послідовностях, однак не завжди зрозуміло, які саме ознаки найбільше впливають на кінцевий прогноз. Для біологічних задач це є суттєвим обмеженням, оскільки важливо не лише отримати числове передбачення, а й зрозуміти, які елементи мРНК пов’язані з підвищенням або зниженням ефективності трансляції. Без такої інтерпретації складніше використовувати модель для пояснення впливу конкретних мутацій або для цілеспрямованого проєктування нових послідовностей.

2 РОЗРОБЛЕННЯ МЕТОДУ МУЛЬТИМОДАЛЬНОГО ПРОГНОЗУВАННЯ ЕФЕКТИВНОСТІ ТРАНСЛЯЦІЇ МРНК

2.1 Формальна постановка задачі прогнозування ефективності трансляції мРНК

Проведений у першому розділі аналіз показав, що ефективність трансляції мРНК залежить від сукупності послідовнісних, структурних, позиційних і контекстних факторів. До таких факторів належать повна послідовність транскрипту, функціональна організація мРНК на 5′UTR, CDS та 3′UTR, наявність upstream AUG, GC-вміст, локальна вторинна структура РНК, мінімальна вільна енергія, а також клітинний контекст за наявності відповідних даних. Тому задача прогнозування ефективності трансляції не може бути зведена лише до аналізу однієї ділянки мРНК або одного типу ознак.

У межах даної роботи задача полягає у розробленні методу мультимодального прогнозування ефективності трансляції мРНК за повною послідовністю транскрипту на основі глибинного навчання. Основна ідея методу полягає в тому, щоб подати різнорідні характеристики однієї мРНК у вигляді декількох взаємодоповнювальних модальностей, обробити їх окремими компонентами моделі та об’єднати в єдине приховане представлення для подальшого регресійного прогнозування.

Нехай задано набір транскриптів:

$$D = \{(X_i, y_i)\}_{\{i=1\}}^{\{n\}}, \tag{2.1}$$

де D – навчальний набір даних;

X_i – сукупність вхідних ознак і-го транскрипту;

y_i – експериментально виміряне значення трансляційної активності і-го транскрипту;

n – кількість транскриптів у наборі даних.

У загальному випадку вхідний опис і-го транскрипту можна подати як набір модальностей:

$$X_i = \{S_i, R_i, F_i, K_i\}, \tag{2.2}$$

де S_i – повна нуклеотидна послідовність мРНК;

R_i – позиційне представлення функціональних ділянок мРНК, зокрема 5′UTR, CDS та 3′UTR;

F_i – числові та структурні ознаки мРНК;

K_i – ознаки клітинного або експериментального контексту, якщо такі дані доступні.

Повна послідовність мРНК S_i розглядається як упорядкована послідовність нуклеотидів:

$$S_i = (s_1, s_2, \dots, s_L), \tag{2.3}$$

де s_j – нуклеотид у позиції j;

L – довжина послідовності мРНК.

У практичній реалізації нуклеотиди можуть бути подані символами A, C, G, U або A, C, G, T залежно від формату вихідного набору даних. Для задачі машинного навчання така послідовність перетворюється на числове представлення, яке може бути використане нейронною мережею. При цьому важливо зберегти порядок нуклеотидів, оскільки регуляторні мотиви, кодони та локальні структурні властивості залежать не лише від складу послідовності, а й від позиційного розташування її елементів.

Позиційне представлення R_i використовується для того, щоб модель могла розрізняти функціональні ділянки повної мРНК. Один і той самий нуклеотидний мотив може мати різну біологічну інтерпретацію залежно від того, чи розташований він у 5′UTR, CDS або 3′UTR. Наприклад, AUG

у 5′UTR може розглядатися як upstream AUG, тоді як AUG у CDS є кодоном метіоніну в межах білкової послідовності. Тому для прогнозування за повною послідовністю мРНК доцільно додатково враховувати маски або ознаки, що описують належність позицій до окремих функціональних ділянок.

До складу F_i входять числові та структурні характеристики мРНК. До таких ознак можуть належати довжина повної послідовності, довжини 5′UTR, CDS та 3′UTR, GC-вміст, кількість upstream AUG, частоти k-мерів, мінімальна вільна енергія для окремих ділянок або локальних вікон, а також нормалізовані структурні показники. Ці ознаки доповнюють послідовнісне представлення, оскільки деякі біологічно важливі властивості можуть бути складними для безпосереднього виділення з сирої послідовності при обмеженому обсязі навчальних даних.

Контекстна компонента K_i використовується лише за наявності відповідної інформації в наборі даних. До неї можуть належати тип клітин, тканина, експериментальні умови або інші характеристики біологічного середовища, у якому було виміряно трансляційну активність. Необхідність врахування клітинного контексту пов’язана з тим, що translation efficiency не є сталою властивістю лише самої послідовності мРНК і може змінюватися залежно від клітинного середовища [15]. Якщо такі дані відсутні, модель може бути реалізована без контекстної компоненти, а її використання розглядатиметься як напрям подальшого розширення методу.

Вихідним значенням моделі є кількісний показник трансляційної активності. У межах даної роботи основною цільовою змінною розглядається translation efficiency, оскільки вона безпосередньо відповідає темі прогнозування ефективності трансляції мРНК. За наявності іншого типу експериментальних даних цільовою змінною також може бути mean ribosome loading. Обидва показники є числовими, тому задача належить до задач регресійного прогнозування.

Необхідно побудувати функцію прогнозування:

$$\widehat{\{y\}}_i = f\theta(X_i), \quad (2.4)$$

де $\widehat{\{y\}}_i$– прогнозоване значення трансляційної активності і-го транскрипту;

$f\theta$ – модель глибинного навчання з параметрами θ;

X_i – мультимодальне вхідне представлення і-го транскрипту.

Метою навчання є підбір таких параметрів θ, за яких прогнозовані значення $\widehat{\{y\}}_i$будуть якомога ближчими до експериментально виміряних значень y_i. Для цього використовується функція втрат, що кількісно оцінює похибку прогнозування на навчальному наборі даних. У найпростішому випадку такою функцією може бути середньоквадратична похибка:

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2. \quad (2.5)$$

Мінімізація цієї функції дозволяє навчити модель встановлювати залежність між властивостями мРНК та відповідним показником трансляційної активності. Оскільки вхідні дані мають різну природу, доцільним є окреме оброблення послідовнісних, позиційних, структурно-статистичних та контекстних ознак із подальшим об’єднанням отриманих представлень. Це забезпечує формування комплексного представлення вхідних даних, у якому враховується інформація, отримана з кожної доступної групи ознак.

Таким чином, формальна постановка задачі полягає у побудові регресійної мультимодальної моделі, яка на основі повної послідовності мРНК, розмітки її функціональних ділянок, структурних і статистичних ознак, а також клітинного контексту за його наявності прогнозує числовий показник ефективності трансляції. Така постановка дозволяє перейти від аналізу окремих елементів транскрипту до комплексного врахування факторів, що впливають на синтез білка.

2.2 Обґрунтування вибору мультимодального підходу

Вибір мультимодального підходу для задачі прогнозування ефективності трансляції мРНК зумовлений тим, що трансляційна активність не визначається одним окремим фактором. Як було показано у першому розділі, на рівень синтезу білка впливають послідовність мРНК, функціональна належність її ділянок, локальна вторинна структура, статистичні характеристики транскрипту та клітинний контекст. Ці джерела інформації мають різну природу, тому їх недоцільно зводити лише до одного типу вхідного представлення.

У задачах побудови інтелектуальних систем важливе значення має формалізація предметної галузі, структуроване подання інформації та встановлення зв’язків між її елементами. У роботі, присвяченій розробці методу генерації веб-контенту на основі онтологічного підходу, підкреслюється доцільність використання структурованого подання інформації для автоматизованої обробки даних у межах інтелектуальної системи [17]. Для задачі прогнозування ефективності трансляції мРНК ця ідея може бути інтерпретована як необхідність узгодженого подання різних характеристик одного транскрипту: послідовності, функціональних ділянок, структурних ознак і контекстних параметрів.

Мультимодальне машинне навчання загалом спрямоване на побудову моделей, які здатні обробляти та пов’язувати інформацію з декількох різних модальностей. У таких задачах важливими є не лише окремі представлення кожного типу даних, а й спосіб їх поєднання, тобто формування спільного прихованого представлення для подальшого прогнозування [18]. У межах даної роботи під модальностями розуміються різні групи ознак мРНК, що описують один і той самий об’єкт з різних боків.

Першою модальністю є послідовнісне представлення. Воно описує порядок нуклеотидів у повній мРНК і дозволяє моделі виявляти локальні мотиви, кодони, k-мери та інші послідовнісні закономірності. Таке

представлення є базовим, оскільки саме послідовність містить первинну інформацію про 5′UTR, CDS і 3′UTR. Однак використання лише сирої послідовності може бути недостатнім, особливо якщо модель має самостійно навчитися розрізняти функціональні ділянки та виділяти структурно важливі фрагменти.

Другою модальністю є позиційне представлення функціональних ділянок мРНК. Воно необхідне для того, щоб модель могла розуміти, у якій частині транскрипту розташований певний мотив або фрагмент послідовності. Наприклад, AUG у 5′UTR може бути upstream AUG і впливати на ініціацію трансляції, тоді як AUG у CDS є звичайним кодоном у межах білкової послідовності. Тому однакова локальна послідовність може мати різну біологічну інтерпретацію залежно від її позиційного контексту.

Третьою модальністю є структурно-статистичні ознаки. До них належать GC-вміст, довжини функціональних ділянок, кількість upstream AUG, частоти k-мерів, мінімальна вільна енергія та нормалізовані MFE-показники для окремих вікон. Ці ознаки є корисними тому, що вони явно подають певні біологічно значущі властивості мРНК. Наприклад, MFE дозволяє врахувати потенційну стабільність вторинної структури, а GC-вміст може бути пов’язаний із загальною схильністю фрагмента до формування стабільних пар нуклеотидів.

Четвертою потенційною модальністю є клітинний або експериментальний контекст. Необхідність його врахування пояснюється тим, що translation efficiency не є сталою властивістю лише самої послідовності мРНК. Один і той самий транскрипт може демонструвати різну трансляційну активність у різних клітинних типах, тканинах або експериментальних умовах. Це підтверджується сучасними підходами, зокрема RiboNN, де translation efficiency прогнозується з урахуванням багатьох клітинних контекстів [15]. Якщо в обраному наборі даних контекстна інформація відсутня, відповідна модальність може бути

виключена з практичної реалізації, але методологічно вона залишається важливим напрямом розширення моделі.

Перевага мультимодального підходу полягає в тому, що кожна група ознак може бути оброблена способом, який відповідає її природі. Послідовність мРНК доцільно обробляти засобами, здатними виявляти локальні мотиви та позиційні закономірності. Позиційні маски можуть використовуватися для позначення функціональних ділянок транскрипту. Числові ознаки можуть бути подані у вигляді векторів і оброблені повнозв'язними шарами. Контекстні ознаки, за наявності, можуть бути представлені категоріальними або числовими змінними. Після окремої обробки ці представлення можуть бути об'єднані в єдиний вектор ознак. Узагальнено процес об'єднання модальностей можна подати так:

$$z_i = \left[z_i^{\{seq\}};\ z_i^{\{pos\}};\ z_i^{\{feat\}};\ z_i^{\{ctx\}} \right], \tag{2.6}$$

де z_i – спільне мультимодальне представлення і-го транскрипту;

$z_i^{\{seq\}}$ – приховане представлення послідовності мРНК;

$z_i^{\{pos\}}$ – представлення позиційної інформації про 5′UTR, CDS та 3′UTR;

$z_i^{\{feat\}}$ – представлення структурно-статистичних ознак;

$z_i^{\{ctx\}}$ – представлення клітинного або експериментального контексту за його наявності.

Після формування спільного представлення z_i модель може використовувати його для прогнозування кількісного показника трансляційної активності:

$$\widehat{\{y\}}_i = g\theta(z_i), \tag{2.7}$$

де $\widehat{\{y\}}_i$ – прогнозоване значення translation efficiency або mean ribosome loading;

$g\theta$ – регресійна частина моделі з параметрами θ.

Використання мультимодального підходу також дозволяє проводити подальший аналіз внеску окремих груп ознак. Наприклад, можна порівнювати модель, що використовує лише послідовність, із моделлю, яка додатково враховує MFE, GC-вміст, маску CDS або контекстні ознаки. Такий аналіз дає змогу оцінити, які саме модальності покращують якість прогнозування, а які мають обмежений внесок для обраного набору даних.

Окремою перевагою мультимодального підходу є його гнучкість. Якщо в одному наборі даних доступні лише повна послідовність і CDS-маска, модель може бути реалізована без контекстної компоненти. Якщо в іншому наборі даних додатково наявна інформація про клітинний тип або умови експерименту, ця інформація може бути включена як окрема модальність. Таким чином, запропонований підхід не обмежується одним конкретним форматом даних, а задає загальну схему інтеграції різнорідної інформації про мРНК.

Разом із тим, мультимодальний підхід має певні складності. По-перше, різні типи ознак мають різні масштаби, формати та розмірності, тому потребують окремої попередньої обробки. По-друге, не всі модальності можуть бути доступні в кожному наборі даних. По-третє, збільшення кількості ознак і компонентів моделі може підвищувати ризик перенавчання, особливо за обмеженої кількості експериментальних зразків. Тому при розробленні методу необхідно поєднувати біологічно обґрунтовані ознаки з контрольованою складністю моделі.

Таким чином, вибір мультимодального підходу є обґрунтованим, оскільки задача прогнозування ефективності трансляції мРНК потребує одночасного врахування повної послідовності транскрипту, функціональних ділянок, структурних характеристик, статистичних ознак і, за наявності даних, клітинного контексту. Такий підхід відповідає біологічній природі процесу трансляції та дозволяє перейти від ізольованого

аналізу окремих ознак до комплексного прогнозування трансляційної активності мРНК.

2.3 Формування вхідного представлення мРНК

Після формальної постановки задачі необхідно визначити спосіб подання мРНК у вигляді, придатному для використання в моделі глибинного навчання. На біологічному рівні мРНК є послідовністю нуклеотидів із внутрішньою функціональною організацією, однак нейронна мережа не може безпосередньо працювати з текстовими символами A, C, G, U або T. Тому першим етапом побудови методу є перетворення біологічної послідовності в числове представлення з одночасним збереженням інформації про порядок нуклеотидів і належність окремих позицій до функціональних ділянок. Таке представлення має бути достатньо формалізованим для подальшої обробки алгоритмами машинного навчання. Це дозволяє перейти від опису мРНК як біологічного об'єкта до її використання як структурованого вхідного вектора або набору векторів у межах нейронної архітектури. Загальну логіку формування мультимодального вхідного представлення мРНК наведено на рисунку 2.1.

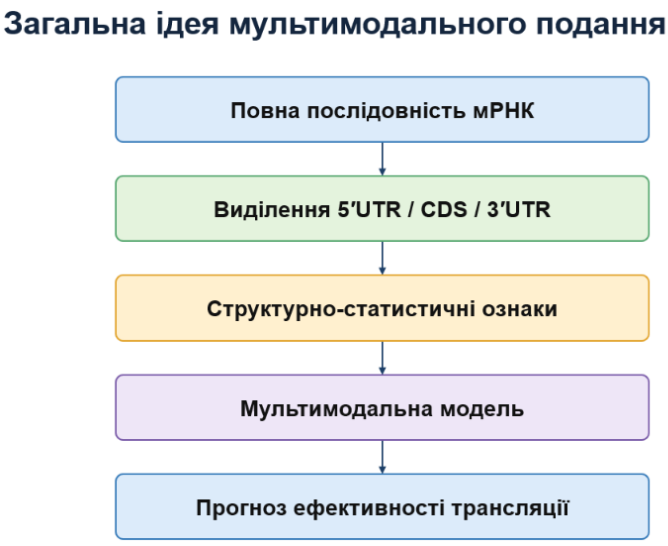


Рисунок 2.1 – Загальна ідея мультимодального подання мРНК

Як показано на рисунку 2.1, формування вхідного представлення починається з повної послідовності мРНК. На її основі виділяються функціональні ділянки 5′UTR, CDS і 3′UTR, що дозволяє зберегти біологічний контекст окремих позицій транскрипту. Далі для цієї ж послідовності формуються структурно-статистичні ознаки, які описують довжину, нуклеотидний склад, локальні мотиви та потенційні структурні характеристики мРНК.

Отримані компоненти розглядаються як взаємодоповнювальні частини одного вхідного представлення. Повна послідовність зберігає порядок нуклеотидів, регіональна інформація вказує на функціональну належність позицій, а числові ознаки узагальнюють структурні та статистичні властивості транскрипту. Таке подання створює основу для подальшого використання мультимодальної моделі прогнозування ефективності трансляції.

У межах даної роботи повна послідовність мРНК розглядається як основне джерело вхідної інформації. Такий підхід відповідає темі роботи, оскільки прогнозування виконується не лише за 5′UTR або CDS, а за повною послідовністю транскрипту. Повна мРНК містить 5′UTR, CDS та 3′UTR, тому її подання має враховувати як сам порядок нуклеотидів, так і позиційний контекст кожного фрагмента. Це важливо, оскільки однакові мотиви можуть мати різне біологічне значення залежно від того, у якій частині транскрипту вони розташовані.

Нехай повна послідовність і-го транскрипту має вигляд:

$$S_i = (s_1, s_2, \dots, s_L), \tag{2.8}$$

де S_i– повна послідовність і-го транскрипту;

s_j – нуклеотид у позиції j;

L – довжина послідовності.

У біологічному записі мРНК використовуються нуклеотиди A, C, G та U. Водночас у багатьох наборах даних послідовності транскриптів подаються у форматі, де замість U використовується T, тобто у вигляді A, C, G, T. Для обчислювальної моделі це є технічним представленням послідовності, тому всі символи можуть бути приведені до єдиного алфавіту. У практичній реалізації доцільно використовувати алфавіт:

$$A = \{A, C, G, T, N\}, \tag{2.9}$$

де A, C, G і T – основні нуклеотиди;

N – невизначені або невідомі позиції.

Для подання послідовності в числовому вигляді кожному символу алфавіту ставиться у відповідність ціле число:

$$A \rightarrow 1, C \rightarrow 2, G \rightarrow 3, T \rightarrow 4, N \rightarrow 0. \tag{2.10}$$

Таке кодування дозволяє перетворити текстову послідовність у вектор індексів:

$$Q_i = (q_1, q_2, \dots, q_L), \tag{2.11}$$

де Q_i – числове представлення послідовності;

q_j – числовий код нуклеотиду в позиції j.

Значення 0 доцільно використовувати для невизначених нуклеотидів або для доповнення коротших послідовностей до фіксованої довжини. Це дозволяє формувати батчі даних під час навчання нейронної мережі. Якщо послідовності мають різну довжину, вони можуть бути доповнені спеціальним символом PAD або обрізані до максимальної довжини L_{max}. При цьому важливо використовувати маску допустимих позицій, щоб модель або функція втрат не враховувала штучно додані елементи як біологічно значущі. Маску реальних позицій можна подати так:

$$M_j = \begin{cases} 1, \text{якщо позиція } j \text{ відповідає реальному нуклеотиду} \\ 0, \text{якщо позиція } j \text{ є доповненням.} \end{cases}, \quad (2.12)$$

де M_j – значення маски для позиції j.

Після числового кодування послідовність може бути перетворена у векторні представлення за допомогою шару embedding. На відміну від простого one-hot кодування, embedding-представлення дозволяє моделі навчити щільні вектори для кожного нуклеотиду або службового символу. У такому випадку кожен код q_j відображається у вектор фіксованої розмірності:

$$e_j = Emb(q_j), \quad (2.13)$$

де e_j – векторне представлення нуклеотиду в позиції j;

$Emb(\cdot)$ – навчувана функція embedding-перетворення.

У результаті повна послідовність мРНК подається не як рядок символів, а як матриця:

$$E_i = (e_1, e_2, \dots, e_L), \quad (2.14)$$

де E_i – послідовнісне векторне представлення і-го транскрипту.

Таке представлення зберігає порядок нуклеотидів і може бути використане згортковими або іншими нейромережевими шарами для виявлення локальних мотивів, k-мерів, кодонів та інших послідовнісних закономірностей. Це особливо важливо для задачі прогнозування ефективності трансляції, оскільки значення має не лише наявність певного мотиву, а й його положення в межах повної мРНК [15].

Однак послідовнісного представлення недостатньо для повного опису транскрипту. Модель також повинна мати інформацію про те, які позиції належать до 5′UTR, CDS або 3′UTR. Для цього формується позиційне

представлення функціональних ділянок. Якщо відома межа CDS, то для кожної позиції повної мРНК можна визначити її належність до однієї з трьох основних ділянок:

$$r_j = \begin{cases} 0, \text{якщо позиція } j \text{ належить до } 5'UTR, \\ 1, \text{якщо позиція } j \text{ належить до } CDS, \\ 2, \text{якщо позиція } j \text{ належить до } 3'UTR. \end{cases}, \quad (2.15)$$

де r_j– код функціональної ділянки для позиції j.

Таке представлення дозволяє моделі відрізняти однакові нуклеотидні фрагменти, розташовані в різних частинах мРНК. Наприклад, триплет AUG у 5′UTR може бути регуляторним upstream AUG, тоді як у CDS він є кодоном метіоніну. Тому позиційна інформація є необхідною для коректної біологічної інтерпретації послідовності. Альтернативно функціональні ділянки можуть бути подані у вигляді one-hot маски:

$$R_j = [I(j \in 5'UTR), I(j \in CDS), I(j \in 3'UTR)], \quad (2.16)$$

де R_j – вектор позиційної розмітки для позиції j;

$I(\cdot)$ – індикаторна функція, що дорівнює 1, якщо умова виконується, і 0 в іншому випадку.

У такому випадку кожна позиція описується не одним числовим кодом, а тривимірним вектором, який явно показує належність позиції до певної функціональної ділянки. Це зручно для подальшого об’єднання з послідовнісним embedding-представленням. Наприклад, для кожної позиції можна сформувати розширене представлення:

$$h_j = [e_j;\ R_j], \quad (2.17)$$

де h_j – об’єднане позиційно-послідовнісне представлення позиції j;

e_j – embedding-представлення нуклеотиду;

R_j – one-hot представлення функціональної ділянки.

У результаті модель отримує не лише інформацію про те, який нуклеотид знаходиться в певній позиції, а й інформацію про біологічну роль цієї позиції в структурі транскрипту. Це дає змогу краще враховувати різницю між регуляторними некодуючими ділянками та кодуючою послідовністю.

Окремо потрібно враховувати змінну довжину мРНК. Повні транскрипти можуть суттєво відрізнятися за довжиною, тому безпосереднє використання повних послідовностей у нейронній мережі може бути обчислювально складним. Для практичної реалізації можливі декілька підходів. Перший полягає у доповненні послідовностей до фіксованої довжини L_{max} використанні маски реальних позицій. Другий полягає в обрізанні надто довгих послідовностей із пріоритетним збереженням біологічно важливих ділянок, наприклад 5′UTR, ділянки навколо старт-кодону, початкової частини CDS та частини 3′UTR. Третій підхід полягає у використанні локальних вікон, для яких окремо обчислюються послідовнісні та структурні ознаки.

У межах запропонованого методу повна послідовність мРНК розглядається як основа для формування кількох груп вхідних ознак. По-перше, з неї формується послідовнісне представлення для нейронної мережі. По-друге, на основі меж CDS визначаються функціональні ділянки 5′UTR, CDS та 3′UTR. По-третє, з цієї ж послідовності можуть бути обчислені додаткові статистичні та структурні ознаки, зокрема GC-вміст, кількість upstream AUG, частоти k-мерів та MFE для окремих ділянок.

2.4 Формування структурних та статистичних ознак мРНК

Окрім послідовнісного та позиційного представлення, для прогнозування ефективності трансляції доцільно використовувати

додаткові структурні та статистичні ознаки мРНК. Такі ознаки дозволяють явно подати біологічно значущі характеристики транскрипту, які можуть бути важливими для трансляції, але не завжди ефективно виділяються моделлю безпосередньо з сирої нуклеотидної послідовності. Сюди належать: GC-вміст, кількість upstream AUG, частоти k-мерів, мінімальна вільна енергія та нормалізовані структурні показники.

Першою групою статистичних ознак є довжини функціональних ділянок мРНК. Оскільки повна мРНК складається з 5′UTR, CDS та 3′UTR, для кожного транскрипту можна визначити довжину кожної з цих ділянок. Якщо відомі позиції початку та завершення CDS, то довжини функціональних ділянок можна подати так:

$$L_{5'UTR} = p_{start} - 1, \tag{2.18}$$

$$L_{CDS} = p_{stop} - + 1, \tag{2.19}$$

$$L_{3'UTR} = L - p_{stop}, \tag{2.20}$$

де $L_{5'UTR}$ – довжина 5′-нетрансльованої ділянки;

p_{start} – позиція початку CDS;

L_{CDS} – довжина кодуючої послідовності;

p_{stop} – позиція завершення CDS;

$L_{\{3'UTR\}}$ – довжина 3′-нетрансльованої ділянки;

L – довжина повної послідовності мРНК.

Довжини ділянок є важливими ознаками, оскільки вони відображають структурну організацію транскрипту. Наприклад, довга 5′UTR може містити більше регуляторних елементів, upstream AUG або локальних вторинних структур, які впливають на ініціацію трансляції. Довжина CDS пов’язана з розміром білкового продукту та тривалістю елонгації, а 3′UTR може містити

регуляторні мотиви, що впливають на стабільність мРНК та її посттранскрипційну регуляцію.

Наступною ознакою є GC-вміст. Він визначає частку гуаніну та цитозину в аналізованій послідовності або окремій її ділянці:

$$GC_r = \frac{N_G^{(r)} + N_C^{(r)}}{L_r}, \tag{2.21}$$

де GC_r – GC-вміст ділянки r;

$N_G^{(r)}$ – кількість нуклеотидів G у ділянці r;

$N_C^{(r)}$ – кількість нуклеотидів C у ділянці r;

L_r – довжина ділянки r.

У межах методу GC-вміст доцільно обчислювати не лише для повної мРНК, а й окремо для 5′UTR, CDS та 3′UTR. Такий підхід дає змогу врахувати, що однаковий загальний GC-вміст може відповідати різним розподілам нуклеотидів між функціональними ділянками. Наприклад, високий GC-вміст у 5′UTR може мати інший вплив на трансляцію, ніж високий GC-вміст у віддаленій частині 3′UTR.

Кількість upstream AUG характеризує кількість триплетів AUG, розташованих перед основним старт-кодоном CDS. Такі елементи можуть впливати на сканування рибосомного комплексу. Для i-го транскрипту кількість upstream AUG можна визначити за формулою:

$$uAUG = \sum_{j=1}^{p_{start}-3} I(s_j s_{j+1} s_{j+2} = AUG), \tag{2.22}$$

де uAUG – кількість upstream AUG у 5′UTR;

p_{start} – позиція основного старт-кодону CDS;

s_j – нуклеотид у позиції j;

$I(\cdot)$ – індикаторна функція, що дорівнює 1, якщо умова виконується, і 0 в іншому випадку.

Якщо послідовність у наборі даних подана в ДНК-подібному алфавіті A, C, G, T, то при практичному обчисленні цієї ознаки замість AUG використовується відповідний триплет ATG. У тексті роботи біологічно коректним є запис AUG, а в програмній реалізації може використовуватися ATG залежно від формату даних.

Ще однією групою ознак є k-мери. k-мером називається фрагмент послідовності довжини k. Наприклад, для k = 3 такими фрагментами є триплети нуклеотидів. Частота конкретного k-мера може бути подана як:

$$freq(m) = \frac{count(m)}{L_r - k + 1}, \tag{2.23}$$

де $freq(m)$ – відносна частота k-мера m;

$count(m)$ – кількість входжень k-мера m у відповідній ділянці;

L_r – довжина аналізованої ділянки;

k – довжина k-мера.

k-мерні ознаки дозволяють описати локальний склад послідовності без явного ручного визначення всіх можливих біологічних мотивів. Для задачі прогнозування ефективності трансляції доцільно розглядати частоти k-мерів окремо для 5′UTR, CDS та 3′UTR або для вибраних локальних вікон. Наприклад, триплетні ознаки можуть бути пов’язані з кодонним складом CDS, тоді як k-мери в 5′UTR можуть відображати регуляторні мотиви, що впливають на ініціацію трансляції.

Важливою структурною ознакою є мінімальна вільна енергія, або MFE. Цей показник використовується для кількісного опису стабільності передбаченої вторинної структури РНК. У межах методу MFE може бути обчислена для окремих функціональних ділянок або локальних вікон мРНК. Для прогнозування вторинної структури та MFE доцільно використовувати спеціалізовані інструменти, зокрема RNAfold із пакета ViennaRNA [13].

Оскільки MFE залежить від довжини аналізованого фрагмента, безпосереднє порівняння значень для ділянок різної довжини може бути некоректним. Тому для кожного вікна або функціональної ділянки доцільно використовувати нормалізоване значення:

$$MFE_{norm}^{(r)} = \frac{MFE^{(r)}}{L_r}, \tag{2.24}$$

де $MFE_{norm}^{(r)}$ – нормалізована мінімальна вільна енергія ділянки r;

$MFE^{(r)}$ – мінімальна вільна енергія цієї ділянки;

L_r – довжина ділянки r.

Для задачі прогнозування ефективності трансляції найбільш інформативними можуть бути не лише глобальні структурні ознаки всієї мРНК, а й локальні структурні характеристики. Зокрема, доцільно розглядати 5′UTR, ділянка навколо старт-кодону, початкову частину CDS та інші фрагменти, де вторинна структура може впливати на доступність послідовності для рибосомного комплексу. Такий підхід дозволяє врахувати локальні структурні ефекти, які можуть бути втрачені при використанні одного загального MFE для всієї послідовності.

У практичному методі для кожного транскрипту формується вектор структурно-статистичних ознак:

$$F_i = \begin{bmatrix} L_i, L_{5'UTR,i}, L_{CDS,i}, L_{3'UTR,i}, GC_i, GC_{5'UTR,i}, GC_{CDS,i}, \\ GC_{3'UTR,i}, uAUG_i, KMER_i, MFE_i \end{bmatrix}, \tag{2.25}$$

де F_i – вектор структурно-статистичних ознак і-го транскрипту;

L_i – довжина повної мРНК;

$L_{5'UTR,i}, L_{CDS,i}, L_{3'UTR,i}$ – довжини функціональних ділянок;

$GC_i, GC_{5'UTR,i}, GC_{CDS,i}$, $GC_{3'UTR,i}$, – GC-вміст повної мРНК та окремих ділянок;

$uAUG_i$ – кількість upstream AUG;

$KMER_i$ – вектор k-мерних ознак;

MFE_i – набір MFE-ознак для вибраних ділянок або локальних вікон.

Перед поданням числових ознак до моделі їх доцільно нормалізувати або стандартизувати. Це необхідно тому, що різні ознаки мають різні масштаби: довжина транскрипту може вимірюватися тисячами нуклеотидів, GC-вміст є часткою в межах від 0 до 1, а MFE може бути від'ємним значенням у ккал/моль. Стандартизацію ознаки x можна подати так:

$$x' = \frac{x - \mu_x}{\sigma_x}, \tag{2.26}$$

де x' – стандартизоване значення ознаки;

x – початкове значення ознаки;

μ_x – середнє значення ознаки в навчальному наборі;

σ_x – стандартне відхилення ознаки в навчальному наборі.

Стандартизація дозволяє зменшити вплив масштабу окремих ознак на процес навчання моделі та зробити числові характеристики придатнішими для об'єднання з іншими модальностями. При цьому параметри нормалізації повинні визначатися лише за навчальною вибіркою, а потім застосовуватися до валідаційної та тестової вибірок. Це запобігає витоку інформації з тестових даних у процес навчання.

2.5 Архітектура мультимодальної моделі глибинного навчання

Після формування вхідного представлення мРНК та обчислення структурно-статистичних ознак необхідно визначити архітектуру моделі, яка буде використовувати ці дані для прогнозування ефективності трансляції. Оскільки вхідна інформація має різну природу, доцільно будувати модель не як одну послідовну нейронну мережу для всіх ознак одразу, а як мультимодальну архітектуру з кількома окремими гілками

обробки. Кожна гілка відповідає за певний тип інформації, а після виділення прихованих представлень ці гілки об’єднуються в єдиний вектор, який використовується для регресійного прогнозування. Такий підхід дає змогу незалежно опрацьовувати різнорідні дані та враховувати особливості їхнього представлення на різних етапах моделювання. Крім того, модульна структура архітектури спрощує адаптацію моделі до нових типів ознак і забезпечує більшу гнучкість під час експериментального налаштування.

У межах запропонованого методу модель складається з послідовнісно-регіонального енкодера, гілки структурно-статистичних ознак, шару об’єднання представлень та регресійної голови. Контекстна гілка може бути додана лише за наявності відповідних даних, однак у межах використаного набору даних вона не застосовувалася. Така архітектура дозволяє поєднати аналіз повної послідовності мРНК, функціональної організації транскрипту, структурних характеристик РНК та клітинного контексту. Узагальнену архітектуру мультимодальної моделі прогнозування наведено на рисунку 2.2.

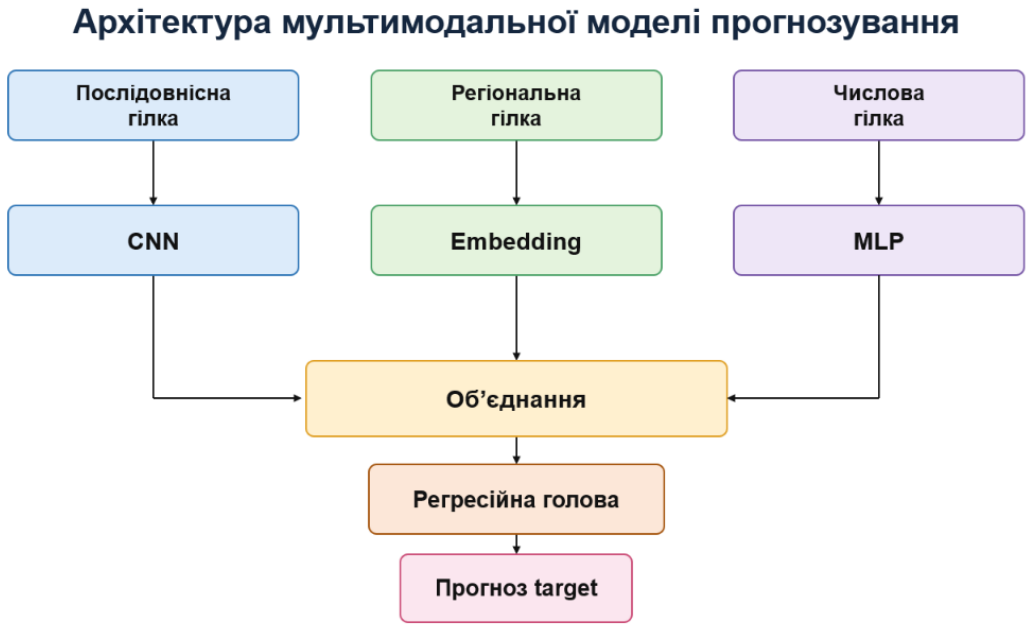


Рисунок 2.2 – Архітектура мультимодальної моделі прогнозування

Як видно з рисунка 2.2, модель складається з трьох основних гілок обробки. Послідовнісна гілка призначена для аналізу повної нуклеотидної послідовності мРНК за допомогою згорткової нейронної мережі. Регіональна гілка використовує інформацію про належність позицій до 5′UTR, CDS або 3′UTR і перетворює її на векторне представлення. Числова гілка обробляє структурно-статистичні ознаки, сформовані на попередньому етапі.

Після окремої обробки різні представлення об’єднуються в єдиний вектор, який надходить до регресійної голови. Оскільки задача полягає у прогнозуванні числового значення translation efficiency, вихід моделі є одним регресійним значенням target. Така архітектура дозволяє одночасно використовувати локальні патерни послідовності, функціональний контекст позицій і явно обчислені числові характеристики транскрипту.

Послідовнісна гілка призначена для обробки повної нуклеотидної послідовності мРНК. На вхід цієї гілки подається числове або embedding-представлення послідовності, сформоване на попередньому етапі. Основна задача цієї гілки полягає у виявленні локальних послідовнісних закономірностей, таких як короткі мотиви, k-мери, кодони, фрагменти навколо старт-кодону та інші позиційно залежні патерни. Для цього доцільно використовувати одномірні згорткові шари, оскільки вони здатні аналізувати локальні вікна послідовності та виділяти ознаки, що повторюються в різних позиціях транскрипту. Подібна логіка використовується у сучасних моделях прогнозування translation efficiency, зокрема в RiboNN, де врахування просторового розташування ди- та три-нуклеотидних ознак є важливим для якості прогнозування [15].

Нехай E_i є embedding-представленням повної послідовності i-го транскрипту. Тоді приховане представлення послідовнісної гілки можна подати як:

$$z_i^{seq} = CNN_{seq}(E_i), \tag{2.27}$$

де z_i^{seq} – приховане представлення послідовності мРНК;

$CNN_{seq}(\cdot)$ – послідовність одномірних згорткових, активаційних і pooling-шарів.

Одномірна згортка дає змогу виділяти локальні послідовнісні ознаки. Для позиції t та фільтра j згорткове перетворення можна подати так:

$$h_{t,j} = \sigma\left(b_j + \sum_{r=0}^{K-1} \sum_{d=1}^{D} W_{j,d,r} e_{t+r,d} \right), \quad (2.28)$$

де $h_{t,j}$ – значення j-го фільтра в позиції t;

$\sigma(\cdot)$ – функція активації;

b_j – зміщення;

K – розмір згорткового вікна;

D – розмірність embedding-вектора;

$W_{j,d,r}$ – вагові коефіцієнти фільтра;

$e_{t+r,d}$ – d-й компонент embedding-вектора в позиції t+r.

У практичній реалізації як функцію активації доцільно використовувати ReLU:

$$ReLU(x) = \max(0, x). \quad (2.29)$$

Використання ReLU дозволяє вводити нелінійність у модель і зменшує проблему насичення активацій. Після згорткових шарів може застосовуватися pooling, який агрегує локальні ознаки та формує компактніше представлення послідовності. Наприклад, глобальний max-pooling можна подати так:

$$z_j = \max_t h_{t,j}, \quad (2.30)$$

де z_j – агреговане значення j-го фільтра;

$h_{t,j}$ – значення цього фільтра в позиції t.

Позиційна гілка призначена для обробки інформації про функціональні ділянки мРНК. На її вхід подається маска або кодування ділянок 5′UTR, CDS та 3′UTR. Така інформація необхідна для того, щоб модель могла розрізняти однакові мотиви в різних частинах транскрипту. Позиційна гілка може бути реалізована як окрема згорткова гілка для масок функціональних ділянок або як додаткові канали, що об’єднуються з embedding-представленням послідовності на ранньому етапі.

Якщо позиційна інформація додається безпосередньо до послідовнісного представлення, то для кожної позиції формується розширений вектор:

$$h_j^{input} = [e_j;\ R_j], \tag{2.31}$$

де h_j^{input} – об’єднане представлення позиції j;

e_j – embedding-представлення нуклеотиду;

R_j – one-hot або інше числове представлення функціональної ділянки.

Такий варіант відповідає ранньому об’єднанню послідовнісної та позиційної інформації. Його перевага полягає в тому, що згорткові фільтри одразу отримують інформацію не лише про нуклеотиди, а й про біологічний контекст кожної позиції. Альтернативно позиційна гілка може бути оброблена окремо, а її вихід буде об’єднаний з іншими модальностями.

Гілка структурно-статистичних ознак призначена для обробки вектора F_i, сформованого в підрозділі 2.4. На відміну від послідовності, ці ознаки вже мають вигляд числового вектора фіксованої розмірності. Тому для їх обробки доцільно використовувати повнозв’язну нейронну мережу, або multilayer perceptron. Така гілка дозволяє моделі врахувати довжини ділянок, GC-вміст, кількість upstream AUG, k-мерні частоти та MFE-ознаки.

Приховане представлення цієї гілки можна подати так:

$$z_i^{feat} = MLP_{feat}(F_i), \quad (2.32)$$

де z_i^{feat} – приховане представлення структурно-статистичних ознак;

$MLP_{feat}(\cdot)$ – повнозв'язна нейронна мережа для обробки числових ознак;

F_i – вектор структурно-статистичних ознак і-го транскрипту.

Один шар MLP можна записати так:

$$a^l = \sigma(W^l a^{l-1} + b^l), \quad (2.33)$$

де a^l – вектор активацій l-го шару;

$\sigma(\cdot)$ – функція активації;

W^l – матриця ваг l-го шару;

a^{l-1} – вхід попереднього шару;

b^l – вектор зміщень.

Контекстна гілка є необов'язковою і використовується лише за наявності клітинного або експериментального контексту в наборі даних. Якщо доступна інформація про тип клітин, тканину або умови експерименту, вона може бути подана у вигляді one-hot кодування, embedding-представлення або числового вектора. Така гілка дозволяє моделі враховувати, що одна й та сама мРНК може мати різну ефективність трансляції в різних біологічних умовах. У загальному вигляді контекстну гілку можна подати так:

$$z_i^{ctx} = MLP_{ctx}(K_i), \quad (2.34)$$

де z_i^{ctx} – приховане представлення клітинного або експериментального контексту;

$MLP_{ctx}(\cdot)$ – компонент моделі для обробки контекстної інформації;

K_i – контекстні ознаки і-го прикладу.

Якщо у практичному наборі даних контекстні ознаки відсутні, компонент z_i^{ctx} не використовується, а модель працює з послідовнісною, позиційною та структурно-статистичною модальностями. Такий варіант не суперечить загальній постановці методу, оскільки контекстна гілка є розширенням архітектури, яке може бути активоване за наявності відповідних даних.

Після окремої обробки модальностей виконується їх об'єднання. У межах запропонованого методу доцільно використовувати конкатенацію прихованих представлень:

$$z_i = \left[z_i^{seq};\ z_i^{pos};\ z_i^{feat};\ z_i^{ctx}\right], \tag{2.35}$$

де z_i – спільне мультимодальне представлення і-го транскрипту;

z_i^{seq} – представлення послідовності;

z_i^{pos} – представлення позиційної інформації;

z_i^{feat} – представлення структурно-статистичних ознак;

z_i^{ctx} – представлення клітинного контексту за його наявності.

Якщо контекстна модальність не використовується, формула набуває вигляду:

$$z_i = \left[z_i^{seq};\ z_i^{pos};\ z_i^{feat}\right]. \tag{2.36}$$

Об'єднане представлення z_i подається до регресійної голови моделі. Регресійна голова складається з одного або декількох повнозв'язних шарів і формує кінцеве числове передбачення translation efficiency або mean ribosome loading:

$$\hat{y}_i = W_{out} \cdot \varphi\left(W_f z_i + b_f\right) + b_{out}, \tag{2.37}$$

де $\hat{y}_i$ – прогнозоване значення трансляційної активності;

W_{out}, b_{out} – параметри вихідного шару;

$\varphi(\cdot)$ – нелінійна функція активації;

$W_f + b_f$ – параметри прихованого шару регресійної голови.

Оскільки задача є регресійною, вихідний шар моделі має один нейрон без функції softmax. Це відрізняє дану постановку від задач класифікації, у яких модель прогнозує належність об'єкта до одного з кількох класів. У даному випадку модель повинна повернути одне числове значення, що характеризує трансляційну активність транскрипту.

Для зменшення ризику перенавчання в архітектурі можуть використовуватися dropout, нормалізація ознак, регуляризація ваг та рання зупинка навчання. Це особливо важливо для біологічних наборів даних, де кількість зразків може бути обмеженою, а кількість потенційних ознак високою. Крім того, порівняння повної мультимодальної моделі з її спрощеними варіантами дозволяє оцінити внесок кожної модальності в якість прогнозування.

Загальна логіка запропонованої архітектури полягає в тому, що послідовнісна гілка виявляє локальні мотиви та патерни в повній мРНК, позиційна гілка задає біологічний контекст цих патернів, гілка структурно-статистичних ознак додає явно обчислені характеристики транскрипту, а контекстна гілка за наявності даних враховує умови, у яких відбувається трансляція. Окреме оброблення кожної групи ознак дозволяє відповідним гілкам моделі спеціалізуватися на виділенні найбільш характерних для них закономірностей. Під час подальшого об'єднання ці ознаки взаємно доповнюють одна одну та формують більш повне представлення досліджуваної мРНК. Після об'єднання цих представлень модель формує інтегральний прогноз ефективності трансляції. Такий підхід відповідає загальній ідеї мультимодального машинного навчання, де різні типи даних обробляються окремо, але використовуються спільно для розв'язання однієї прогнозної задачі [18].

2.6 Методика навчання та оцінювання якості моделі

Після визначення архітектури мультимодальної моделі необхідно сформувати методику її навчання та оцінювання якості. Оскільки прогнозованим значенням є кількісний показник трансляційної активності, задача належить до задач регресії. Тому якість моделі повинна оцінюватися не за класифікаційними метриками, а за величиною похибки між експериментально виміряними та передбаченими значеннями, а також за показниками узгодженості між ними.

Для навчання та перевірки моделі набір даних поділяється на три неперетинні частини: навчальну, валідаційну та тестову вибірки:

$$D = D_{train} \cup D_{val} \cup D_{test}, \tag{2.38}$$

де D – повний набір даних;

D_{train} – навчальна вибірка;

D_{val} – валідаційна вибірка;

D_{test} – тестова вибірка.

Навчальна вибірка використовується для оптимізації параметрів моделі. Валідаційна вибірка застосовується для контролю якості під час навчання, вибору гіперпараметрів та ранньої зупинки. Тестова вибірка використовується лише після завершення навчання для фінального оцінювання узагальнювальної здатності моделі. Такий поділ дозволяє зменшити ризик переоцінювання якості моделі на даних, які вже використовувалися для підбору параметрів.

Якщо в наборі даних для одного гена наявні декілька транскриптів, бажано виконувати поділ не лише випадково за окремими записами, а з урахуванням групування за геном. Це необхідно для того, щоб дуже схожі транскрипти одного гена не потрапили одночасно до навчальної та тестової вибірок. Такий підхід робить оцінювання більш строгим, оскільки модель

перевіряється на транскриптах, які не мають близьких аналогів у навчальному наборі.

Основною функцією втрат для навчання моделі є середньоквадратична похибка. Для навчальної вибірки вона може бути подана так:

$$L_{train}(\theta) = \frac{1}{|D_{train}|} \sum_{i \in D_{train}} (y_i - \hat{y}_i)^2, \quad (2.39)$$

де $L_{train}(\theta)$ – значення функції втрат на навчальній вибірці;

θ – параметри моделі;

$|D_{train}|$ – кількість прикладів у навчальній вибірці;

y_i – експериментально виміряне значення трансляційної активності;

$\hat{y}_i$ – передбачене моделлю значення.

Мета навчання полягає в знаходженні таких параметрів моделі, за яких значення функції втрат буде мінімальним:

$$\theta^* = \arg\min_{\theta} L_{train}(\theta). \quad (2.40)$$

Середньоквадратична похибка є зручною для навчання нейронної мережі, оскільки вона сильніше штрафує великі помилки прогнозування. Це важливо для задачі прогнозування translation efficiency, де значні відхилення між передбаченим і реальним значенням можуть свідчити про те, що модель не врахувала важливі біологічні ознаки транскрипту.

Навчання моделі виконується протягом декількох епох. Під епохою розуміють один повний прохід моделі через навчальну вибірку. Після кожної епохи якість моделі перевіряється на валідаційній вибірці. Якщо валідаційна похибка перестає зменшуватися протягом заданої кількості епох, може застосовуватися рання зупинка навчання. Це дозволяє зменшити ризик перенавчання, коли модель починає запам’ятовувати навчальні

приклади, але гірше працює на нових даних. Для оцінювання якості прогнозування використовується декілька метрик. Першою з них є середня абсолютна похибка:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \quad (2.41)$$

де MAE – середня абсолютна похибка;

n – кількість прикладів у вибірці.

MAE показує середній абсолютний розмір помилки прогнозування. Ця метрика є інтерпретованою, оскільки вимірюється в тих самих одиницях, що й цільова змінна. Вона менш чутлива до окремих великих помилок, ніж середньоквадратична похибка. Другою метрикою є корінь із середньоквадратичної похибки:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}. \quad (2.42)$$

RMSE також вимірюється в одиницях цільової змінної, але сильніше штрафує великі відхилення між реальними та прогнозованими значеннями. Тому спільне використання MAE і RMSE дозволяє оцінити як середній рівень помилки, так і наявність значних відхилень у прогнозах [20].

Окрім метрик похибки, доцільно використовувати коефіцієнт кореляції Пірсона. Він показує, наскільки добре модель відтворює лінійну узгодженість між передбаченими та реальними значеннями:

$$r = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}\sqrt{\sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})^2}}, \quad (2.43)$$

де r – коефіцієнт кореляції Пірсона;

$\bar{y}$ – середнє значення експериментально виміряних показників;

$\bar{\hat{y}}$ – середнє значення передбачених показників.

Для додаткового аналізу може використовуватися коефіцієнт кореляції Спірмена. На відміну від кореляції Пірсона, він оцінює узгодженість між рангами значень, тобто показує, наскільки добре модель зберігає порядок транскриптів за рівнем трансляційної активності. Це важливо, якщо практична задача полягає не лише в точному передбаченні числового значення, а й у ранжуванні мРНК за очікуваною ефективністю трансляції.

Коефіцієнт Спірмена можна подати як кореляцію Пірсона між рангами:

$$\rho = r\big(rank(y), rank(\hat{y})\big), \tag{2.44}$$

де ρ – коефіцієнт кореляції Спірмена;

$rank(y)$ – ранги експериментально виміряних значень;

$rank(\hat{y})$ – ранги передбачених значень.

Для об’єктивної оцінки запропонованого методу доцільно порівнювати повну мультимодальну модель із простішими варіантами. Зокрема, можна розглядати такі конфігурації: модель лише за послідовністю, модель із послідовністю та позиційними масками, модель із додаванням структурно-статистичних ознак, а також повна модель з усіма доступними модальностями. Таке порівняння дозволяє оцінити внесок кожної групи ознак у загальну якість прогнозування.

Окремо може бути проведено ablation study, тобто послідовне вилучення окремих груп ознак із моделі. Наприклад, можна оцінити якість прогнозування без MFE-ознак, без k-мерів, без позиційної розмітки або без контекстної інформації. Якщо після вилучення певної модальності якість моделі помітно знижується, це свідчить про її значущий внесок у прогнозування.

Під час навчання необхідно контролювати не лише значення метрик, а й стабільність процесу оптимізації. Для цього доцільно аналізувати криві навчальної та валідаційної похибки. Якщо навчальна похибка продовжує зменшуватися, а валідаційна починає зростати, це може свідчити про перенавчання. Якщо ж обидві похибки залишаються високими, модель може бути недостатньо складною або вхідні ознаки можуть бути недостатньо інформативними.

Таким чином, методика навчання та оцінювання якості моделі включає поділ даних на навчальну, валідаційну та тестову вибірки, оптимізацію параметрів моделі за допомогою функції втрат, використання алгоритму Adam, контроль перенавчання та розрахунок регресійних метрик якості. Основними показниками оцінювання є MAE, RMSE, кореляція Пірсона та кореляція Спірмена. Їх спільне використання дозволяє оцінити модель як з точки зору величини похибки, так і з точки зору здатності правильно відтворювати відносний порядок транскриптів за ефективністю трансляції.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДУ МУЛЬТИМОДАЛЬНОГО ПРОГНОЗУВАННЯ ЕФЕКТИВНОСТІ ТРАНСЛЯЦІЇ МРНК

3.1 Загальна логіка програмної реалізації

Програмна реалізація методу мультимодального прогнозування ефективності трансляції мРНК була побудована як послідовний відтворюваний пайплайн обробки даних, формування ознак, навчання моделей та збереження результатів. Такий підхід було обрано з огляду на те, що задача прогнозування translation efficiency за повною послідовністю мРНК включає не лише побудову нейронної мережі, а й декілька попередніх етапів: завантаження біологічного набору даних, перевірку його коректності, виділення функціональних ділянок транскрипту, обчислення додаткових ознак, підготовку вибірок і контроль якості експерименту.

Основна ідея програмної реалізації полягає в тому, щоб кожен етап дослідження був оформлений як окремий програмний модуль або скрипт. Це дозволяє уникнути ситуації, коли вся логіка роботи з даними та моделлю знаходиться в одному великому файлі, який складно перевіряти, змінювати та повторно запускати. Окремі скрипти відповідають за окремі задачі: завантаження даних, аналіз вихідної таблиці, попередню обробку, формування ознак, навчання базових моделей, підготовку даних для PyTorch, реалізацію архітектур глибинного навчання, навчання нейронних моделей, підбір гіперпараметрів та фінальне оцінювання.

Загальна логіка роботи програмного пайплайну має такий вигляд:

– вихідний набір даних;

– завантаження та збереження raw-файлу;

– первинний аналіз і аудит даних;

– попередня обробка та фільтрація некоректних записів;

– виділення 5′UTR, CDS та 3′UTR;

– формування навчальної, валідаційної та тестової вибірок;

– обчислення структурно-статистичних ознак;

– навчання базових моделей машинного навчання;

– підготовка послідовностей і числових ознак для нейронної мережі;

– навчання sequence-only та мультимодальної моделей;

– підбір гіперпараметрів;

– фінальне оцінювання моделі;

– збереження метрик, прогнозів, графіків і навчених моделей.

Таким чином, реалізація охоплює повний цикл експериментального дослідження, а не лише окрему процедуру навчання нейронної мережі. Це важливо для кваліфікаційної роботи, оскільки якість кінцевої моделі залежить не тільки від архітектури, а й від коректності попередньої обробки біологічних даних. Наприклад, неправильна інтерпретація CDS-треку або некоректне обрізання довгих послідовностей могли б призвести до втрати біологічно значущої інформації та спотворення результатів прогнозування.

Першим етапом пайплайну є завантаження вихідного набору даних. У програмній реалізації передбачено окремий скрипт для отримання набору translation-efficiency-human та збереження його у локальному каталозі проєкту. Вихідний файл зберігається у каталозі data/raw і розглядається як незмінна копія початкових даних. Такий принцип є важливим, оскільки всі подальші перетворення мають виконуватися не над єдиною оригінальною таблицею, а над її похідними копіями. Це дозволяє повторити обробку з початку, якщо виникає потреба змінити логіку фільтрації або формування ознак.

Другим етапом є первинний аналіз і аудит даних. Він реалізований окремо від попередньої обробки, оскільки аналіз вихідної таблиці та її зміна є різними задачами. На етапі аудиту програма перевіряє структуру набору даних, наявність очікуваних стовпців, типи даних, пропущені значення, дублікати, довжини послідовностей і позиційних треків. Також виконується перевірка можливості відновлення функціональних ділянок мРНК. Результати цього етапу зберігаються у вигляді текстових звітів, таблиць і

графіків. Надалі ці матеріали можуть бути використані для опису набору даних у практичній частині роботи.

Третім етапом є попередня обробка даних. На цьому етапі програма вже не лише аналізує, а й формує очищені дані для моделювання. Зокрема, виконується уніфікація запису послідовностей, перевірка цільової змінної, відновлення меж CDS, виділення 5′UTR, CDS та 3′UTR, а також вилучення записів, для яких неможливо коректно визначити функціональні ділянки. Після цього дані поділяються на навчальну, валідаційну та тестову вибірки. Такий поділ потрібний для того, щоб навчання, налаштування гіперпараметрів і фінальне оцінювання виконувалися на різних частинах даних.

Четвертим етапом є формування структурно-статистичних ознак. Оскільки запропонований метод є мультимодальним, модель повинна отримувати не тільки сиру нуклеотидну послідовність, а й додаткові числові характеристики транскрипту. У програмній реалізації для кожного запису обчислюються довжини функціональних ділянок, GC-вміст, кількість upstream ATG, k-мерні ознаки, характеристики splice-треку та MFE-ознаки за наявності відповідного програмного модуля. Отриманий вектор числових ознак зберігається окремо та надалі використовується як вхід для структурно-статистичної гілки мультимодальної моделі.

П’ятим етапом є навчання базових моделей машинного навчання. Цей етап потрібний для того, щоб мати орієнтир якості, з яким можна порівнювати результати глибинного навчання. Базові моделі використовують сформовані числові ознаки, але не працюють безпосередньо з повною нуклеотидною послідовністю як послідовністю символів. У програмному пайплайні ці моделі винесено в окремий скрипт, щоб їхнє навчання та оцінювання не змішувалися з реалізацією нейронної мережі.

Шостим етапом є підготовка даних для моделей глибинного навчання. Для цього реалізовано окремий PyTorch Dataset, який перетворює

нуклеотидну послідовність у числове представлення, формує позиційні треки, підключає вектор числових ознак і повертає цільове значення. Оскільки повні мРНК мають різну довжину, у реалізації передбачено динамічне доповнення послідовностей у межах mini-batch. Це дозволяє працювати з повними послідовностями без жорсткого обрізання всіх транскриптів до одного фіксованого розміру.

Сьомим етапом є реалізація моделей глибинного навчання. У проєкті передбачено дві основні архітектури. Перша архітектура, SequenceOnlyCNN, використовує лише послідовнісну інформацію та слугує нейромережевим baseline. Друга архітектура, MultimodalTEModel, поєднує кілька джерел інформації: повну послідовність мРНК, позиційне подання функціональних ділянок і числові структурно-статистичні ознаки. Таке розділення дозволяє оцінити доцільність мультимодального підходу порівняно з використанням лише послідовності.

Восьмим етапом є навчання моделей і підбір гіперпараметрів. Для навчання використовується окремий скрипт, який завантажує підготовлені вибірки, створює модель, задає функцію втрат, оптимізатор, scheduler, виконує навчання по епохах, контролює якість на валідаційній вибірці та зберігає найкращий checkpoint. Окремий скрипт відповідає за підбір гіперпараметрів мультимодальної моделі. Такий підхід дозволяє відокремити звичайне навчання від експериментів із конфігураціями моделі.

Дев’ятим етапом є фінальне оцінювання. Після завершення навчання та вибору найкращої конфігурації модель оцінюється на тестовій вибірці. Цей етап винесено в окремий скрипт, оскільки тестова вибірка не повинна використовуватися під час підбору параметрів або прийняття рішень щодо архітектури. Програма обчислює регресійні метрики, зберігає прогнозовані значення, формує таблиці для порівняння моделей і створює графіки, необхідні для подальшого аналізу.

Окрему роль у програмній реалізації відіграє конфігураційний файл. У ньому зберігаються шляхи до даних, параметри debug-режиму, значення seed, параметри навчання, параметри архітектури та налаштування експерименту. Це дозволяє змінювати умови запуску без редагування основного коду. Наприклад, можна швидко перемикатися між тестовим режимом для перевірки працездатності пайплайну та повним режимом для проведення фінального експерименту.

У результаті програмна реалізація побудована як послідовна система взаємопов’язаних етапів. Кожен етап має чітко визначені вхідні та вихідні дані, а всі проміжні результати зберігаються у відповідних каталогах. Така організація спрощує перевірку коректності реалізації, дозволяє повторити експеримент і забезпечує прозорість отриманих результатів. У наступних підрозділах детально розглянуто програмне середовище, структуру файлів проєкту та реалізацію окремих етапів пайплайну.

3.2 Програмне середовище, бібліотеки та конфігурація проєкту

Програмну реалізацію методу виконано мовою Python. Вибір Python зумовлений наявністю розвиненої екосистеми бібліотек для обробки табличних даних, машинного навчання, глибинного навчання, роботи з біологічними послідовностями та побудови графіків. Крім того, Python є одним із найпоширеніших інструментів для експериментальних досліджень у галузі штучного інтелекту, що спрощує відтворення результатів, інтеграцію готових бібліотек і подальше розширення програмного рішення.

Програмний проєкт реалізовано у вигляді набору Python-скриптів, які виконуються послідовно. Така організація дозволяє розділити логіку роботи на окремі етапи: завантаження даних, аудит набору, попередня обробка, формування ознак, навчання базових моделей, навчання моделей глибинного навчання, підбір гіперпараметрів і фінальне оцінювання. Кожен

скрипт відповідає за окрему частину пайплайну та зберігає результати своєї роботи у відповідних каталогах проєкту.

Для роботи з вихідним набором даних використано бібліотеку Hugging Face Datasets. Вона застосовується на етапі завантаження набору translation-efficiency-human з репозиторію Hugging Face Hub. У програмній реалізації для цього використовується функція load_dataset, яка дозволяє отримувати набори даних за їхнім іменем у репозиторії та перетворювати їх у структури, придатні для подальшої обробки [21], [22]. Це забезпечує відтворюваність джерела даних, оскільки набір може бути повторно завантажений без ручного пошуку файлів.

Для роботи з табличними даними використано бібліотеку pandas. Вона застосовується для читання Parquet-файлів, формування таблиць, фільтрації записів, розрахунку статистик, об’єднання даних, збереження результатів у форматах CSV і Parquet. Основний вихідний файл даних зберігається у форматі Parquet, оскільки цей формат є зручним для компактного збереження великих табличних структур і підтримує ефективне читання вкладених або складних типів даних. Для завантаження локального файлу в програмній реалізації використовується функція pandas.read_parquet, яка повертає дані у вигляді DataFrame [23].

Бібліотека NumPy використовується для числових обчислень, роботи з масивами, обробки позиційних треків, обчислення довжин, масок і допоміжних векторів. Вона є базовим інструментом для операцій над числовими даними, що виникають під час аналізу послідовностей мРНК. Зокрема, NumPy використовується для пошуку позицій у CDS-треках, обчислення індексів функціональних ділянок і перетворення списків значень у числові масиви.

Для реалізації класичних моделей машинного навчання використано бібліотеку scikit-learn. Вона застосовується для навчання базових регресійних моделей, масштабування числових ознак, обчислення метрик і формування навчальної, валідаційної та тестової вибірок. Окремо важливим

є використання GroupShuffleSplit, який дозволяє виконувати розбиття даних із урахуванням груп. У межах даної роботи як група використовується gene, що дозволяє контролювати потрапляння записів одного гена до різних вибірок [24].

Для побудови моделей глибинного навчання використано бібліотеку PyTorch. Вона застосовується для реалізації власних класів Dataset, DataLoader, embedding-шарів, згорткових шарів, повнозв'язних шарів, функцій втрат, оптимізаторів і циклів навчання. PyTorch було обрано через його гнучкість при роботі з нестандартними вхідними даними, зокрема з послідовностями різної довжини та мультимодальними представленнями.

У програмній реалізації важливу роль відіграє механізм DataLoader. Він відповідає за формування mini-batch під час навчання моделі. Оскільки мРНК у наборі даних мають різну довжину, звичайне об'єднання прикладів у batch є недостатнім. Для цього у проєкті використовується власна функція collate_fn, яка формує batch із послідовностей різної довжини, виконує динамічне padding-доповнення та створює маску реальних позицій. У документації PyTorch collate_fn визначається як функція, що об'єднує список окремих зразків у mini-batch [25].

Для побудови графіків і візуалізацій використано бібліотеку matplotlib. Вона застосовується на етапах первинного аналізу даних, оцінювання результатів і побудови графіків для подальшого включення у кваліфікаційну роботу. За допомогою matplotlib формуються гістограми розподілу цільової змінної, графіки розподілу довжин послідовностей, графіки залишків, криві навчання та діаграми порівняння результатів моделей.

Для роботи з конфігураційними файлами використано бібліотеку PyYAML. Налаштування проєкту винесено у файл configs/config.yaml. У цьому файлі зберігаються шляхи до вхідних і вихідних даних, параметри режиму відлагодження, значення seed, параметри навчання, архітектурні параметри моделей, гіперпараметри та інші налаштування. Такий підхід

дозволяє змінювати параметри експерименту без ручного редагування основного програмного коду.

Файл конфігурації виконує декілька функцій. По-перше, він централізує всі основні налаштування проєкту. По-друге, він дозволяє відокремити логіку програми від конкретних параметрів запуску. По-третє, він підвищує відтворюваність експериментів, оскільки конфігурація, з якою було отримано певний результат, може бути збережена разом із метриками та checkpoint моделі. У випадку зміни архітектури або режиму навчання достатньо змінити відповідне значення у config.yaml, не переписуючи логіку скриптів.

У проєкті передбачено debug-режим. Він використовується для швидкої перевірки працездатності окремих етапів пайплайну на зменшеному обсязі даних або з меншою кількістю епох навчання. Такий режим є корисним на етапі розробки, оскільки дозволяє швидко виявляти помилки в коді без повного запуску всього експерименту. При цьому вихідний raw-файл не повинен скорочуватися або перезаписуватися в debug-режимі, оскільки він розглядається як незмінна копія початкового набору даних.

Для забезпечення відтворюваності результатів у програмній реалізації використовується фіксоване значення seed. Воно задається у конфігураційному файлі та застосовується для генераторів випадкових чисел Python, NumPy і PyTorch. Це дозволяє зменшити випадкові відмінності між запусками, зокрема під час розбиття даних на вибірки, ініціалізації параметрів моделі та формування mini-batch. Повна ідентичність результатів у глибинному навчанні може залежати від апаратного середовища та версій бібліотек, однак фіксація seed є необхідним кроком для підвищення відтворюваності експериментів.

Окремо у проєкті передбачено роботу з MFE-ознаками. Для цього може використовуватися модуль RNA з пакета ViennaRNA, якщо він доступний у програмному середовищі. MFE-ознаки не є обов’язковими для

запуску всього пайплайну: реалізація передбачає безпечне продовження роботи у випадку, якщо відповідний модуль не встановлено. Такий підхід дозволяє запускати основні етапи проєкту навіть у середовищах, де немає спеціалізованих біоінформатичних бібліотек, і водночас використовувати структурні ознаки там, де це можливо.

Таким чином, програмне середовище проєкту поєднує бібліотеки для роботи з даними, машинного навчання, глибинного навчання, візуалізації та конфігурації експериментів. Така комбінація інструментів дозволяє реалізувати повний пайплайн мультимодального прогнозування ефективності трансляції мРНК: від завантаження вихідного набору даних до навчання та оцінювання моделей. Використання конфігураційного файлу, фіксованого seed, окремих каталогів для даних і результатів, а також окремих скриптів для кожного етапу забезпечує структурованість, контрольованість і відтворюваність програмної реалізації.

3.3 Структура файлів програмного проєкту

Програмний проєкт організовано у вигляді набору каталогів і скриптів, кожен з яких відповідає за окремий етап реалізації методу мультимодального прогнозування ефективності трансляції мРНК. Така структура дозволяє розділити логіку роботи з даними, формування ознак, навчання моделей і аналіз результатів. Це спрощує перевірку коректності окремих етапів, повторний запуск експериментів і подальше розширення програмного рішення.

Кореневий каталог проєкту містить конфігураційні файли, вихідний код, вхідні дані та результати експериментів. Основними елементами структури є каталоги configs, data, results і src. Такий поділ відображає загальну логіку програмної реалізації: налаштування експерименту зберігаються окремо від коду, вихідні та оброблені дані – окремо від результатів, а всі програмні модулі згруповано в каталозі src.

Каталог configs містить файл config.yaml, у якому зберігаються основні параметри запуску проєкту. До таких параметрів належать шляхи до вхідних і вихідних файлів, значення seed, параметри debug-режиму, налаштування навчання, параметри архітектури моделей і параметри підбору гіперпараметрів. Використання окремого конфігураційного файлу дозволяє змінювати умови експерименту без ручного редагування основних скриптів. Це особливо важливо під час навчання моделей, оскільки різні етапи пайплайну можуть потребувати різних режимів запуску.

Каталог data призначено для збереження вхідних і проміжних даних. Він містить два основні підкаталоги: data/raw і data/processed. У data/raw зберігається вихідний набір даних te_human.parquet. Цей файл розглядається як початкова незмінна копія даних, завантажених із зовнішнього джерела. Усі подальші перетворення виконуються над похідними файлами, щоб зберегти можливість повторити обробку з початку. Каталог data/processed використовується для збереження очищених вибірок, сформованих після попередньої обробки, а також таблиць ознак, які використовуються під час навчання моделей.

Каталог results містить результати роботи програмного пайплайну. Він поділяється на декілька підкаталогів: results/metrics, results/figures, results/models і results/predictions. У results/metrics зберігаються текстові звіти, таблиці статистик, метрики моделей, результати підбору гіперпараметрів і підсумкові порівняльні таблиці. У results/figures зберігаються графіки, побудовані під час аналізу даних, навчання та оцінювання моделей. У results/models зберігаються checkpoint-файли навчених моделей. У results/predictions зберігаються таблиці з прогнозами моделей для тестової вибірки або окремих прикладів.

Каталог src є основним каталогом вихідного коду. У ньому розміщено скрипти, які реалізують окремі етапи програмного пайплайну. Кожен скрипт має конкретне призначення і працює з файлами, створеними на попередніх етапах. Завдяки цьому весь процес можна виконувати

послідовно: спочатку завантажити дані, потім перевірити їхню структуру, виконати попередню обробку, сформувати ознаки, навчити моделі та провести фінальне оцінювання.

Скрипт 01_download_data.py відповідає за завантаження вихідного набору даних і збереження його у локальному каталозі data/raw. Його задача полягає лише в отриманні початкового файлу та підготовці його до подальшої роботи в межах проєкту. На цьому етапі дані не очищуються, не фільтруються і не поділяються на вибірки. Такий підхід дозволяє відокремити отримання даних від їхньої подальшої обробки.

Скрипт 02_inspect_data.py реалізує первинний аналіз і аудит вихідного набору даних. Він перевіряє структуру таблиці, наявність очікуваних стовпців, типи даних, пропущені значення, дублікати, коректність довжин послідовностей і позиційних треків. Окрім цього, скрипт відновлює межі функціональних ділянок мРНК, перевіряє стартові та стоп-кодони, виявляє потенційно проблемні записи та зберігає звіти й графіки. Важливо, що цей скрипт не змінює вихідний набір даних, а лише аналізує його.

Скрипт 03_preprocess.py відповідає за попередню обробку даних. На цьому етапі відбувається очищення послідовностей, уніфікація алфавіту, перевірка цільової змінної, коректне визначення меж CDS, виділення 5′UTR, CDS і 3′UTR, а також фільтрація записів, для яких неможливо надійно відновити функціональні ділянки. Після цього скрипт формує навчальну, валідаційну та тестову вибірки і зберігає їх у каталозі data/processed. Саме цей етап створює основу для подальшого формування ознак і навчання моделей.

Скрипт 04_features.py реалізує формування структурно-статистичних ознак мРНК. Він використовує очищені дані, отримані після попередньої обробки, і для кожного транскрипту обчислює числові характеристики. До них належать довжини повної мРНК і функціональних ділянок, GC-вміст, кількість upstream ATG, k-мерні ознаки, характеристики splice-треку та

MFE-ознаки за наявності відповідного модуля. Отримані ознаки зберігаються у вигляді окремих таблиць для навчальної, валідаційної та тестової вибірок.

Скрипт 05_baseline.py призначено для навчання базових регресійних моделей машинного навчання. Він використовує сформовані числові ознаки і навчає класичні моделі, які надалі можуть бути використані як орієнтир для порівняння з моделями глибинного навчання. Виділення базових моделей в окремий файл дозволяє перевірити інформативність числових ознак незалежно від нейронної архітектури.

Файл 06_dataset.py містить реалізацію класів і допоміжних функцій для підготовки даних у форматі, придатному для PyTorch. У цьому модулі виконується кодування нуклеотидних послідовностей, підготовка позиційної інформації, підключення числових ознак і формування цільової змінної. Також у ньому реалізовано механізм формування mini-batch для послідовностей різної довжини. Цей файл є проміжною ланкою між підготовленими даними та моделями глибинного навчання.

Файл 07_models.py містить реалізацію нейронних архітектур. У ньому визначено sequence-only модель, яка використовує лише нуклеотидну послідовність, а також мультимодальну модель, яка поєднує послідовнісну інформацію, позиційне представлення та числові структурно-статистичні ознаки. Винесення моделей в окремий файл дозволяє змінювати архітектуру без внесення змін у скрипти попередньої обробки, навчання або оцінювання.

Скрипт 08_train.py реалізує навчання моделей глибинного навчання. Він завантажує підготовлені вибірки, створює модель відповідно до параметрів конфігурації, задає функцію втрат, оптимізатор, scheduler, виконує навчання по епохах і зберігає найкращий checkpoint. Також у цьому скрипті передбачено контроль якості на валідаційній вибірці та механізм ранньої зупинки. Його основна задача – навчити модель за заданою конфігурацією та зберегти результати навчання.

Скрипт 09_evaluate.py відповідає за фінальне оцінювання навченої моделі. Він завантажує checkpoint, застосовує модель до тестової вибірки, обчислює регресійні метрики, зберігає таблицю прогнозів і формує файли з підсумковими показниками. Цей скрипт відокремлено від навчання, оскільки тестова вибірка не повинна використовуватися під час підбору параметрів або вибору моделі.

Скрипт 10_analyze_results.py призначено для додаткового аналізу результатів після оцінювання моделі. Він може використовуватися для побудови графіків похибок, аналізу залишків, дослідження залежності помилок від довжини послідовностей або функціональних ділянок. Такий аналіз потрібний для подальшої інтерпретації якості роботи моделі, але самі результати цього етапу розглядаються вже в експериментальному розділі.

Скрипт 11_predict.py реалізує отримання прогнозу для окремих прикладів або підготовлених вхідних даних. Його призначення полягає в демонстрації того, як навчена модель може бути використана після завершення навчання. На відміну від 09_evaluate.py, який працює з тестовою вибіркою, цей скрипт орієнтований на застосування збереженої моделі до нових або вибраних прикладів.

Скрипт 12_app.py є допоміжним демонстраційним модулем. Він може використовуватися для спрощеного запуску або демонстрації роботи окремих компонентів проєкту. Наявність такого файлу не є обов’язковою для навчання моделі, однак він може бути корисним для перевірки працездатності кінцевої системи або демонстрації результатів.

Скрипт 13_tune_deep.py реалізує підбір гіперпараметрів мультимодальної моделі. Він запускає декілька конфігурацій моделі з різними значеннями параметрів, навчає їх на навчальній вибірці, оцінює на валідаційній вибірці та зберігає найкращу конфігурацію. Виділення цього етапу в окремий файл дозволяє не змішувати звичайне навчання моделі з експериментами з архітектурними та навчальними параметрами.

Окрім основних скриптів, у каталозі src містяться допоміжні файли. Файл utils.py містить спільні функції, які використовуються в різних частинах проєкту: завантаження конфігурації, створення каталогів, налаштування логування, фіксація seed, обчислення метрик і допоміжні операції з послідовностями. Такий підхід дозволяє уникнути дублювання коду в різних скриптах. Файл check_cuda.py використовується для перевірки доступності GPU та середовища CUDA. Файл export_raw_sequences_csv.py є допоміжним інструментом для експорту послідовностей у зручному табличному форматі, якщо це потрібно для перегляду або додаткового аналізу.

Таким чином, структура програмного проєкту побудована відповідно до етапів розробленого методу. Кожен каталог і кожен скрипт мають чітко визначене призначення. Така організація дозволяє послідовно виконувати всі етапи дослідження, контролювати проміжні результати, повторювати експерименти та використовувати збережені артефакти для подальшого аналізу в експериментальній частині роботи.

3.4 Реалізація завантаження та аудиту вихідного набору даних

Першим практичним етапом програмного пайплайну є отримання вихідного набору даних і перевірка його структури. Для цього в проєкті реалізовано два окремі скрипти: 01_download_data.py та 02_inspect_data.py. Їхнє розділення є принциповим, оскільки завантаження даних і аудит даних виконують різні функції. Перший скрипт відповідає за отримання та локальне збереження початкової таблиці, тоді як другий скрипт виконує її детальну перевірку без внесення змін до raw-файлу.

Скрипт 01_download_data.py призначений для завантаження набору translation-efficiency-human і збереження його у каталозі data/raw. На цьому етапі дані не очищуються, не фільтруються і не поділяються на навчальну, валідаційну та тестову вибірки. Основне завдання скрипта полягає в тому,

щоб створити локальну копію вихідного набору даних, яка надалі використовується всіма іншими етапами пайплайну.

Для завантаження даних використовується бібліотека Hugging Face Datasets. Вона дозволяє отримувати набори даних із Hugging Face Hub за допомогою функції load_dataset. У документації Hugging Face зазначено, що load_dataset може завантажувати датасети з Hub за назвою репозиторію, після чого дані можна використовувати в Python-коді для подальшої обробки [22]. У програмній реалізації назва набору даних зберігається у конфігураційному файлі, тому її можна змінити без редагування основного коду.

Після завантаження Hugging Face Dataset перетворюється у pandas DataFrame. Це потрібно для уніфікації подальшої обробки, оскільки більшість етапів пайплайну працюють із табличним представленням даних. Далі отримана таблиця зберігається у файл data/raw/te_human.parquet. Формат Parquet було обрано через його компактність і зручність для збереження структурованих табличних даних. У pandas для читання таких файлів використовується функція read_parquet, яка завантажує Parquet-об’єкт і повертає DataFrame [23].

Важливою особливістю реалізації є те, що скрипт завантаження не змінює зміст вихідного набору даних. Навіть якщо в конфігурації ввімкнено debug-режим, raw-файл має зберігатися повністю. Це зроблено для того, щоб каталог data/raw містив незмінну копію початкових даних. Усі скорочення, фільтрації або перетворення повинні виконуватися на наступних етапах і зберігатися у data/processed. Такий підхід дозволяє в будь-який момент повторити обробку з початку, не завантажуючи датасет повторно і не ризикуючи втратити частину вихідної інформації.

Після завершення завантаження скрипт виводить на екран базову інформацію про отриману таблицю: її розмір і список стовпців. Це дає змогу одразу переконатися, що файл було завантажено коректно і що структура таблиці відповідає очікуванням. Однак детальна перевірка даних не

виконується на цьому етапі, оскільки вона винесена в окремий скрипт аудиту.

Скрипт 02_inspect_data.py реалізує первинний аналіз і аудит вихідного набору даних. Його головна задача полягає не в підготовці даних для моделі, а в перевірці того, чи можна безпечно використовувати таблицю для подальшої обробки. Тому цей скрипт не змінює raw-файл, не видаляє записи і не створює навчальні вибірки. Він лише читає вихідний файл, аналізує його структуру, формує звіти та зберігає допоміжні таблиці й графіки.

На початку роботи 02_inspect_data.py завантажує файл data/raw/te_human.parquet і перевіряє наявність очікуваних стовпців. Для даної роботи очікуваними є transcript_id, gene, chromosome, sequence, cds, splice і target. Ця перевірка потрібна для того, щоб уникнути помилок на наступних етапах, де ці стовпці використовуються безпосередньо. Якщо певний стовпець відсутній або має неочікуваний тип, це має бути виявлено ще до попередньої обробки.

Далі виконується перевірка пропущених значень. Для кожного стовпця обчислюється кількість порожніх або відсутніх значень. Це важливо, оскільки пропуски в sequence, cds, splice або target можуть зробити окремий запис непридатним для навчання моделі. Наприклад, відсутність послідовності не дозволяє сформувати послідовнісне представлення, а відсутність target не дозволяє використовувати запис у задачі регресії з учителем.

Окремо виконується перевірка дублікатів. Скрипт аналізує як повні дублікати рядків, так і дублікати за окремими ключовими полями. Зокрема, перевіряються transcript_id, gene і sequence. Це потрібно для розуміння структури набору даних і можливих ризиків витоку інформації між вибірками. Наявність однакових або близьких записів у різних частинах даних може штучно покращити оцінку моделі, тому інформація про дублікати повинна бути відома до етапу розбиття.

Наступним кроком є аналіз цільової змінної target. Скрипт перевіряє, чи може target бути інтерпретований як числова величина, а також обчислює базові статистичні характеристики: мінімальне значення, максимальне значення, середнє, медіану, стандартне відхилення та квартилі. Така перевірка підтверджує, що задача є регресійною, а не класифікаційною, і що для подальшого оцінювання потрібно використовувати регресійні метрики.

Після перевірки target скрипт аналізує довжини нуклеотидних послідовностей. Для кожного запису обчислюється довжина sequence, а також формується статистика розподілу довжин. Цей аналіз є важливим для подальшої реалізації моделей глибинного навчання, оскільки послідовності мРНК можуть мати значно різну довжину. Інформація про мінімальну, медіанну, середню та максимальну довжину використовується для вибору режиму обробки послідовностей і для обґрунтування використання dynamic padding.

Важливим етапом аудиту є перевірка узгодженості sequence, cds і splice. Для кожного запису скрипт порівнює довжину основної послідовності з довжиною позиційних треків. Якщо довжина cds або splice не збігається з довжиною sequence, такий запис потенційно не може бути коректно використаний для відновлення позиційної інформації. Тому такі випадки фіксуються у звіті про якість даних.

Окрему увагу в 02_inspect_data.py приділено аналізу CDS-треку. У наборі даних стовпець cds містить позиційне представлення кодуючої ділянки. Для коректної подальшої роботи необхідно зрозуміти, як саме цей трек позначає CDS. У реалізації передбачено визначення позицій, у яких cds має активні значення, після чого за ними відновлюються межі кодуючої ділянки. Початок CDS визначається як перша активна позиція, а кінець CDS – на основі останньої активної позиції з урахуванням довжини кодона.

На основі відновлених меж CDS скрипт визначає функціональні ділянки мРНК: 5′UTR, CDS і 3′UTR. На цьому етапі ці ділянки використовуються лише для аналізу, а не для остаточного формування

навчальних вибірок. Скрипт обчислює довжину кожної ділянки, перевіряє можливість їхнього виділення і фіксує потенційно проблемні випадки. Це дозволяє ще до попередньої обробки оцінити, наскільки коректно набір даних підходить для мультимодального представлення.

Також у межах аудиту перевіряються стартові та стоп-кодони. Для кожного запису після відновлення CDS можна визначити перший кодон кодуючої ділянки та останній кодон. Така перевірка дозволяє виявити записи з неканонічними стартовими або стоп-кодонами. У межах скрипта ці записи не видаляються автоматично, оскільки не всі нестандартні випадки обов'язково є технічною помилкою. Вони фіксуються у файлі з проблемами як окремий тип попередження.

Результати роботи 02_inspect_data.py зберігаються у каталозі results/metrics та results/figures. У results/metrics формуються текстові та табличні звіти, що описують структуру набору даних і потенційні проблеми. Основним текстовим звітом є dataset_summary.txt. Він містить узагальнену інформацію про кількість записів, стовпці, пропущені значення, дублікати, статистику target, довжини послідовностей, коректність позиційних треків і виявлені проблеми.

Крім текстового звіту, скрипт зберігає декілька таблиць. Таблиця dataset_statistics.csv містить числові характеристики окремих записів, зокрема довжини послідовностей і функціональних ділянок. Таблиця data_quality_issues.csv містить список потенційних проблем, які були знайдені під час аудиту. Таблиця region_length_statistics.csv використовується для збереження статистик довжин 5′UTR, CDS і 3′UTR. Таблиця chromosome_counts.csv містить інформацію про кількість записів за хромосомами. Таблиця basic_feature_correlations.csv містить базові кореляції простих ознак із цільовою змінною.

У каталозі results/figures зберігаються графіки, сформовані на етапі аудиту. До них належать графіки розподілу цільової змінної, розподілу довжин послідовностей, розподілу довжин функціональних ділянок,

розподілу записів за хромосомами та залежності target від довжини послідовності. Ці графіки не використовуються для зміни даних, але потрібні для подальшої інтерпретації набору в експериментальному розділі.

Важливо, що аудит вихідного набору даних відокремлений від попередньої обробки. Це означає, що 02_inspect_data.py не приймає остаточних рішень про видалення записів. Він лише виявляє потенційні проблеми й документує їх. Остаточна фільтрація виконується вже у 03_preprocess.py, де записи обробляються відповідно до логіки формування навчальних вибірок. Такий поділ дозволяє зробити пайплайн прозорим: спочатку дані аналізуються, а потім обробляються на основі результатів цього аналізу.

Таким чином, реалізація завантаження та аудиту набору даних забезпечує контроль якості початкової інформації перед її використанням у моделях машинного навчання. Скрипт 01_download_data.py створює локальну незмінну копію вихідного набору, а скрипт 02_inspect_data.py перевіряє його структуру, цілісність і придатність для подальшої обробки. Така організація є необхідною для відтворюваного експерименту, оскільки дозволяє простежити весь шлях даних від початкового файлу до навчальних вибірок і моделей.

3.5 Реалізація попередньої обробки даних

Попередня обробка даних реалізована у скрипті 03_preprocess.py. Цей етап є проміжним між аудитом вихідного набору даних і формуванням ознак для моделей машинного навчання. На відміну від скрипту 02_inspect_data.py, який лише аналізує raw-файл і фіксує потенційні проблеми, скрипт 03_preprocess.py уже виконує практичне перетворення даних: очищує записи, відновлює функціональні ділянки мРНК, вилучає технічно некоректні приклади та формує навчальну, валідаційну і тестову вибірки.

Вхідним файлом для цього етапу є локальна копія вихідного набору даних, збережена у каталозі data/raw. Скрипт зчитує таблицю з файлу te_human.parquet, перевіряє наявність необхідних стовпців і готує дані до подальшого використання. Основними стовпцями, потрібними для попередньої обробки, є sequence, cds, splice, target, transcript_id і gene. Послідовність використовується для формування послідовнісного представлення, cds – для визначення кодуючої ділянки, splice – для позиційної інформації про сплайсинг, target – як цільова регресійна змінна, а gene – для групового розбиття даних.

Першим кроком попередньої обробки є уніфікація запису нуклеотидних послідовностей. Усі значення sequence приводяться до верхнього регістру, що дозволяє уникнути відмінностей між однаковими нуклеотидами, записаними великими або малими літерами. Якщо в послідовності трапляється символ U, він замінюється на T. Такий підхід пов’язаний із тим, що в аналізованому наборі даних послідовності подано в алфавіті A, C, G, T, хоча з біологічної точки зору вони відповідають мРНК, де замість T використовується U. Тому в програмній реалізації використовується технічне представлення A/C/G/T, а в біологічній інтерпретації воно відповідає A/C/G/U.

Після уніфікації регістру та алфавіту виконується перевірка символів послідовності. Допустимими вважаються символи A, C, G, T і N. Символ N використовується для позначення невизначеного нуклеотиду. Якщо послідовність містить символи, що не належать до допустимого алфавіту, такий запис має бути позначений як проблемний. Це потрібно для того, щоб на наступних етапах кодування послідовності не виникали помилки через невідомі символи.

Окремо перевіряється цільова змінна target. Оскільки задача прогнозування ефективності трансляції мРНК формулюється як задача регресії, target має бути числовим значенням. Скрипт перетворює target у числовий формат і перевіряє, чи не виникають при цьому некоректні

значення. Записи з відсутнім або нечисловим target не можуть використовуватися для навчання з учителем, оскільки для них неможливо обчислити функцію втрат.

Найважливішою частиною попередньої обробки є інтерпретація CDS-треку. У наборі даних стовпець cds не використовується як звичайна бінарна маска, де всі позиції кодуючої ділянки позначені одиницею. Натомість активні позиції в cds відповідають початкам кодонів у межах кодуючої ділянки. Тому для відновлення повних меж CDS недостатньо просто взяти всі позиції з одиницями як готову маску кодуючої ділянки. Потрібно визначити першу активну позицію як початок CDS, а останню активну позицію використати для визначення кінця CDS з урахуванням довжини кодона.

У програмній реалізації початок CDS визначається як індекс першої активної позиції у cds-треку. Кінець CDS визначається як індекс останньої активної позиції плюс три нуклеотиди. Додавання трьох позицій пояснюється тим, що остання активна позиція відповідає початку останнього кодона, а сам кодон має довжину три нуклеотиди. Така логіка дозволяє відновити повну кодуючу ділянку, а не лише позиції початку кодонів.

Після визначення cds_start і cds_end виконується виділення трьох функціональних ділянок мРНК. Ділянка 5′UTR визначається як частина послідовності від початку transcript до cds_start. Ділянка CDS визначається як фрагмент від cds_start до cds_end. Ділянка 3′UTR визначається як частина послідовності після cds_end до кінця transcript. Таким чином, для кожного коректного запису формується повне структурне розбиття зрілої мРНК на основні функціональні ділянки.

Крім самих послідовностей функціональних ділянок, скрипт формує допоміжні числові поля. До них належать довжина повної послідовності, довжина 5′UTR, довжина CDS, довжина 3′UTR, позиція початку CDS, позиція кінця CDS, стартовий кодон і стоп-кодон. Ці поля

використовуються як для подальшого формування числових ознак, так і для контролю коректності обробки. Наприклад, довжина CDS має бути додатною, а cds_end не повинен виходити за межі повної послідовності.

На етапі попередньої обробки також перевіряється відповідність довжин sequence, cds і splice. Якщо позиційні треки мають іншу довжину, ніж основна послідовність, то такий запис не можна коректно використовувати для позиційного представлення. У такому випадку запис позначається як некоректний і не включається до очищеного набору даних. Це дозволяє уникнути помилок під час формування region labels, splice-ознак і mini-batch для нейронної мережі.

Некоректні записи не видаляються безслідно. У програмній реалізації передбачено збереження інформації про вилучені або проблемні приклади у звітах каталогу results/metrics. Такий підхід забезпечує прозорість попередньої обробки: можна перевірити, які записи не потрапили до подальшого моделювання і з якої причини. Це важливо для відтворюваності експерименту та для пояснення логіки очищення даних у кваліфікаційній роботі.

Після очищення і виділення функціональних ділянок формується очищена таблиця даних. Вона зберігається у каталозі data/processed і використовується всіма наступними етапами пайплайну. Збереження очищеної таблиці окремо від raw-файлу дозволяє не повторювати попередню обробку при кожному запуску формування ознак або навчання моделей. При цьому raw-файл залишається незмінним і може бути використаний для повторної обробки у випадку зміни логіки preprocessing.

Наступним етапом є поділ очищених даних на навчальну, валідаційну та тестову вибірки. Навчальна вибірка використовується для оптимізації параметрів моделей. Валідаційна вибірка використовується для вибору гіперпараметрів, ранньої зупинки та вибору найкращого checkpoint. Тестова вибірка залишається відкладеною до фінального оцінювання і не використовується під час навчання або налаштування моделі.

Для поділу даних у програмній реалізації використовується груповий підхід за значенням gene. Це зроблено для зменшення ризику витоку інформації між вибірками. Якщо записи, пов’язані з одним і тим самим геном, потрапили б одночасно до навчальної та тестової вибірок, модель могла б частково використовувати подібність таких прикладів, що призвело б до завищеної оцінки якості. Тому всі записи з однаковим значенням gene повинні потрапляти лише до однієї з вибірок.

Для реалізації такого розбиття використовується GroupShuffleSplit з бібліотеки scikit-learn. Цей інструмент генерує випадкове розбиття з урахуванням груп, тобто ділить не окремі рядки незалежно один від одного, а множину унікальних груп [24]. У даному випадку групою є gene, тому поділ виконується так, щоб записи одного гена не розділялися між train, validation і test.

Формування трьох вибірок виконується у два кроки. Спочатку очищений набір поділяється на навчальну частину і тимчасову частину, яка містить дані для validation і test. Потім тимчасова частина додатково ділиться на валідаційну та тестову вибірки. Така процедура дозволяє отримати три неперетинні частини даних і водночас зберегти контроль над частками вибірок.

Після завершення розбиття скрипт перевіряє, чи немає перетину gene між сформованими вибірками. Для цього обчислюються множини gene для train, validation і test, після чого перевіряються їхні попарні перетини. Якщо перетини відсутні, це означає, що групове розбиття виконано коректно. У випадку виявлення перетину така ситуація мала б бути зафіксована як помилка, оскільки вона порушує логіку незалежного оцінювання.

Кожна з отриманих вибірок зберігається окремим файлом у каталозі data/processed. Це дозволяє наступним скриптам працювати з уже готовими частинами даних і не виконувати розбиття повторно. Такий підхід є важливим для відтворюваності, оскільки всі моделі навчаються та

оцінюються на одних і тих самих зафіксованих train, validation і test вибірках.

У результаті роботи 03_preprocess.py формуються очищені дані та набір проміжних артефактів, які використовуються далі в пайплайні. Цей етап забезпечує узгодженість біологічної структури мРНК із програмним представленням даних. Саме після попередньої обробки кожен транскрипт має визначені межі 5′UTR, CDS і 3′UTR, числову цільову змінну, коректну послідовність і належність до однієї з вибірок.

3.6 Реалізація формування структурно-статистичних ознак

Формування структурно-статистичних ознак реалізовано у скрипті 04_features.py. Цей етап є одним із ключових у програмній реалізації мультимодального підходу, оскільки саме тут для кожного транскрипту створюється додаткова числова модальність, яка доповнює послідовнісне представлення мРНК. Якщо на попередньому етапі дані було очищено, структуровано і поділено на навчальну, валідаційну та тестову вибірки, то на етапі формування ознак кожен запис перетворюється на набір кількісних характеристик, що описують довжину, нуклеотидний склад, функціональні ділянки та локальні структурні властивості мРНК.

Вхідними даними для скрипта 04_features.py є оброблені файли train.parquet, val.parquet і test.parquet, які були створені на етапі попередньої обробки. Для кожної з цих вибірок скрипт формує відповідну таблицю ознак. Такий підхід дозволяє гарантувати, що всі ознаки для навчальної, валідаційної та тестової частин будуть обчислені однаковим способом, але без змішування самих вибірок. У результаті створюються окремі файли train_features.parquet, val_features.parquet і test_features.parquet у каталозі data/processed.

Основна логіка формування ознак полягає в тому, щоб перетворити біологічно значущі характеристики мРНК у числові величини, які можуть

бути використані як вхід для класичних моделей машинного навчання та для MLP-гілки мультимодальної нейронної мережі. На відміну від сирої послідовності, яка подається до згорткової гілки моделі у вигляді закодованого ряду нуклеотидів, структурно-статистичні ознаки є компактним числовим описом окремих властивостей транскрипту.

Першою групою ознак є довжинні характеристики. Для кожного транскрипту обчислюється довжина повної послідовності мРНК, а також довжини трьох основних функціональних ділянок: 5′UTR, CDS і 3′UTR. Ці ознаки є важливими, оскільки довжина окремих ділянок може впливати на трансляційну активність. Наприклад, довга 5′UTR може містити додаткові регуляторні елементи або upstream старт-кодони, а довжина CDS і 3′UTR може бути пов’язана з особливостями стабільності транскрипту, швидкості трансляції та посттранскрипційної регуляції.

Другою групою ознак є GC-вміст. У скрипті обчислюється GC-вміст як для повної послідовності, так і окремо для 5′UTR, CDS і 3′UTR. GC-вміст визначається як частка нуклеотидів G і C серед усіх валідних нуклеотидів відповідного фрагмента. Для цього в коді використовується окрема допоміжна функція, яка приймає фрагмент послідовності та повертає числове значення частки GC. Якщо фрагмент має нульову довжину або не містить валідних нуклеотидів, функція повертає коректне службове значення, щоб уникнути ділення на нуль.

Окреме обчислення GC-вмісту для різних функціональних ділянок є важливим, оскільки однаковий загальний GC-вміст повної мРНК може приховувати різний розподіл нуклеотидного складу між 5′UTR, CDS і 3′UTR. Наприклад, послідовність може мати помірний GC-вміст у середньому, але високий GC-вміст у 5′UTR, що потенційно може бути пов’язано зі складнішою вторинною структурою в ділянці ініціації трансляції. Тому в мультимодальному представленні доцільно використовувати не тільки загальну характеристику, а й регіональні значення.

Третьою групою ознак є кількість upstream ATG перед основним старт-кодоном. Біологічно ця ознака відповідає кількості upstream AUG у 5′UTR. Однак у програмній реалізації використовується запис ATG, оскільки послідовності в наборі даних представлені в алфавіті A, C, G, T. При біологічному поясненні цей триплет відповідає AUG у молекулі мРНК. Скрипт визначає ділянку 5′UTR як фрагмент до початку CDS і рахує в ній кількість входжень триплету ATG. Така ознака може бути корисною для моделі, оскільки upstream старт-кодони можуть впливати на ефективність ініціації трансляції основної кодуючої ділянки.

Четвертою групою є k-мерні ознаки. У реалізації використовуються 3-мери, тобто всі можливі фрагменти довжини три нуклеотиди. Оскільки алфавіт складається з чотирьох основних символів A, C, G і T, кількість можливих 3-мерів становить 64. Для кожного транскрипту скрипт обчислює частоти появи таких 3-мерів у послідовності або у вибраних її частинах. Отримані частоти додаються до таблиці ознак як окремі числові стовпці.

Використання 3-мерних ознак має практичне значення, оскільки такі фрагменти можуть частково відображати локальний нуклеотидний склад, кодонний склад і короткі мотиви послідовності. Для класичних моделей машинного навчання k-мерні частоти є одним зі способів подати послідовність у фіксованому числовому вигляді. Для мультимодальної нейронної моделі вони виконують допоміжну роль, доповнюючи безпосередню послідовнісну гілку більш узагальненим статистичним описом.

П’ятою групою ознак є характеристики splice-треку. Оскільки вихідний набір даних містить не лише послідовність і CDS-трек, а й splice-трек, програмна реалізація передбачає використання інформації про позиційні особливості сплайсингу. На цьому етапі такі дані перетворюються на числові характеристики, які можуть бути використані моделлю разом з іншими структурними ознаками. Це дозволяє не ігнорувати додаткову позиційну модальність, присутню в наборі даних.

Окремим компонентом є обчислення MFE-ознак. MFE, або мінімальна вільна енергія, використовується як числова характеристика потенційної стабільності вторинної структури РНК-фрагмента. У програмній реалізації передбачено обчислення локальних MFE-ознак для фрагментів, які можуть бути важливими для трансляції. Зокрема, можуть використовуватися фрагмент 5′UTR, ділянка навколо старт-кодона та початкова частина CDS. Такі фрагменти обираються тому, що вони пов’язані з ініціацією трансляції та ранніми етапами руху рибосоми по мРНК.

Для обчислення MFE у проєкті використовується модуль RNA з пакета ViennaRNA, якщо він доступний у середовищі виконання. Оскільки цей модуль може бути відсутній на деяких комп’ютерах або в деяких віртуальних середовищах, код реалізовано таким чином, щоб відсутність ViennaRNA не призводила до аварійного завершення всього пайплайну. Якщо модуль доступний, MFE-ознаки обчислюються та додаються до таблиці. Якщо модуль недоступний, скрипт продовжує роботу без цих ознак і фіксує відповідне повідомлення. Такий підхід робить реалізацію стійкішою та переносимішою.

Оскільки обчислення MFE може бути відносно повільним для великої кількості послідовностей, у реалізації передбачено кешування результатів. Кешування дозволяє не виконувати повторний розрахунок для однакових фрагментів під час повторних запусків скрипта. Це особливо корисно на етапі розробки, коли формування ознак може запускатися кілька разів після змін у коді або конфігурації. Завдяки кешуванню зменшується час повторного виконання пайплайну.

Перед збереженням таблиць ознак скрипт виконує додаткову перевірку числових значень. Ознаки повинні бути представлені у форматі, придатному для машинного навчання, тобто мати числовий тип і не містити некоректних значень, які можуть призвести до помилок під час масштабування, навчання або обчислення метрик. Якщо певна ознака не

може бути обчислена для конкретного запису, у коді має бути передбачена контрольована обробка такого випадку. Це дозволяє уникнути прихованих помилок на наступних етапах.

У результаті роботи 04_features.py для кожної вибірки створюється окрема таблиця числових ознак. Вона зберігається окремо від основної таблиці з послідовностями, оскільки використовується різними частинами пайплайну. Базові моделі машинного навчання працюють саме з цими числовими таблицями. Мультимодальна нейронна модель використовує їх як одну з вхідних модальностей поряд із закодованою повною послідовністю та позиційною інформацією про функціональні ділянки.

Важливо, що формування ознак виконується після розбиття даних на train, validation і test. Це дозволяє обробляти кожну частину даних незалежно і зберігати чітку відповідність між основними таблицями та таблицями ознак. Крім того, такий підхід спрощує подальше масштабування ознак: параметри нормалізації можуть визначатися на навчальній вибірці, а потім застосовуватися до валідаційної та тестової вибірок.

Формування структурно-статистичних ознак є важливим етапом саме для мультимодального методу. Послідовнісна гілка нейронної мережі навчається автоматично виділяти локальні патерни з повної мРНК, але числові ознаки надають моделі додаткову узагальнену інформацію. Завдяки цьому модель може одночасно використовувати як низькорівневе представлення нуклеотидної послідовності, так і вищорівневі біологічно інтерпретовані характеристики транскрипту.

Таким чином, скрипт 04_features.py реалізує перетворення очищених даних у структурно-статистичну модальність. На цьому етапі повна послідовність мРНК доповнюється числовими характеристиками її довжини, складу, функціональних ділянок, локальних мотивів, splice-треку та потенційної вторинної структури. Отримані таблиці ознак використовуються як для базових моделей машинного навчання, так і для мультимодальної моделі глибинного навчання.

3.7 Реалізація підготовки даних для PyTorch

Підготовка даних для моделей глибинного навчання реалізована у файлі 06_dataset.py. Цей модуль є проміжною ланкою між попередньо обробленими таблицями даних і нейронними моделями, реалізованими у PyTorch. Його основне призначення полягає в тому, щоб перетворити біологічні та табличні дані у формат тензорів, придатний для подальшого передавання в модель під час навчання, валідації та тестового оцінювання.

На попередніх етапах пайплайну дані зберігаються у вигляді таблиць Parquet. Кожен запис містить повну послідовність мРНК, межі функціональних ділянок, цільове значення target і набір числових ознак. Однак нейронна мережа не може безпосередньо працювати з рядком символів A, C, G, T як із текстовим значенням. Тому перед навчанням послідовність необхідно закодувати у числовому вигляді, сформувати тензори однакового формату для кожного прикладу та забезпечити коректне об’єднання прикладів у mini-batch.

У модулі 06_dataset.py реалізовано клас Dataset, який відповідає стандартній логіці роботи з даними у PyTorch. Він зберігає таблицю з підготовленими записами, таблицю числових ознак і список назв ознак, що використовуються моделлю. Під час звернення до окремого індексу Dataset повертає всі дані, необхідні для одного прикладу: закодовану нуклеотидну послідовність, позиційне представлення функціональних ділянок, вектор числових ознак і цільове значення.

Кодування нуклеотидів виконується шляхом заміни кожного символу послідовності на ціле число. Такий підхід дозволяє передавати послідовність до embedding-шару нейронної мережі. У програмній реалізації кожному нуклеотиду відповідає окремий індекс. Символи A, C, G і T кодуються різними значеннями, а службові або невизначені позиції обробляються окремо. Це дозволяє моделі навчатися власних векторних

представлень нуклеотидів замість використання ручного one-hot кодування на вході.

Окремо формується позиційне представлення функціональних ділянок мРНК. Для кожної позиції послідовності визначається, до якої ділянки вона належить: 5′UTR, CDS або 3′UTR. Такий трек дає моделі інформацію не лише про сам нуклеотид, а й про його функціональний контекст. Це важливо, оскільки однаковий локальний мотив може мати різне біологічне значення залежно від того, чи знаходиться він у 5′UTR, кодуючій ділянки або 3′UTR.

Формування region labels виконується на основі меж cds_start і cds_end, визначених на етапі попередньої обробки. Позиції до cds_start позначаються як 5′UTR, позиції від cds_start до cds_end – як CDS, а позиції після cds_end – як 3′UTR. У результаті для кожної послідовності створюється числовий вектор такої самої довжини, як і сама послідовність. Цей вектор передається до моделі разом із нуклеотидним рядом.

Крім послідовнісних даних, Dataset повертає вектор структурно-статистичних ознак. Цей вектор формується на основі таблиць, створених скриптом 04_features.py. Він містить числові характеристики транскрипту, зокрема довжини функціональних ділянок, GC-вміст, k-мерні частоти, ознаки splice-треку та інші розраховані характеристики. У мультимодальній моделі цей вектор подається до окремої MLP-гілки, тоді як нуклеотидна послідовність і region labels обробляються послідовнісною гілкою.

Цільове значення target також перетворюється у числовий тензор. Оскільки задача є регресійною, target подається як дійсне число, а не як мітка класу. Це важливо для подальшого використання регресійних функцій втрат і метрик. На рівні Dataset кожен приклад містить одне значення target, що відповідає ефективності трансляції відповідного транскрипту.

Однією з основних складностей підготовки даних є різна довжина послідовностей мРНК. У звичайному випадку mini-batch у PyTorch формується шляхом об’єднання тензорів однакового розміру. Однак повні

мРНК можуть мати суттєво різну довжину, тому їх не можна безпосередньо об’єднати в один batch без додаткової обробки. Для вирішення цієї задачі у програмній реалізації використано динамічне padding-доповнення.

Динамічне padding-доповнення виконується у спеціальній функції collate_fn. Ця функція отримує список окремих прикладів, які мають увійти до одного mini-batch, визначає максимальну довжину послідовності в межах цього batch і доповнює коротші послідовності службовими значеннями до цієї довжини. У PyTorch параметр collate_fn використовується саме для визначення способу об’єднання окремих samples у batch [25].

Перевага динамічного padding полягає в тому, що послідовності не доповнюються до максимальної довжини в усьому наборі даних. Замість цього довжина padding визначається окремо для кожного mini-batch. Це зменшує кількість штучно доданих позицій і робить навчання ефективнішим з погляду використання пам’яті. Для довгих мРНК такий підхід є особливо важливим, оскільки глобальне доповнення всіх записів до максимальної довжини призвело б до значного збільшення розміру тензорів.

Разом із доповненням послідовностей формується маска реальних позицій. Вона вказує, які елементи тензора відповідають справжнім нуклеотидам, а які були додані лише для вирівнювання довжин у batch. Така маска використовується моделлю для того, щоб не враховувати padding-позиції під час pooling або інших операцій агрегації. Без такої маски модель могла б сприймати службові значення як частину біологічної послідовності, що призвело б до спотворення представлення.

Аналогічне доповнення виконується і для region labels. Оскільки region labels мають таку саму довжину, як і нуклеотидна послідовність, вони також повинні бути вирівняні в межах batch. Для padding-позицій використовується окреме службове значення, яке не відповідає жодній реальній функціональній ділянці мРНК. Це дозволяє моделі відрізняти реальні ділянці 5′UTR, CDS і 3′UTR від штучно доданих позицій.

У програмній реалізації також передбачено режим full-sequence. Його призначення полягає в тому, щоб працювати з повною послідовністю мРНК, а не лише з її початковим фрагментом. Це важливо для даної роботи, оскільки тема передбачає прогнозування ефективності трансляції саме за повною послідовністю. Якщо модель використовувала б лише обмежений фрагмент, наприклад перші кілька тисяч нуклеотидів, частина 3′UTR або довгої CDS могла б бути втрачена. У full-sequence режимі така втрата інформації мінімізується.

Разом із full-sequence режимом може використовуватися групування за довжиною, тобто bucket by length. Його ідея полягає в тому, щоб приклади близької довжини частіше потрапляли в один mini-batch. Це дозволяє зменшити кількість padding-позицій, оскільки всередині batch послідовності мають більш подібні довжини. Такий підхід не змінює самі дані, але робить навчання ефективнішим і зменшує зайве використання пам’яті.

Важливо, що Dataset і collate_fn не виконують навчання моделі та не змінюють параметри нейронної мережі. Їхня задача полягає лише у підготовці даних у потрібному форматі. Завдяки такому розділенню логіки код навчання може працювати з уже готовими batch-об’єктами і не містити повторної логіки кодування послідовностей, доповнення або формування масок.

У результаті один mini-batch для мультимодальної моделі містить кілька типів даних. До нього входять тензор закодованих нуклеотидних послідовностей, тензор region labels, маска реальних позицій, матриця числових ознак і вектор цільових значень. Такий формат відповідає мультимодальній природі запропонованого методу: одна частина моделі отримує послідовнісні дані, інша – структурно-статистичні ознаки, а регресійна голова прогнозує числове значення translation efficiency.

Таким чином, файл 06_dataset.py забезпечує коректний перехід від біологічного табличного представлення даних до тензорного представлення, необхідного для PyTorch. Реалізовані в ньому механізми

кодування нуклеотидів, формування region labels, підключення числових ознак, dynamic padding і valid mask є необхідними для подальшого навчання sequence-only та мультимодальної моделей. Саме цей етап дозволяє використовувати повні мРНК різної довжини без втрати основної структурної інформації.

3.8 Реалізація архітектур моделей глибинного навчання

Архітектури моделей глибинного навчання реалізовано у файлі 07_models.py. Цей модуль містить програмний опис нейронних мереж, які використовуються для прогнозування ефективності трансляції мРНК. У межах роботи було реалізовано дві основні архітектури: SequenceOnlyCNN та MultimodalTEModel. Перша модель використовує лише нуклеотидну послідовність, а друга поєднує послідовність, позиційну інформацію про функціональні ділянки мРНК та числові структурно-статистичні ознаки.

Розділення моделей на sequence-only та мультимодальну архітектуру є важливим з погляду програмної реалізації та подальшого експериментального аналізу. SequenceOnlyCNN використовується як нейромережевий базовий варіант, який дозволяє оцінити можливості згорткової моделі, що працює тільки з повною нуклеотидною послідовністю. MultimodalTEModel реалізує основну ідею запропонованого методу, оскільки використовує декілька джерел інформації про транскрипт.

Усі моделі реалізовано як класи, що наслідують базовий клас PyTorch nn.Module. Такий підхід відповідає стандартній логіці побудови моделей у PyTorch: у конструкторі класу визначаються шари моделі, а у методі forward описується порядок проходження вхідних даних через ці шари. Завдяки цьому архітектури можна використовувати в єдиному циклі навчання, незалежно від того, чи модель працює лише з послідовністю, чи з кількома модальностями.

Першою реалізованою архітектурою є SequenceOnlyCNN. Вхідними даними для цієї моделі є закодована повна послідовність мРНК. Кожен нуклеотид попередньо перетворюється на ціле число, після чого послідовність подається до embedding-шару. Embedding-шар перетворює дискретні індекси нуклеотидів у щільні векторні представлення. Це дозволяє моделі не працювати безпосередньо з умовними номерами символів, а навчати власні векторні подання нуклеотидів під час оптимізації.

Після embedding-шару дані передаються до послідовності одномірних згорткових шарів. Використання Conv1D є доцільним для аналізу нуклеотидних послідовностей, оскільки такі шари можуть виділяти локальні патерни в рядку символів. У контексті мРНК такими локальними патернами можуть бути короткі мотиви, ділянки з певним нуклеотидним складом або фрагменти, пов’язані з регуляцією трансляції. Згорткові шари дозволяють моделі автоматично формувати ознаки з послідовності без ручного задання всіх можливих мотивів.

Після кожного згорткового шару застосовується нелінійна функція активації. У реалізації використовується ReLU, яка дозволяє моделі навчати нелінійні залежності між локальними ознаками послідовності та цільовим значенням. Також у моделі може використовуватися dropout, який зменшує ризик перенавчання шляхом випадкового вимикання частини нейронів під час навчання. Наявність dropout є особливо важливою для моделей глибинного навчання, оскільки кількість параметрів у них значно більша, ніж у класичних регресійних моделей.

Оскільки вхідні послідовності мають різну довжину, після згорткової частини необхідно отримати вектор фіксованого розміру для кожного транскрипту. Для цього використовується операція pooling. У програмній реалізації застосовано masked pooling, який враховує лише реальні позиції послідовності та ігнорує padding. Це важливо, оскільки під час формування mini-batch коротші послідовності доповнюються службовими значеннями.

Якщо не використовувати маску, модель могла б помилково включати padding-позиції в узагальнене представлення транскрипту.

Після pooling отримується фіксований вектор ознак, який передається до регресійної голови. Регресійна голова реалізована як невелика багатошарова повнозв’язна мережа. Її задача полягає в тому, щоб перетворити вектор, отриманий зі згорткової частини, в одне числове значення. Це значення є прогнозом ефективності трансляції для відповідного транскрипту. Оскільки задача є регресійною, на виході моделі використовується один нейрон без softmax.

Другою та основною архітектурою є MultimodalTEModel. Вона розширює sequence-only підхід за рахунок використання додаткових модальностей. У цій моделі послідовність мРНК не розглядається ізольовано. До неї додається позиційна інформація про те, до якої функціональної ділянки належить кожна позиція, а також числові структурно-статистичні ознаки, сформовані на попередньому етапі пайплайну.

Послідовнісна частина мультимодальної моделі починається з nucleotide embedding. Як і в SequenceOnlyCNN, нуклеотиди перетворюються у щільні векторні представлення. Паралельно використовується region embedding, який кодує функціональний контекст кожної позиції. Region labels позначають належність позиції до 5′UTR, CDS або 3′UTR. Завдяки цьому модель отримує не лише інформацію про те, який нуклеотид знаходиться в певній позиції, а й інформацію про те, у якій частині мРНК він розташований.

Поєднання nucleotide embedding і region embedding дозволяє сформувати розширене позиційне представлення послідовності. У такому представленні однаковий нуклеотид може мати різний контекст залежно від функціональної ділянки. Це важливо для задачі прогнозування translation efficiency, оскільки регуляторне значення однакового мотиву може відрізнятися в 5′UTR, CDS і 3′UTR. Наприклад, триплет ATG у 5′UTR може

бути пов’язаний з upstream старт-кодоном, тоді як у CDS він є звичайним кодоном метіоніну.

Після об’єднання нуклеотидного та регіонального представлень дані передаються до згорткової гілки. Ця гілка за своєю логікою подібна до sequence-only моделі: вона містить одномірні згорткові шари, нелінійні активації, dropout і masked pooling. Однак її вхід є інформативнішим, оскільки містить не тільки символи послідовності, а й позиційний контекст. У результаті згорткова гілка формує приховане представлення мРНК, яке враховує локальні патерни та їхнє розташування у функціональній структурі транскрипту.

Окремою частиною MultimodalTEModel є гілка числових ознак. Її вхідними даними є вектор структурно-статистичних характеристик, сформований у 04_features.py. Цей вектор містить узагальнену інформацію про транскрипт: довжини функціональних ділянок, GC-вміст, k-мерні частоти, upstream ATG, splice-ознаки та інші числові характеристики. Для обробки цього вектора використовується MLP-гілка, яка складається з повнозв’язних шарів, функцій активації та dropout.

Призначення MLP-гілки полягає в тому, щоб перетворити ручно сформовані числові ознаки у приховане представлення, сумісне з представленням послідовнісної гілки. Це дозволяє моделі навчитися використовувати не тільки локальні патерни в нуклеотидному ряді, а й глобальні характеристики транскрипту. Наприклад, послідовнісна гілка може виділяти мотиви та локальні залежності, тоді як числова гілка надає інформацію про довжину ділянок, загальний склад послідовності та інші агреговані властивості.

Після обробки окремих модальностей їхні приховані представлення об’єднуються. У програмній реалізації це виконується шляхом конкатенації векторів, отриманих із послідовнісної та числової гілок. Об’єднаний вектор містить інформацію з різних джерел і передається до фінальної регресійної

голови. Саме на цьому етапі модель навчається комбінувати послідовнісні, позиційні та структурно-статистичні сигнали для прогнозування target.

Регресійна голова мультимодальної моделі має один вихідний нейрон. Це відповідає постановці задачі як регресії, де необхідно передбачити одне неперервне числове значення ефективності трансляції. У моделі не використовується softmax або інша функція, характерна для класифікації, оскільки класи в цій задачі не визначаються. Вихід моделі безпосередньо порівнюється з фактичним значенням target за допомогою регресійної функції втрат.

У реалізації моделей також враховано роботу з padding. Оскільки послідовності в межах mini-batch доповнюються до однакової довжини, згорткові шари отримують тензори, що містять як реальні позиції, так і службові padding-позиції. Для запобігання впливу padding на фінальне представлення використовується masked pooling. Перед pooling службові позиції маскуються, тому агрегування виконується лише за реальними нуклеотидами. Це підвищує коректність роботи моделі з послідовностями різної довжини.

Ще однією особливістю реалізації є параметризованість архітектури. Основні характеристики моделей задаються через конфігураційний файл або через параметри підбору гіперпараметрів. До таких характеристик належать розмір nucleotide embedding, розмір region embedding, кількість каналів у згорткових шарах, розмір прихованого шару для числових ознак і значення dropout. Такий підхід дозволяє запускати різні конфігурації моделі без переписування коду 07_models.py.

Винесення архітектур у окремий файл має практичне значення для підтримки проєкту. Скрипти навчання та оцінювання можуть імпортувати потрібну модель і створювати її відповідно до параметрів конфігурації. Якщо потрібно змінити архітектуру, наприклад кількість згорткових шарів або спосіб об’єднання модальностей, ці зміни можна внести в одному місці,

не змінюючи логіку завантаження даних, формування batch або обчислення метрик.

Реалізовані архітектури відповідають загальній логіці запропонованого методу. SequenceOnlyCNN дозволяє перевірити послідовнісну складову, а MultimodalTEModel реалізує повне мультимодальне прогнозування. Така структура коду забезпечує можливість порівняння різних підходів у межах одного пайплайну, але сам аналіз результатів цього порівняння виконується вже в експериментальному розділі роботи.

Таким чином, файл 07_models.py містить програмну реалізацію нейронних архітектур, необхідних для прогнозування ефективності трансляції мРНК. SequenceOnlyCNN обробляє повну нуклеотидну послідовність за допомогою embedding-шару, Conv1D-блоків, masked pooling і регресійної голови. MultimodalTEModel розширює цей підхід шляхом додавання region embedding і гілки числових структурно-статистичних ознак. У результаті модель може використовувати як локальні патерни повної послідовності, так і додаткові ознаки, що описують функціональну та структурну організацію транскрипту.

3.9 Реалізація навчання, підбору гіперпараметрів та збереження моделей

Навчання моделей глибинного навчання реалізовано у скрипті 08_train.py, а підбір гіперпараметрів мультимодальної моделі – у скрипті 13_tune_deep.py. Таке розділення дозволяє відокремити стандартне навчання моделі за заданою конфігурацією від експериментального пошуку кращих параметрів. У результаті основний пайплайн залишається зрозумілим і відтворюваним, а підбір гіперпараметрів може запускатися окремо, без зміни логіки попередньої обробки даних або архітектури моделей.

Вхідними даними для навчання є підготовлені вибірки, створені на етапі попередньої обробки, а також таблиці структурно-статистичних ознак, сформовані скриптом 04_features.py. Скрипт навчання завантажує навчальну та валідаційну вибірки, створює об’єкти Dataset і DataLoader, ініціалізує модель відповідно до параметрів конфігурації, задає функцію втрат, оптимізатор і додаткові механізми контролю навчання. Після цього виконується послідовне навчання моделі по епохах.

На початку роботи скрипт визначає обчислювальний пристрій. Якщо доступна відеокарта з підтримкою CUDA, модель і тензори переносяться на GPU, що пришвидшує навчання. Якщо GPU недоступний, навчання виконується на CPU. Такий підхід робить код переносимим між різними середовищами: його можна запускати як на локальному комп’ютері з відеокартою, так і в середовищі без GPU, хоча час навчання в цих випадках буде відрізнятися.

Після вибору пристрою виконується завантаження конфігурації. У config.yaml задаються основні параметри навчання: розмір batch, learning rate, кількість епох, patience для ранньої зупинки, dropout, розміри embedding-шарів, кількість каналів у згорткових шарах, параметри оптимізатора та інші налаштування. Завдяки цьому навчальний скрипт не містить жорстко зафіксованих значень і може використовуватися для запуску різних конфігурацій без редагування коду.

Далі створюються DataLoader для навчальної та валідаційної вибірок. Вони відповідають за формування mini-batch під час навчання. Для послідовностей різної довжини використовується спеціальна функція collate_fn, описана в попередньому підрозділі. Вона виконує динамічне padding-доповнення та формує маску реальних позицій. Завдяки цьому модель може працювати з повними послідовностями мРНК різної довжини.

У процесі навчання для кожного batch виконується стандартна послідовність операцій глибинного навчання. Спочатку вхідні дані передаються до моделі, яка формує прогнозовані значення target. Потім

обчислюється функція втрат між прогнозом і фактичними значеннями. Після цього виконується зворотне поширення помилки, обчислюються градієнти параметрів моделі та оптимізатор оновлює ваги. Такий цикл повторюється для всіх batch навчальної вибірки.

У програмній реалізації передбачено використання різних функцій втрат. Для задачі регресії базовим варіантом є середньоквадратична помилка, тобто MSELoss. Вона сильно штрафує великі помилки прогнозування, оскільки похибка підноситься до квадрата. Також у проєкті використовується Smooth L1 loss. У документації PyTorch ця функція описується як критерій, що використовує квадратичний член для малих абсолютних помилок і L1-член для більших помилок, тому вона може бути менш чутливою до окремих великих відхилень [26].

Для оптимізації параметрів використовується Adam або AdamW залежно від конфігурації. Adam є адаптивним методом стохастичної оптимізації, який використовує оцінки першого та другого моментів градієнта [19]. AdamW є модифікацією Adam із відокремленим weight decay, що часто використовується під час навчання нейронних мереж. У межах програмного пайплайну вибір оптимізатора задається в конфігурації або у параметрах підбору гіперпараметрів.

Під час навчання також може застосовуватися gradient clipping. Його задача полягає в обмеженні величини градієнтів. Це корисно для стабілізації навчання, особливо в умовах, коли модель працює з довгими послідовностями і може виникати ризик надто великих оновлень параметрів. У програмній реалізації значення максимальної норми градієнта задається як параметр конфігурації. Якщо gradient clipping увімкнено, перед кроком оптимізатора градієнти приводяться до допустимого діапазону.

Для керування learning rate у процесі навчання може використовуватися scheduler. Його призначення полягає в тому, щоб змінювати швидкість навчання залежно від перебігу оптимізації. Зокрема, ReduceLROnPlateau зменшує learning rate, якщо контрольована метрика

перестає покращуватися протягом заданої кількості епох [27]. Такий механізм дозволяє робити більші кроки на початку навчання і менші кроки на пізніших етапах, коли модель наближається до локального мінімуму.

Після кожної епохи виконується оцінювання моделі на валідаційній вибірці. На цьому етапі модель переводиться в режим evaluation, а обчислення градієнтів вимикається. Це потрібно для того, щоб оцінювання не впливало на параметри моделі та виконувалося швидше. Для валідаційної вибірки обчислюються регресійні метрики, які використовуються для контролю якості навчання і вибору найкращого checkpoint.

У скрипті навчання передбачено збереження найкращої моделі. Після кожної епохи поточне значення валідаційної метрики порівнюється з найкращим попереднім значенням. Якщо модель покращила результат, її параметри зберігаються як best checkpoint. Такий підхід дозволяє не використовувати обов’язково останню епоху навчання, оскільки наприкінці навчання модель може почати перенавчатися або погіршувати якість на валідаційній вибірці.

Механізм ранньої зупинки використовується для припинення навчання у випадку, якщо якість на валідаційній вибірці не покращується протягом заданої кількості епох. Параметр patience визначає, скільки епох модель може навчатися без покращення. Якщо протягом цього періоду покращення не відбувається, навчання зупиняється. Це дозволяє зменшити час експериментів і знизити ризик перенавчання.

Паралельно з checkpoint моделі зберігається історія навчання. Вона містить значення функції втрат і метрик для навчальної та валідаційної вибірок по епохах. Історія зберігається у вигляді CSV-файлу, що дозволяє надалі будувати графіки навчання, аналізувати стабільність оптимізації та перевіряти, чи не виникало різкого перенавчання. Графіки навчання зберігаються у каталозі results/figures.

Окрему частину програмної реалізації становить підбір гіперпараметрів у скрипті 13_tune_deep.py. Його задача полягає в

автоматизованому запуску декількох конфігурацій мультимодальної моделі. Для кожної конфігурації створюється модель з відповідними параметрами, виконується навчання на навчальній вибірці, оцінювання на валідаційній вибірці та збереження результатів у таблицю. Після завершення всіх запусків вибирається конфігурація, яка показала найкраще значення обраної валідаційної метрики.

У межах підбору гіперпараметрів змінюються параметри як архітектури, так і процесу навчання. До архітектурних параметрів належать розмір nucleotide embedding, розмір region embedding, кількість і розмір згорткових каналів, розмір прихованого шару для числової гілки та dropout. До параметрів навчання належать learning rate, weight decay, batch size, тип функції втрат, параметри scheduler і gradient clipping. Такий набір параметрів дозволяє перевірити різні варіанти складності моделі та режимів оптимізації.

Під час підбору гіперпараметрів тестова вибірка не використовується. Вона залишається відкладеною для фінального оцінювання вже після вибору конфігурації. Це важливо для коректності експерименту, оскільки використання тестових даних під час підбору параметрів призвело б до витоку інформації та завищення оцінки якості. Тому скрипт 13_tune_deep.py працює лише з навчальною та валідаційною вибірками.

Результати підбору гіперпараметрів зберігаються у каталозі results/metrics. Для кожного запуску фіксуються параметри конфігурації, статус завершення, найкраща епоха, значення валідаційних метрик і шлях до checkpoint. Окремо зберігається файл із найкращою конфігурацією, який може бути використаний для повторного запуску фінальної моделі або для документування обраних параметрів у роботі.

Після завершення підбору найкраща модель зберігається у каталозі results/models. У проєкті передбачено збереження checkpoint під стандартним ім’ям best_model.pt, а також можливість зафіксувати фінальну мультимодальну модель окремим файлом. Такий підхід дозволяє чітко

відокремити робочі checkpoint різних запусків від фінальної моделі, яка використовується для тестового оцінювання та подальшого аналізу.

Checkpoint моделі містить не лише параметри нейронної мережі, а й службову інформацію, необхідну для повторного завантаження моделі. До такої інформації належать конфігурація, назви числових ознак, параметри архітектури, значення метрик і номер епохи. Завдяки цьому модель можна відновити без ручного відтворення всіх параметрів. Це підвищує відтворюваність експерименту і зменшує ризик помилок під час фінального оцінювання.

Збереження моделей і результатів організовано таким чином, щоб усі артефакти експерименту були доступні після завершення запуску. Навчальні історії, таблиці підбору гіперпараметрів, checkpoint моделей і конфігураційні файли зберігаються у відповідних підкаталогах results. Це дозволяє перевірити, за яких умов була навчена модель, які параметри використовувалися і який checkpoint був обраний для фінального оцінювання.

Таким чином, скрипти 08_train.py і 13_tune_deep.py реалізують повний цикл навчання та налаштування моделей глибинного навчання. Перший скрипт забезпечує навчання моделі за заданою конфігурацією, контроль якості на валідаційній вибірці, ранню зупинку та збереження найкращого checkpoint. Другий скрипт автоматизує підбір гіперпараметрів мультимодальної моделі та фіксує найкращу конфігурацію. Завдяки цьому програмна реалізація підтримує як звичайний запуск навчання, так і систематичний пошук параметрів для фінальної моделі.

3.10 Реалізація оцінювання та збереження артефактів

Фінальне оцінювання моделей і збереження артефактів реалізовано у скриптах 09_evaluate.py, 10_analyze_results.py та 11_predict.py. Ці модулі відповідають за завершальний етап програмного пайплайну, на якому

навчена модель завантажується з checkpoint, застосовується до тестової вибірки або окремих прикладів, а отримані результати зберігаються у вигляді таблиць, графіків і файлів прогнозів. Основна задача цього етапу полягає в тому, щоб відокремити процедуру навчання від процедури незалежного оцінювання.

Скрипт 09_evaluate.py використовується для фінального оцінювання моделі на тестовій вибірці. На відміну від навчального скрипта, він не оновлює параметри моделі та не бере участі у виборі гіперпараметрів. Його задача полягає в тому, щоб завантажити вже навчений checkpoint, відновити архітектуру моделі з відповідними параметрами, застосувати модель до тестових даних і зберегти отримані прогнози та метрики. Такий поділ дозволяє уникнути змішування навчання й тестового оцінювання.

Першим кроком оцінювання є завантаження checkpoint моделі з каталогу results/models. Checkpoint містить збережені параметри нейронної мережі, конфігурацію моделі, назви числових ознак і службову інформацію, необхідну для повторного створення архітектури. У PyTorch типовий підхід до збереження та завантаження моделей полягає в роботі зі state_dict, тобто словником параметрів моделі, який можна зберегти та пізніше завантажити в екземпляр тієї самої архітектури [28]. У програмній реалізації цей підхід використовується для відновлення фінальної моделі перед тестовим оцінюванням.

Після завантаження checkpoint створюється модель відповідного типу. Якщо у checkpoint збережено параметри мультимодальної архітектури, скрипт створює MultimodalTEModel з тими самими параметрами embedding-шарів, згорткових каналів, числової гілки та dropout. Це потрібно для того, щоб структура моделі під час оцінювання збігалася зі структурою, яка використовувалася під час навчання. Після створення моделі в неї завантажуються навчені ваги, а сама модель переводиться на доступний обчислювальний пристрій.

Перед запуском прогнозування модель переводиться в режим оцінювання. Це потрібно для коректної роботи шарів, поведінка яких відрізняється під час навчання та тестування, зокрема dropout. Під час оцінювання також вимикається обчислення градієнтів, оскільки параметри моделі вже не оновлюються. Такий режим зменшує витрати пам’яті й пришвидшує виконання прогнозування.

Для тестової вибірки створюється Dataset і DataLoader за тією самою логікою, що й під час навчання. Це забезпечує однакове кодування нуклеотидних послідовностей, формування region labels, підключення числових ознак, динамічне padding-доповнення та створення масок реальних позицій. Завдяки цьому модель отримує тестові дані в такому самому форматі, у якому вона отримувала дані під час навчання та валідації.

Під час проходження тестової вибірки скрипт збирає фактичні значення target і прогнозовані значення моделі. Для кожного batch модель формує прогноз, після чого ці прогнози переносяться у загальний список результатів. Після завершення проходження всіх batch формується повна таблиця прогнозів для тестової вибірки. Ця таблиця зберігається у каталозі results/predictions і може містити ідентифікатори транскриптів, фактичні значення target, прогнозовані значення та похибку для кожного запису.

Окремо в 09_evaluate.py реалізовано обчислення регресійних метрик. Для оцінювання якості прогнозування використовуються MAE, RMSE, коефіцієнт кореляції Пірсона, коефіцієнт кореляції Спірмена та, за потреби, R^2. MAE і RMSE характеризують абсолютну величину похибки прогнозування, тоді як кореляційні метрики показують, наскільки добре модель відтворює відносний порядок і лінійну залежність між фактичними та прогнозованими значеннями. Самі числові значення цих метрик аналізуються в експериментальному розділі.

Результати оцінювання зберігаються у файлі deep_learning_metrics.csv у каталозі results/metrics. Такий формат зручний для подальшого використання в таблицях кваліфікаційної роботи, порівняння з базовими

моделями та повторної перевірки результатів. Оскільки файл з метриками зберігається окремо від текстового опису, його можна використовувати як формалізований артефакт експерименту.

Крім таблиці метрик, скрипт оцінювання зберігає прогнози моделі. Файл deep_predictions_test.csv містить результати роботи моделі для тестової вибірки. Наявність такого файлу є важливою, оскільки він дозволяє не лише подати загальні метрики, а й виконати подальший аналіз окремих прикладів. Наприклад, на його основі можна визначити записи з найбільшою похибкою, побудувати графік прогнозованих і фактичних значень або дослідити залежність похибки від довжини послідовності.

Для візуального аналізу результатів створюються графіки. До них належать графік порівняння фактичних і прогнозованих значень, розподіл залишків, розподіл цільової змінної та, за наявності збереженої історії навчання, крива навчання. Ці графіки зберігаються у каталозі results/figures. Вони не змінюють результат оцінювання, але допомагають інтерпретувати поведінку моделі та виявити можливі систематичні помилки.

Скрипт 10_analyze_results.py призначений для додаткового аналізу вже отриманих результатів. Він використовує збережені прогнози, метрики та допоміжні таблиці, сформовані на попередніх етапах. Його задача полягає в тому, щоб побудувати додаткові зведення та графіки, які можуть бути використані в експериментальному розділі. Наприклад, такий аналіз може включати дослідження залишків, порівняння моделей, аналіз похибок для різних груп транскриптів або побудову підсумкової таблиці результатів.

Окремим результатом роботи пайплайну є фінальна порівняльна таблиця моделей. Вона формується на основі метрик базових моделей і метрик моделі глибинного навчання. Така таблиця зберігається у файлі final_model_comparison.csv. Її призначення полягає в тому, щоб об’єднати результати різних підходів в одному форматі. У третьому розділі важливо зафіксувати лише сам механізм формування такої таблиці, тоді як її зміст і висновки на основі порівняння розглядаються в четвертому розділі.

Скрипт 11_predict.py реалізує застосування навченої моделі до окремих прикладів або підготовлених вхідних даних. Якщо 09_evaluate.py орієнтований на тестову вибірку і формальне оцінювання якості, то 11_predict.py демонструє, як збережену модель можна використовувати після завершення навчання. Такий скрипт є корисним для практичного застосування, оскільки дозволяє отримати прогноз translation efficiency для конкретного транскрипту без повторного запуску всього процесу навчання.

Для отримання прогнозу окремий приклад має пройти ті самі етапи підготовки, що й записи навчального набору. Послідовність очищується та кодується, для неї формуються позиційні ділянок, обчислюються числові ознаки, після чого дані передаються до моделі. Це забезпечує узгодженість між режимом навчання та режимом використання моделі. Якщо під час прогнозування пропустити певний етап підготовки, вхідні дані можуть не відповідати формату, на якому модель навчалася.

Для наочного представлення практичного застосування навченої моделі було розроблено демонстраційний інтерфейс, який дає змогу вибрати запис із набору даних або ввести повну послідовність мРНК вручну та отримати результат прогнозування. Основні етапи роботи демонстраційного інтерфейсу наведено в додатку А.

У межах проєкту також зберігаються допоміжні артефакти, які не є безпосередніми прогнозами, але потрібні для відтворюваності. До них належать конфігураційні файли, історії навчання, таблиці підбору гіперпараметрів, списки використаних числових ознак, проміжні звіти про якість даних і файли з описом очищення. Збереження таких артефактів дозволяє простежити, як саме було отримано фінальну модель і які параметри використовувалися на кожному етапі.

Важливо, що всі результати зберігаються у заздалегідь визначених каталогах. Метрики зберігаються у results/metrics, графіки – у results/figures, прогнози – у results/predictions, а checkpoint моделей – у results/models. Така організація робить структуру експерименту прозорою та зручною для

перевірки. Крім того, вона зменшує ризик втрати результатів або змішування файлів різних етапів.

Фінальне оцінювання відокремлено від підбору гіперпараметрів. Це означає, що тестова вибірка використовується лише після того, як модель і її параметри вже обрані за результатами навчання та валідації. Такий підхід є необхідним для коректного оцінювання узагальнювальної здатності моделі. Якщо тестові дані використовувати під час вибору конфігурації, оцінка якості може стати необ’єктивною.

Таким чином, реалізація оцінювання та збереження артефактів завершує програмний пайплайн. Скрипт 09_evaluate.py забезпечує формальне тестове оцінювання фінальної моделі, 10_analyze_results.py використовується для додаткового аналізу збережених результатів, а 11_predict.py демонструє можливість застосування моделі до окремих прикладів. Разом ці компоненти забезпечують не лише отримання числових метрик, а й збереження повного набору артефактів, необхідних для подальшого аналізу, документування та відтворення експерименту.

4 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

4.1 Характеристика вихідного набору даних

Експериментальне дослідження було виконано на наборі даних translation-efficiency-human з колекції mRNABench. Цей набір містить інформацію про людські транскрипти та числове значення ефективності трансляції, що використовується як цільова змінна в задачі регресії. На відміну від наборів, у яких розглядається лише 5′UTR, у даній роботі використовується повна зріла послідовність мРНК разом із позиційними треками CDS і splice. Це відповідає поставленій задачі прогнозування ефективності трансляції за повною послідовністю мРНК.

Первинний аудит набору даних було виконано за допомогою скрипта 02_inspect_data.py. У результаті було перевірено структуру таблиці, наявність очікуваних стовпців, пропущені значення, дублікати, тип цільової змінної, довжини послідовностей, коректність позиційних треків, а також можливість відновлення функціональних ділянок 5′UTR, CDS і 3′UTR. Загальну характеристику вихідного набору даних наведено в таблиці 4.1.

Таблиця 4.1 – Загальна характеристика вихідного набору даних

Характеристика	Значення
Кількість записів	10281
Кількість стовпців	7
Кількість унікальних transcript_id	10281
Кількість унікальних gene	10280
Повні дублікати рядків	0
Дублікати за transcript_id	0
Дублікати за sequence	0

Продовження таблиці 4.1

Характеристика	Значення
Пропущені значення	0
Коректні CDS-треки	10275 / 10281
Коректні splice-треки	10281 / 10281
Зафіксовані потенційні проблеми якості даних	35

Як видно з таблиці 4.1, вихідний набір даних має цілісну структуру та не містить пропущених значень. Усі значення transcript_id є унікальними, а повні дублікати рядків відсутні. Також не було виявлено дублювання повних послідовностей мРНК. При цьому для gene було зафіксовано два дублікати, що означає наявність одного гена, представленого більш ніж одним записом. Така особливість не є критичною для аналізу, однак вона була врахована під час подальшого групового розбиття даних на навчальну, валідаційну та тестову вибірки.

Вихідна таблиця містить сім основних стовпців: transcript_id, gene, chromosome, sequence, cds, splice і target. Стовпець transcript_id містить ідентифікатор транскрипту, gene – назву відповідного гена, chromosome – хромосому, sequence – повну послідовність мРНК, cds – позиційний трек кодуючої ділянки, splice – позиційний трек сплайсингу, а target – числове значення ефективності трансляції. Таким чином, набір даних містить як послідовнісну, так і позиційну інформацію, необхідну для формування мультимодального представлення.

Цільова змінна target є числовою неперервною величиною. Це підтверджує, що поставлена задача належить до задач регресії, а не класифікації. Статистичні характеристики цільової змінної наведено в таблиці 4.2.

Таблиця 4.2 – Статистичні характеристики цільової змінної target

Показник	Значення
Кількість значень	10281
Середнє значення	0.248828
Стандартне відхилення	0.685981
Мінімальне значення	-2.639472
Перший квартиль	-0.219759
Медіана	0.245307
Третій квартиль	0.703057
Максимальне значення	3.489203

З таблиці 4.2 видно, що target має як від'ємні, так і додатні значення. Це свідчить про те, що ефективність трансляції в наборі подана не у вигляді простого невід'ємного абсолютного показника, а як нормалізована або логарифмічно перетворена величина. Розподіл значень target є важливим для подальшого вибору метрик оцінювання, оскільки модель повинна прогнозувати неперервне числове значення в достатньо широкому діапазоні.

На рисунку 4.1 наведено розподіл значень цільової змінної target.

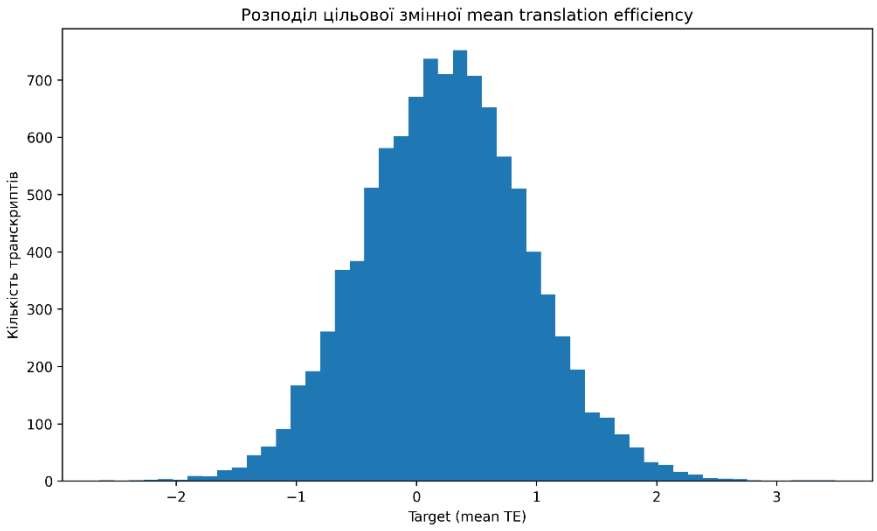


Рисунок 4.1 – Розподіл значень цільової змінної target

Розподіл target показує, що більшість значень зосереджена поблизу центральної ділянки, однак у наборі також присутні транскрипти з помітно нижчими та вищими значеннями ефективності трансляції. Це означає, що модель повинна не лише відтворювати середню тенденцію, а й коректно працювати з прикладами, що мають відхилення від основної маси значень.

Окремо було проаналізовано довжини повних послідовностей мРНК та функціональних ділянок. Оскільки в роботі розглядається повна зріла мРНК, довжина послідовності є важливою характеристикою набору даних. Статистику довжин наведено в таблиці 4.3.

Таблиця 4.3 – Статистика довжин послідовностей та функціональних ділянок мРНК

Показник	Повна послідовність	5'UTR	CDS	3'UTR
Кількість	10281	10275	10275	10275
Середнє значення	3899.09	239.27	1754.15	1907.08
Стандартне відхилення	2339.62	280.35	1342.33	1831.36
Мінімальне значення	303	0	78	0
Перший квартиль	2141	80	879	536.5
Медіана	3391	161	1401	1330
Третій квартиль	5144	300	2194.5	2710.5
Максимальне значення	17497	5140	11088	13708

З таблиці 4.3 видно, що довжини мРНК у наборі суттєво відрізняються між записами. Медіанна довжина повної послідовності становить 3391 нуклеотид, тоді як максимальна довжина досягає 17497 нуклеотидів. Така варіативність підтверджує необхідність спеціальної обробки послідовностей різної довжини під час навчання нейронної мережі. Саме тому в програмній реалізації було використано режим роботи з повними послідовностями та динамічне padding-доповнення у межах mini-batch.

Додатково було встановлено, що 3985 транскриптів, тобто 38,76% набору, мають довжину понад 4096 нуклеотидів. Крім того, 641 транскрипт має довжину понад 8192 нуклеотиди, а 22 транскрипти – понад 12000 нуклеотидів. Це є важливим аргументом проти використання простого фіксованого обрізання послідовності до 4096 нуклеотидів, оскільки в такому випадку значна частина інформації для довгих транскриптів була б втрачена.

На рисунку 4.2 наведено розподіл довжин повних послідовностей мРНК.

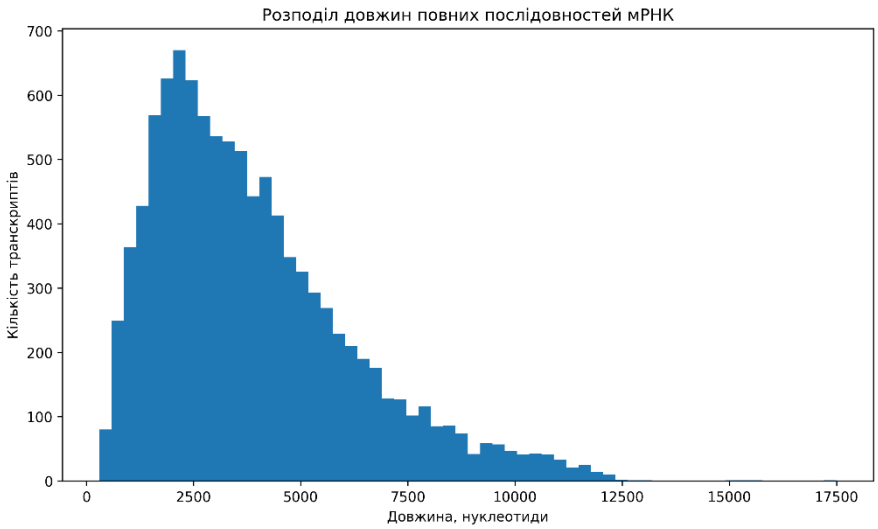


Рисунок 4.2 – Розподіл довжин повних послідовностей мРНК

Розподіл довжин підтверджує, що набір містить як відносно короткі, так і дуже довгі транскрипти. Наявність довгого правого хвоста розподілу означає, що задача обробки повної мРНК потребує більш гнучкого підходу, ніж використання єдиної фіксованої довжини для всіх записів. У даній роботі ця проблема була врахована під час реалізації Dataset і DataLoader для PyTorch.

На рисунку 4.3 наведено розподіл довжин основних функціональних ділянок мРНК.

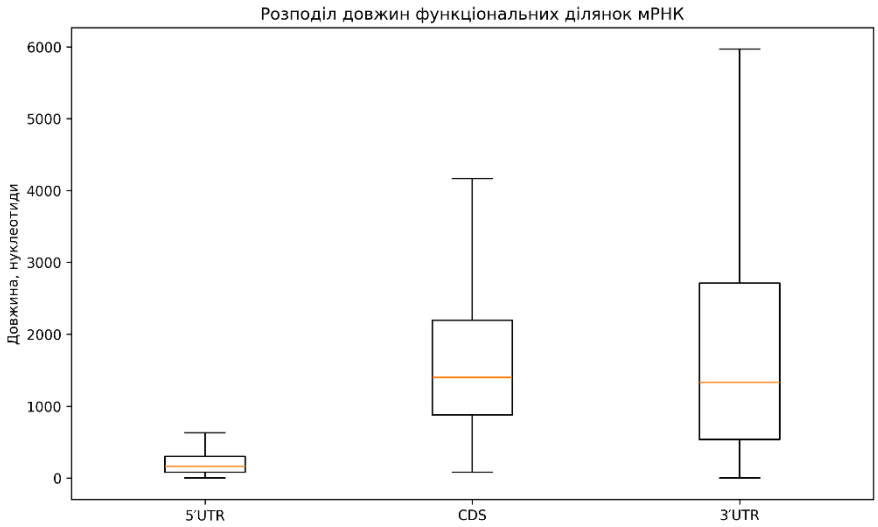


Рисунок 4.3 – Розподіл довжин функціональних ділянок мРНК

З рисунка 4.3 та таблиці 4.3 видно, що найбільшу варіативність мають CDS і 3′UTR. При цьому 5′UTR у більшості випадків є коротшою, але також може досягати значної довжини. Така різнорідність функціональних ділянок обґрунтовує використання не лише повної послідовності, а й позиційної інформації про належність кожного нуклеотиду до 5′UTR, CDS або 3′UTR.

Під час аудиту також було перевірено коректність позиційних треків. Для splice-треку довжина відповідала довжині послідовності в усіх записах. Для CDS-треку було виявлено декілька некоректних випадків, пов’язаних із неможливістю відновлення меж кодуючої ділянки або виходом меж CDS за довжину послідовності. Такі записи були зафіксовані на етапі аудиту і надалі оброблялися на етапі попередньої обробки.

Окремо було проаналізовано стартові та стоп-кодони. Переважна більшість транскриптів із коректним CDS мала канонічний старт-кодон ATG. Водночас було виявлено окремі записи з неканонічними стартовими або стоп-кодонами. У межах аудиту ці випадки не вилучалися автоматично, а були зафіксовані у файлі data_quality_issues.csv як потенційні особливості якості даних. Такий підхід дозволив відокремити етап діагностики від етапу фактичної фільтрації.

Загалом за результатами аудиту було зафіксовано 35 потенційних проблем якості даних. Серед них були записи з неканонічними старт-кодонами, неканонічними стоп-кодонами, некоректними CDS-треками та дублюванням gene. Наявність таких випадків є типовою для реальних біологічних наборів даних і потребує явної фіксації перед подальшим моделюванням. У даній роботі ці проблеми були задокументовані та враховані на наступному етапі попередньої обробки.

Таким чином, первинний аналіз показав, що набір translation-efficiency-human є придатним для розв'язання задачі регресійного прогнозування ефективності трансляції мРНК. Він містить повні послідовності транскриптів, позиційні треки CDS і splice та числову цільову змінну target. Водночас набір має значну варіативність довжин послідовностей і містить окремі проблемні записи, що обґрунтовує необхідність окремого етапу попередньої обробки перед формуванням ознак і навчанням моделей.

4.2 Результати попередньої обробки даних

Після первинного аудиту вихідного набору було виконано попередню обробку даних за допомогою скрипта 03_preprocess.py. Метою цього етапу було перетворення вихідної таблиці у форму, придатну для подальшого формування ознак і навчання моделей. На відміну від етапу аудиту, де проблемні записи лише фіксувалися, під час попередньої обробки виконувалося фактичне очищення даних, виділення функціональних ділянок мРНК та формування навчальної, валідаційної і тестової вибірок.

Основними операціями попередньої обробки були уніфікація запису нуклеотидних послідовностей, перевірка цільової змінної, відновлення меж CDS, виділення 5′UTR, CDS і 3′UTR, фільтрація некоректних записів та групове розбиття даних за gene. Узагальнений результат очищення даних наведено в таблиці 4.4.

Таблиця 4.4 – Результати очищення вихідного набору даних

Показник	Значення
Кількість записів до попередньої обробки	10281
Кількість вилучених записів	6
Кількість записів після попередньої обробки	10275
Основна причина вилучення	Некоректний CDS-трек
Використаний спосіб розбиття	Групове розбиття за gene

Як видно з таблиці 4.4, після попередньої обробки з набору було вилучено лише невелику частину записів. Причиною вилучення були не статистичні відхилення target і не довжина послідовності, а технічна неможливість коректно відновити межі кодуючої ділянки за CDS-треком. Такий підхід є обґрунтованим, оскільки для мультимодального представлення необхідно надійно визначити 5′UTR, CDS і 3′UTR. Якщо межі CDS не можуть бути визначені коректно, запис не може бути використаний для формування позиційної модальності.

На етапі очищення послідовності були приведені до єдиного формату. Усі символи було переведено у верхній регістр, а послідовності інтерпретувалися в алфавіті A, C, G, T і N. Такий формат відповідає технічному запису послідовностей у наборі даних. При цьому з біологічної точки зору символ T відповідає U у молекулі мРНК, що було враховано під час інтерпретації ознак, зокрема upstream ATG як аналога upstream AUG.

Ключовим етапом попередньої обробки було відновлення меж CDS. Оскільки CDS-трек у наборі даних позначає не всю кодуючу ділянку суцільною маскою, а початкові позиції кодонів, для визначення меж було використано спеціальну логіку. Початок CDS визначався як перша активна позиція у CDS-треку, а кінець CDS – як остання активна позиція плюс три нуклеотиди. Додавання трьох нуклеотидів потрібне тому, що остання активна позиція відповідає початку останнього кодона, а сам кодон має довжину три нуклеотиди.

Після визначення меж CDS для кожного коректного запису було сформовано три функціональні ділянки. Ділянка 5′UTR відповідала фрагменту від початку послідовності до першої позиції CDS. Ділянка CDS відповідала фрагменту від cds_start до cds_end. Ділянка 3′UTR відповідала фрагменту після cds_end до кінця повної послідовності. Таке розбиття було необхідне для подальшого формування регіональних ознак, region labels і числових характеристик окремих ділянок мРНК.

Окрім виділення функціональних ділянок, під час попередньої обробки було сформовано службові поля, які використовувалися на наступних етапах пайплайну. До них належать позиції cds_start і cds_end, довжина повної послідовності, довжини 5′UTR, CDS і 3′UTR, стартовий кодон, стоп-кодон та ознаки валідності запису. Ці поля дозволили не виконувати повторне відновлення меж CDS на кожному наступному етапі, а працювати з уже підготовленою таблицею.

Після очищення даних було виконано поділ на навчальну, валідаційну та тестову вибірки. Результати цього розбиття наведено в таблиці 4.5.

Таблиця 4.5 – Результати розбиття даних на навчальну, валідаційну та тестову вибірки

Вибірка	Кількість записів	Кількість gene
Train	7192	7191
Validation	1541	1541
Test	1542	1542
Усього	10275	10274

З таблиці 4.5 видно, що після очищення всі записи були розподілені між трьома вибірками. Навчальна вибірка використовується для оптимізації параметрів моделей, валідаційна – для контролю навчання та підбору гіперпараметрів, а тестова – для фінального оцінювання. Такий поділ відповідає загальній логіці контрольованого машинного навчання, де

фінальна перевірка якості має виконуватися на даних, які не використовувалися під час навчання або вибору конфігурації.

Для зменшення ризику витоку інформації між вибірками було використано групове розбиття за gene. Це означає, що записи, пов’язані з одним і тим самим геном, не повинні одночасно потрапляти до різних частин набору. Такий підхід є важливим, оскільки транскрипти одного гена можуть мати спільні біологічні властивості. Якщо б такі записи опинилися одночасно в train і test, модель могла б отримати завищену оцінку якості через часткову подібність даних.

Групове розбиття було реалізовано з використанням GroupShuffleSplit. Цей інструмент виконує випадкове розбиття з урахуванням груп і дозволяє ділити не окремі записи незалежно один від одного, а групи записів. У межах даної роботи як група використовувалося поле gene. Після формування вибірок було додатково перевірено перетин gene між train, validation і test. Результати цієї перевірки наведено в таблиці 4.6.

Таблиця 4.6 – Перевірка перетину gene між вибірками

Перевірка	Кількість спільних gene
Train / Validation	0
Train / Test	0
Validation / Test	0

Результати таблиці 4.6 підтверджують, що між сформованими вибірками немає перетину за gene. Це означає, що групове розбиття було виконано коректно, а тестова вибірка є незалежною від навчальної за цим критерієм. Така перевірка є важливою для подальшої інтерпретації результатів, оскільки вона знижує ризик завищеної оцінки якості моделі через повторення біологічно близьких записів у різних вибірках.

Важливо, що під час попередньої обробки не виконувалося видалення записів за значенням target або за довжиною послідовності. Довгі транскрипти залишалися в наборі, оскільки метою роботи є прогнозування ефективності трансляції саме за повною послідовністю мРНК. Це дозволило зберегти біологічно значущу інформацію, яка могла б бути втрачена у випадку простого обрізання або фільтрації за довжиною.

Окремо слід зазначити, що записи з неканонічними стартовими або стоп-кодонами не вилучалися автоматично лише на основі цієї ознаки. Такі випадки були зафіксовані під час аудиту як потенційні особливості якості даних, але основним критерієм вилучення на етапі preprocessing була технічна неможливість коректно побудувати структурне представлення мРНК. Це дозволило уникнути надмірної фільтрації та зберегти більшу частину вихідного набору.

У результаті попередньої обробки було отримано очищені та узгоджені дані, придатні для подальшого формування мультимодальних ознак. Кожен запис після цього етапу мав повну послідовність мРНК, визначені межі 5′UTR, CDS і 3′UTR, числове значення target і належність до однієї з трьох вибірок. Це створило основу для наступного етапу – формування структурно-статистичних ознак і підготовки даних до навчання моделей.

4.3 Аналіз сформованих структурно-статистичних ознак

Після попередньої обробки даних було виконано формування структурно-статистичних ознак за допомогою скрипта 04_features.py. Метою цього етапу було створення числового опису кожного транскрипту, який доповнює послідовнісне представлення мРНК. Отримані ознаки використовувалися у двох напрямах: як вхідні дані для базових моделей машинного навчання та як окрема числова модальність у мультимодальній нейронній моделі.

Фінальна матриця числових ознак містила 83 ознаки. До неї увійшли довжинні характеристики, межі CDS, GC-вміст, кількість upstream ATG, частоти 3-мерів, MFE-ознаки та службові контрольні ознаки. Узагальнену структуру сформованих ознак наведено в таблиці 4.7.

Таблиця 4.7 – Групи сформованих структурно-статистичних ознак

Група ознак	Кількість ознак	Приклади ознак
Довжинні ознаки та межі CDS	6	sequence_length, utr5_length, cds_length, utr3_length, cds_start, cds_end
GC-вміст	4	gc_full, gc_5utr, gc_cds, gc_3utr
Upstream ATG	1	upstream_atg_count
Частоти 3-мерів	64	kmer3_ATG, kmer3_TTG, kmer3_CAG
MFE-ознаки	7	mfe_5utr_300, mfe_5utr_300_norm, mfe_start_window_200, mfe_start_window_200_norm, mfe_cds_first_300, mfe_cds_first_300_norm, mfe_available
Службові контрольні ознаки	1	invalid_base_count
Усього	83	-

Як видно з таблиці 4.7, найбільшу частину числового представлення становлять 3-мерні ознаки. Це пояснюється тим, що для алфавіту A, C, G, T існує 64 можливі комбінації триплетів. Такі ознаки дають змогу подати локальний нуклеотидний склад у фіксованому числовому вигляді. Крім них, до моделі було включено ознаки, що описують довжину повної мРНК і функціональних ділянок, регіональний GC-вміст, кількість upstream ATG та локальні MFE-характеристики.

Окремо слід зазначити, що upstream ATG у програмній реалізації відповідає біологічному upstream AUG. Така різниця пов’язана з тим, що послідовності в наборі даних подані в алфавіті A, C, G, T. Тому в коді використовується триплет ATG, а в біологічній інтерпретації він відповідає AUG у молекулі мРНК.

Для первинної оцінки зв’язку окремих простих ознак із цільовою змінною було обчислено коефіцієнти кореляції Пірсона та Спірмена. Ці коефіцієнти не описують повну складну залежність між послідовністю мРНК і translation efficiency, однак дають змогу оцінити напрям і силу простого зв’язку між окремими характеристиками та target. Результати кореляційного аналізу наведено в таблиці 4.8.

Таблиця 4.8 – Кореляція базових ознак із цільовою змінною target

Ознака	Pearson	Spearman
sequence_length	-0.2877	-0.3200
utr5_length	-0.1949	-0.2930
cds_length	-0.4520	-0.5465
utr3_length	-0.0060	-0.0189
gc_full	-0.1703	-0.1642
gc_utr5	-0.0598	-0.0667
gc_cds	-0.1070	-0.1010
gc_utr3	-0.1245	-0.1190
upstream_atg_count	-0.1935	-0.4008
splice_positive_count	-0.2974	-0.3034

З таблиці 4.8 видно, що найсильніший від’ємний зв’язок серед базових характеристик має довжина CDS. Для неї коефіцієнт Спірмена становить -0.5465, що вказує на помітну монотонну залежність між збільшенням довжини кодуючої ділянки та зменшенням значення target.

Також виражений від’ємний зв’язок спостерігається для upstream_atg_count, splice_positive_count і загальної довжини послідовності.

Натомість довжина 3′UTR має майже нульову кореляцію з target у межах простого однофакторного аналізу. Це не означає, що 3′UTR не має біологічного значення, а лише показує, що її довжина сама по собі не формує сильного лінійного або монотонного зв’язку з цільовою змінною в цьому наборі. У складнішій моделі 3′UTR може впливати на прогноз у взаємодії з іншими ознаками, послідовнісними мотивами або структурними характеристиками.

GC-вміст у різних ділянках також має від’ємну кореляцію з target, однак її абсолютні значення є меншими порівняно з довжиною CDS або upstream ATG. Це свідчить про те, що GC-вміст може бути корисним допоміжним сигналом, але сам по собі не є достатнім для якісного прогнозування ефективності трансляції. Саме тому в роботі використовується не одна окрема ознака, а мультимодальне представлення, яке об’єднує багато різних характеристик.

Для додаткового аналізу інформативності ознак було використано оцінку важливості за моделлю Random Forest і permutation importance. На відміну від простих кореляцій, такі оцінки дозволяють врахувати нелінійні залежності та взаємодію ознак у межах моделі. Найважливіші ознаки за результатами цього аналізу наведено в таблиці 4.9.

Таблиця 4.9 – Найважливіші числові ознаки за результатами аналізу importance

Ознака	Random Forest importance	Permutation importance
cds_end	0.3031	0.1011
upstream_atg_count	0.0755	0.0772
cds_length	0.0623	0.0435
mfe_5utr_300_norm	0.0335	0.0168

Продовження таблиці 4.9

Ознака	Random Forest importance	Permutation importance
mfe_start_window_200_norm	0.0263	0.0120
kmer3_TTG	0.0196	0.0052
kmer3_CAG	0.0181	0.0051
kmer3_GAT	0.0148	0.0020
gc_5utr	0.0132	0.0028
kmer3_TCA	0.0118	0.0027

З таблиці 4.9 видно, що найбільшу важливість має ознака cds_end. Вона фактично пов’язана з положенням кінця кодуючої ділянки та частково відображає довжинні властивості транскрипту. Її висока важливість не означає прямої причинної залежності, але показує, що інформація про межі CDS є суттєвою для прогнозування target у межах сформованого набору ознак.

Другою за важливістю ознакою є upstream_atg_count. Це узгоджується з біологічною логікою, оскільки upstream старт-кодони в 5′UTR можуть впливати на ініціацію трансляції основної кодуючої ділянки. Висока permutation importance цієї ознаки вказує на те, що її вилучення або перемішування помітно погіршує якість прогнозування базової моделі.

Серед важливих ознак також присутні MFE-характеристики, зокрема normalized MFE для 5′UTR і ділянки навколо старт-кодона. Це свідчить про те, що локальні структурні властивості мРНК можуть бути корисними для прогнозування ефективності трансляції. Наявність MFE-ознак серед найважливіших підтверджує доцільність включення структурної інформації до числової модальності.

До переліку важливих ознак увійшли також окремі 3-мери, зокрема TTG, CAG, GAT і TCA. Їхня важливість показує, що короткі локальні мотиви або особливості нуклеотидного складу можуть бути пов’язані з

цільовою змінною. Водночас жоден окремий 3-мер не домінує над усіма іншими ознаками, тому їх доцільно розглядати як частину ширшого статистичного опису послідовності.

Службові ознаки invalid_base_count і mfe_available не мали суттєвої важливості в підсумковій таблиці. Це очікувано, оскільки після попередньої обробки послідовності були уніфіковані, а MFE-ознаки були доступні для сформованого набору. Такі контрольні ознаки потрібні для стабільності пайплайну, але вони не є основними біологічними предикторами.

Порівняння таблиць 4.8 і 4.9 показує, що простий кореляційний аналіз і модельна оцінка важливості доповнюють одне одного. Кореляції демонструють загальні однофакторні тенденції, тоді як importance дозволяє побачити, які ознаки реально використовуються моделлю в нелінійному прогнозуванні. Наприклад, довжина CDS має сильну від’ємну кореляцію з target і водночас входить до переліку важливих ознак. MFE-ознаки мають меншу очевидність у простому аналізі, але виявляються корисними для моделі.

Таким чином, аналіз сформованих структурно-статистичних ознак показав, що числова модальність містить декілька інформативних груп характеристик. Найважливішими виявилися ознаки, пов’язані з межами та довжиною CDS, кількістю upstream ATG, локальними MFE-характеристиками та окремими 3-мерними мотивами. Це підтверджує доцільність використання мультимодального підходу, у якому повна послідовність мРНК доповнюється інтерпретованими структурно-статистичними ознаками.

4.4 Результати базових моделей машинного навчання

Після формування структурно-статистичних ознак було виконано навчання базових моделей машинного навчання. Метою цього етапу було отримання контрольного рівня якості, з яким надалі порівнювалася

мультимодальна модель глибинного навчання. Базові моделі використовували лише числові ознаки, сформовані у скрипті 04_features.py, і не працювали безпосередньо з повною нуклеотидною послідовністю як із послідовністю символів.

Для базового порівняння було використано три регресійні моделі: Ridge, RandomForestRegressor і GradientBoostingRegressor. Ridge-регресія була використана як лінійна модель з регуляризацією. RandomForestRegressor і GradientBoostingRegressor були використані як нелінійні ансамблеві моделі, здатні враховувати складніші залежності між числовими ознаками та цільовою змінною. Навчання та оцінювання базових моделей було реалізовано у скрипті 05_baseline.py.

Оцінювання виконувалося на валідаційній і тестовій вибірках. Для аналізу якості використовувалися чотири метрики: MAE, RMSE, коефіцієнт кореляції Пірсона та коефіцієнт кореляції Спірмена. MAE і RMSE характеризують абсолютну похибку прогнозування, тоді як кореляційні метрики показують, наскільки добре модель відтворює зв’язок між фактичними та прогнозованими значеннями target. Результати базових моделей наведено в таблиці 4.10.

Таблиця 4.10 – Результати базових моделей машинного навчання

Модель	Вибірка	MAE	RMSE	Pearson	Spearman
Ridge	Validation	0.4091	0.5185	0.6701	0.6668
	Test	0.4052	0.5227	0.6496	0.6639
RandomForest–Regressor	Validation	0.3808	0.4881	0.7157	0.7065
	Test	0.3690	0.4801	0.7176	0.7125
GradientBoosting–Regressor	Validation	0.3756	0.4785	0.7293	0.7195
	Test	0.3642	0.4745	0.7259	0.7180

З таблиці 4.10 видно, що найнижчу похибку серед базових моделей показала GradientBoostingRegressor. На тестовій вибірці ця модель отримала

RMSE 0.4745 та MAE 0.3642. Також вона має найвищі значення кореляційних метрик серед базових моделей: Pearson 0.7259 і Spearman 0.7180. Це свідчить про те, що градієнтний бустинг краще використовує сформовані структурно-статистичні ознаки, ніж лінійна Ridge-регресія та RandomForestRegressor.

Найнижчу якість серед трьох базових моделей показала Ridge-регресія. На тестовій вибірці її RMSE становить 0.5227, а Pearson – 0.6496. Такий результат є очікуваним, оскільки Ridge є лінійною моделлю і не може повною мірою врахувати нелінійні взаємодії між ознаками. У задачі прогнозування ефективності трансляції мРНК такі взаємодії можуть бути важливими, оскільки target залежить не від однієї характеристики, а від поєднання довжин, нуклеотидного складу, локальних мотивів і структурних властивостей.

RandomForestRegressor показала кращий результат, ніж Ridge-регресія. На тестовій вибірці її RMSE становить 0.4801, а Pearson – 0.7176. Це підтверджує, що нелінійні ансамблеві моделі краще підходять для роботи зі сформованими біологічними ознаками. Водночас RandomForestRegressor поступилася GradientBoostingRegressor за всіма основними метриками, хоча різниця між ними є меншою, ніж між ансамблевими моделями та лінійною Ridge-регресією.

Порівняння результатів на validation і test показує, що базові моделі не демонструють різкого погіршення якості на тестовій вибірці. Це свідчить про відносну стабільність отриманих оцінок і про те, що сформоване розбиття даних не призвело до очевидного дисбалансу між validation і test. Наприклад, для GradientBoostingRegressor значення RMSE на validation становить 0.4785, а на test – 0.4745, тобто якість на тестовій вибірці залишається на тому самому рівні.

Отримані результати показують, що сформовані структурно-статистичні ознаки є інформативними для задачі прогнозування translation efficiency. Навіть без безпосередньої обробки повної нуклеотидної

послідовності класичні моделі змогли досягти помірно високих значень кореляційних метрик. Це підтверджує доцільність включення таких ознак до мультимодальної моделі як окремої числової модальності.

Водночас базові моделі мають обмеження. Вони використовують уже сформований фіксований вектор ознак і не навчаються безпосередньо виділяти локальні патерни з повної послідовності мРНК. Тому вони не можуть повністю замінити послідовнісну гілку моделі глибинного навчання. Їх доцільно розглядати як контрольний рівень якості, який показує, наскільки добре задачу можна розв’язувати за допомогою ручно сформованих числових характеристик.

Таким чином, серед базових моделей найкращий результат показала GradientBoostingRegressor. Її результати використовуються надалі як основний baseline для порівняння з мультимодальною моделлю глибинного навчання. Подальший аналіз у роботі спрямований на перевірку того, чи дозволяє поєднання повної послідовності мРНК, позиційної інформації та структурно-статистичних ознак покращити якість прогнозування порівняно з базовими підходами.

4.5 Результати навчання моделей глибинного навчання

Після навчання базових моделей машинного навчання було виконано навчання моделей глибинного навчання. На цьому етапі порівнювалися дві архітектури: SequenceOnlyCNN і MultimodalTEModel. Перша модель використовувала тільки повну нуклеотидну послідовність мРНК, тоді як друга додатково враховувала позиційну інформацію про функціональні ділянки та числові структурно-статистичні ознаки.

Метою цього етапу було перевірити, чи дає мультимодальне представлення перевагу порівняно з моделлю, яка працює лише з послідовністю. Обидві моделі навчалися в режимі full-sequence, тобто з використанням повних послідовностей мРНК та динамічного padding у

межах mini-batch. Такий режим було обрано для того, щоб не втрачати інформацію з довгих транскриптів і не обмежувати модель лише початковим фрагментом мРНК.

Результати найкращих валідаційних checkpoint для моделей глибинного навчання наведено в таблиці 4.11.

Таблиця 4.11 – Валідаційні результати моделей глибинного навчання

Показник	SequenceOnlyCNN	MultimodalTEModel
Режим послідовності	full	full
Кількість числових ознак	0	83
MAE	0.5242	0.3774
RMSE	0.6646	0.4816
Pearson	0.4599	0.7252
Spearman	0.4438	0.7193

З таблиці 4.11 видно, що мультимодальна модель перевищила sequence-only модель за всіма основними валідаційними метриками. Значення RMSE для SequenceOnlyCNN становило 0.6646, тоді як для MultimodalTEModel воно зменшилося до 0.4816. Також суттєво зросли кореляційні метрики: Pearson збільшився з 0.4599 до 0.7252, а Spearman – з 0.4438 до 0.7193. Це свідчить про те, що використання лише послідовності в межах простої CNN-архітектури було недостатнім для якісного прогнозування target, тоді як додавання структурно-статистичних ознак і позиційної інформації істотно покращило здатність моделі відтворювати залежності в даних.

Для детальнішого аналізу було розглянуто динаміку навчання моделей за епохами. Основні значення метрик для SequenceOnlyCNN наведено в таблиці 4.12.

Таблиця 4.12 – Динаміка навчання SequenceOnlyCNN за вибраними епохами

Епоха	Train RMSE	Validation RMSE	Train Pearson	Validation Pearson
1	0.6889	0.6981	-0.0003	-0.0541
8	0.6860	0.6976	0.0165	0.2665
12	0.6737	0.6725	0.1737	0.2925
17	0.6232	0.6646	0.4112	0.4599
22	0.5957	0.6811	0.4898	0.5083

З таблиці 4.12 видно, що SequenceOnlyCNN поступово покращувала якість на навчальній вибірці, однак покращення на валідаційній вибірці було менш стабільним. Найкраще значення Validation RMSE було отримано не на останній епосі, а на одному з проміжних checkpoint. Це підтверджує доцільність використання механізму збереження найкращої моделі за валідаційною метрикою, а не автоматичного використання останньої епохи навчання.

Водночас значення кореляційних метрик для SequenceOnlyCNN залишилися відносно невисокими. Означає, що модель, яка використовувала тільки послідовність, мала обмежену здатність правильно відтворювати відносний порядок транскриптів за ефективністю трансляції. Імовірною причиною є складність задачі: повна мРНК є довгою послідовністю, у якій сигнали можуть бути розподілені між 5′UTR, CDS і 3′UTR, а їх значення залежить від функціонального контексту. Динаміку навчання мультимодальної моделі наведено в таблиці 4.13.

Таблиця 4.13 – Динаміка навчання MultimodalTEModel за вибраними епохами

Епоха	Train RMSE	Validation RMSE	Train Pearson	Validation Pearson
1	0.5565	0.5154	0.5804	0.6751

Продовження таблиці 4.13

Епоха	Train RMSE	Validation RMSE	Train Pearson	Validation Pearson
3	0.4977	0.4945	0.6852	0.7065
5	0.4806	0.4884	0.7109	0.7147
8	0.4666	0.4816	0.7305	0.7238
9	0.4628	0.4816	0.7357	0.7252
14	0.4497	0.4885	0.7529	0.7174

З таблиці 4.13 видно, що MultimodalTEModel уже на перших епохах демонструвала суттєво кращу якість порівняно з SequenceOnlyCNN. Validation RMSE зменшився з 0.5154 на першій епосі до 0.4816 на найкращому checkpoint, а Validation Pearson досяг 0.7252. Після цього подальше зменшення Train RMSE не приводило до покращення Validation RMSE, що свідчить про досягнення оптимальної ділянки навчання для цієї конфігурації.

Порівняння таблиць 4.12 і 4.13 показує, що мультимодальна модель не лише досягла нижчої похибки, а й навчалася значно ефективніше. Вже на першій епосі MultimodalTEModel мала кращі валідаційні метрики, ніж SequenceOnlyCNN після багатьох епох навчання. Це означає, що додаткові джерела інформації підвищили інформативність вхідного представлення ще на початкових етапах оптимізації моделі. Такий результат узгоджується із загальною ідеєю мультимодального підходу, оскільки модель отримує не один тип даних, а кілька взаємодоповнювальних описів одного й того самого транскрипту.

Це свідчить про те, що додаткові числові та позиційні ознаки надають моделі інформацію, яку складно автоматично виділити лише зі згорткової обробки довгої нуклеотидної послідовності. Завдяки цьому модель може швидше сформувати корисне внутрішнє представлення мРНК і стабільніше зменшувати похибку на валідаційній вибірці. На рисунку 4.4 наведено криву навчання моделі SequenceOnlyCNN.

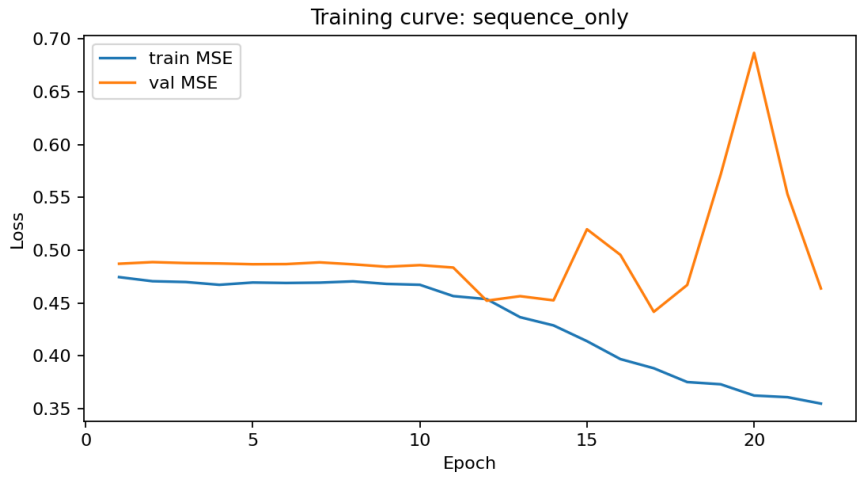


Рисунок 4.4 – Крива навчання моделі SequenceOnlyCNN

Крива навчання SequenceOnlyCNN показує, що модель поступово зменшувала похибку на навчальній вибірці, але її валідаційна похибка залишалася значно вищою порівняно з мультимодальною моделлю. Це може свідчити про недостатню інформативність лише послідовнісного представлення в межах використаної CNN-архітектури. Модель отримувала повну мРНК, але не мала явно поданих агрегованих характеристик, таких як довжина CDS, GC-вміст, upstream ATG, MFE-ознаки та k-мерні частоти.

На рисунку 4.5 наведено криву навчання моделі MultimodalTEModel.

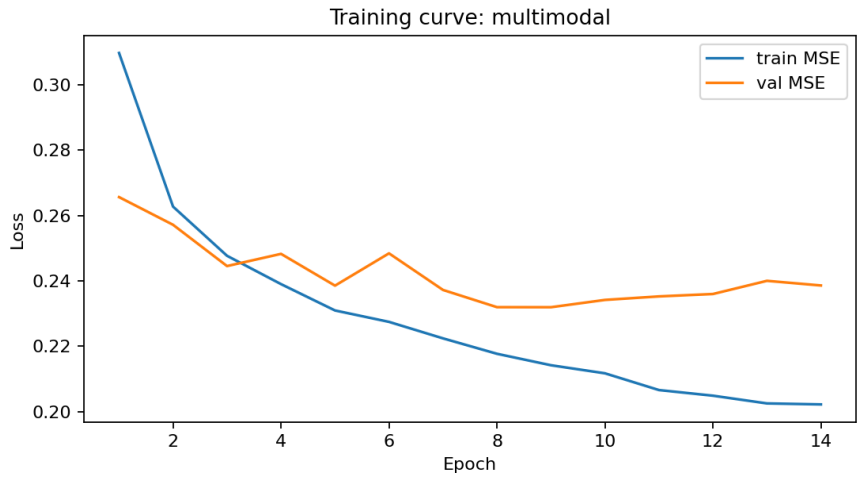


Рисунок 4.5 – Крива навчання моделі MultimodalTEModel

Крива навчання MultimodalTEModel демонструє швидше досягнення нижчої похибки на валідаційній вибірці. Модель краще використовувала доступну інформацію, оскільки отримувала не тільки нуклеотидну послідовність, а й позиційний контекст функціональних ділянок та числові структурно-статистичні ознаки. Це дозволило їй формувати більш повне представлення транскрипту.

Отримані результати показують, що сама повна послідовність мРНК містить корисну інформацію, але для даної задачі її використання у sequence-only CNN виявилося недостатнім. Більш якісний результат було отримано тоді, коли до послідовнісної гілки було додано позиційну інформацію та числові ознаки. Це підтверджує доцільність мультимодального підходу, запропонованого в роботі.

Водночас sequence-only модель має важливе експериментальне значення. Вона виконує роль нейромережевого baseline і дозволяє оцінити внесок додаткових модальностей. Оскільки мультимодальна модель показала суттєво кращі валідаційні показники, подальший підбір гіперпараметрів було зосереджено саме на MultimodalTEModel.

Таким чином, результати навчання моделей глибинного навчання показали перевагу мультимодальної архітектури над sequence-only підходом. Поєднання повної послідовності мРНК, позиційної інформації та структурно-статистичних ознак забезпечило нижчу похибку прогнозування та вищі кореляційні показники на валідаційній вибірці. Це стало підставою для подальшого систематичного підбору гіперпараметрів мультимодальної моделі.

4.6 Підбір гіперпараметрів мультимодальної моделі

Після порівняння SequenceOnlyCNN і MultimodalTEModel подальші експерименти було зосереджено на мультимодальній архітектурі. Це пов’язано з тим, що саме вона показала кращі валідаційні результати та

реалізує основну ідею роботи: поєднання повної послідовності мРНК, позиційної інформації про функціональні ділянки та структурно-статистичних ознак. Для покращення якості прогнозування було виконано підбір гіперпараметрів за допомогою скрипта 13_tune_deep.py.

Підбір гіперпараметрів виконувався на навчальній і валідаційній вибірках, тоді як тестова вибірка використовувалася лише для фінального незалежного оцінювання обраної моделі. Це дозволило уникнути витоку інформації з тестових даних під час вибору найкращої конфігурації.

Під час підбору змінювалися параметри навчання та архітектури моделі: learning rate, weight decay, batch size, тип функції втрат, scheduler, gradient clipping, розміри nucleotide і region embedding, кількість каналів у згорткових шарах, розмір прихованого шару числової гілки та dropout. Загальну характеристику підбору гіперпараметрів наведено в таблиці 4.14.

Таблиця 4.14 – Загальна характеристика підбору гіперпараметрів мультимодальної моделі

Показник	Значення
Модель	MultimodalTEModel
Кількість trials	10
Використана вибірка для навчання	Train
Використана вибірка для вибору конфігурації	Validation
Критерій вибору найкращої конфігурації	Validation RMSE
Режим обробки послідовностей	Full-sequence
Формування batch	Dynamic padding
Кількість числових ознак	83
Основний збережений checkpoint	best_model.pt
Фінально зафіксована модель	final_multimodal_model.pt

З таблиці 4.14 видно, що підбір виконувався саме для мультимодальної моделі в режимі обробки повних послідовностей. Це

важливо, оскільки експеримент не зводився до роботи з фіксовано обрізаними фрагментами мРНК. Модель отримувала повні транскрипти, а різна довжина послідовностей оброблялася за допомогою dynamic padding у межах mini-batch.

У результаті підбору було отримано декілька успішних конфігурацій. Найкращі результати trials за значенням validation RMSE наведено в таблиці 4.15.

Таблиця 4.15 – Найкращі конфігурації за результатами підбору гіперпараметрів

Показник	Trial 3	Trial 1	Trial 9	Trial 4	Trial 8
Best epoch	20	14	15	17	12
Validation RMSE	0.4733	0.4772	0.4782	0.4784	0.4786
Validation MAE	0.3731	0.3715	0.3708	0.3761	0.3730
Pearson	0.7350	0.7302	0.7316	0.7326	0.7287
Spearman	0.7229	0.7243	0.7236	0.7256	0.7281
Learning rate	0.0001	0.0010	0.0001	0.0001	0.0005
Weight decay	0.0001	0.0000	0.0005	0.0005	0.00001
Dropout	0.1	0.2	0.2	0.3	0.2
Batch size	8	8	8	4	—
Loss	Smooth L1	MSE	Smooth L1	Smooth L1	—

Як видно з таблиці 4.15, найкращою конфігурацією за критерієм validation RMSE став Trial 3. Його значення Validation RMSE становило 0.4733, MAE – 0.3731, Pearson – 0.7350, Spearman – 0.7229. Порівняно з іншими конфігураціями, цей trial забезпечив найнижчу похибку RMSE на валідаційній вибірці. Саме тому його checkpoint було обрано як основну модель для подальшого фінального оцінювання.

Важливо зазначити, що різниця між кількома найкращими trials була відносно невеликою. Наприклад, Trial 1, Trial 9, Trial 4 і Trial 8 також

показали близькі значення validation RMSE. Це свідчить про те, що мультимодальна архітектура є достатньо стабільною в межах кількох близьких конфігурацій, але найкраща якість була отримана при меншому dropout і помірному weight decay.

Параметри найкращої конфігурації наведено в таблиці 4.16.

Таблиця 4.16 – Параметри найкращої мультимодальної конфігурації

Параметр	Значення
Trial	3
Model name	MultimodalTEModel
Best epoch	20
Learning rate	0.0001
Weight decay	0.0001
Dropout	0.1
Batch size	8
Loss	Smooth L1
Optimizer	AdamW
Embedding dim	32
Region embedding dim	4
Conv channels	128, 64
Numeric hidden dim	128
Gradient clipping	1.0
Scheduler	Так
Bucket by length	Так
Smooth L1 beta	1.0

З таблиці 4.16 видно, що найкраща конфігурація використовувала порівняно невелике значення learning rate, помірне weight decay і dropout 0.1. Це означає, що модель краще навчалася при більш обережному оновленні ваг і менш агресивній регуляризації dropout. Збільшення dropout у деяких

інших trials не призвело до покращення RMSE, що може свідчити про те, що надмірне випадкове вимикання нейронів зменшувало здатність моделі використовувати наявні ознаки.

Архітектурно найкраща конфігурація мала nucleotide embedding розміром 32, region embedding розміром 4, згорткові канали 128 і 64, а також прихований шар числової гілки розміром 128. Така структура забезпечувала достатню ємність для обробки повної послідовності та числових ознак, але не була надмірно складною. Це важливо для задачі, де кількість записів є обмеженою, а занадто велика модель могла б швидко перенавчатися.

Вибір Smooth L1 як функції втрат для найкращої конфігурації також є обґрунтованим. На відміну від MSE, Smooth L1 є менш чутливою до окремих великих похибок. Для біологічних даних це може бути корисним, оскільки окремі транскрипти можуть мати значення target, які суттєво відрізняються від основної маси спостережень. При цьому Smooth L1 зберігає здатність ефективно оптимізувати модель для малих і середніх похибок.

Після завершення підбору checkpoint найкращої конфігурації було збережено як best_model.pt. Додатково фінальна мультимодальна модель була зафіксована як final_multimodal_model.pt. Це дозволило відокремити модель, обрану за результатами підбору гіперпараметрів, від проміжних checkpoint інших trials. Саме ця фінальна модель використовується для подальшого незалежного оцінювання на тестовій вибірці.

Таким чином, підбір гіперпараметрів дозволив уточнити конфігурацію мультимодальної моделі та вибрати checkpoint з найкращим validation RMSE. Найкращою стала конфігурація з learning rate 0.0001, weight decay 0.0001, dropout 0.1, batch size 8, Smooth L1 loss, embedding dim 32, conv channels 128 і 64 та numeric hidden dim 128. Отримана модель була збережена як фінальна та використана на наступному етапі для тестового оцінювання.

4.7 Фінальне тестове оцінювання мультимодальної моделі

Після завершення підбору гіперпараметрів було виконано фінальне тестове оцінювання мультимодальної моделі з використанням checkpoint final_multimodal_model.pt, який відповідав найкращій конфігурації за валідаційним RMSE. Оцінювання проводилося на тестовій вибірці, яка не використовувалася під час навчання, валідації або підбору гіперпараметрів.

Тестова вибірка містила 1542 транскрипти. Як було показано в підрозділі 4.2, розбиття виконувалося з урахуванням gene-груп, тому транскрипти одного й того самого гена не потрапляли одночасно до різних вибірок. Це робить тестове оцінювання більш строгим, оскільки модель перевірялася на генах, відсутніх під час навчання.

Фінальне оцінювання виконувалося за метриками MAE, RMSE, Pearson і Spearman, що дозволяють оцінити як абсолютну похибку прогнозування, так і кореляційний зв’язок між фактичними й прогнозованими значеннями target. Результати фінального тестового оцінювання наведено в таблиці 4.17.

Таблиця 4.17 – Фінальні тестові результати мультимодальної моделі

Показник	Значення
Модель	MultimodalTEModel
Режим послідовності	full
Кількість числових ознак	83
MAE	0.3611
RMSE	0.4692
Pearson	0.7308
Spearman	0.7232

З таблиці 4.17 видно, що фінальна мультимодальна модель досягла RMSE 0.4692 та MAE 0.3611 на тестовій вибірці. Значення Pearson

становило 0.7308, а Spearman – 0.7232. Це свідчить про те, що модель не лише зменшує абсолютну похибку прогнозування, а й досить добре відтворює відносний порядок транскриптів за значенням ефективності трансляції.

Для перевірки стабільності результатів доцільно порівняти показники найкращої валідаційної конфігурації та фінальні тестові метрики. Таке порівняння наведено в таблиці 4.18.

Таблиця 4.18 – Порівняння валідаційних і тестових результатів фінальної мультимодальної моделі

Вибірка	MAE	RMSE	Pearson	Spearman
Validation	0.3731	0.4733	0.7350	0.7229
Test	0.3611	0.4692	0.7308	0.7232

Як видно з таблиці 4.18, якість моделі на тестовій вибірці залишилася близькою до валідаційної. RMSE на тестовій вибірці навіть дещо нижчий, ніж на validation, а значення Spearman майже не змінилося. Це свідчить про відсутність різкого перенавчання на валідаційній вибірці та про достатню стабільність отриманої конфігурації.

Водночас незначне зменшення Pearson з 0.7350 на validation до 0.7308 на test є очікуваним, оскільки тестова вибірка містить незалежні gene-групи. Така різниця є малою і не свідчить про суттєве погіршення узагальнювальної здатності моделі. Навпаки, близькість validation і test показників підтверджує, що обрана мультимодальна модель зберігає якість на даних, які не брали участі в навчанні та підборі параметрів, і підтверджує надійність проведеного оцінювання.

На рисунку 4.6 наведено діаграму розсіювання фактичних і прогнозованих значень target для тестової вибірки.



Рисунок 4.6 – Порівняння фактичних і прогнозованих значень target на тестовій вибірці

Діаграма на рисунку 4.6 дозволяє візуально оцінити якість прогнозування. Чим ближче точки розташовані до діагоналі, тим точніше модель відтворює фактичні значення target. Наявність позитивної кореляції між фактичними та прогнозованими значеннями узгоджується з отриманим Pearson 0.7308. Водночас розсіювання точок навколо діагоналі показує, що задача не є тривіальною, а частина транскриптів прогнозується з помітною похибкою.

На рисунку 4.7 наведено розподіл залишків моделі на тестовій вибірці.

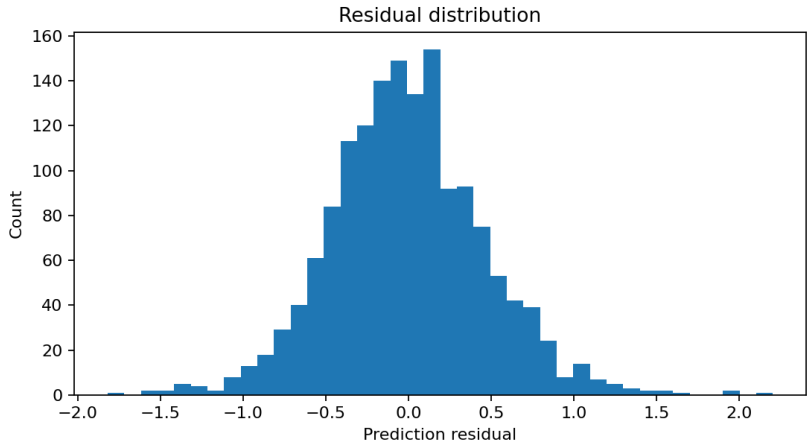


Рисунок 4.7 – Розподіл залишків мультимодальної моделі на тестовій вибірці

Розподіл залишків на рисунку 4.7 показує характер помилок прогнозування. Якщо залишки зосереджені поблизу нуля, це означає, що модель не має вираженого систематичного зміщення в один бік. Водночас наявність ширших хвостів розподілу вказує на існування окремих транскриптів, для яких модель помиляється сильніше. Такі випадки можуть бути пов’язані з біологічними факторами, які не були представлені у вхідних даних, наприклад із клітинним контекстом, особливостями регуляції трансляції або експериментальним шумом у target.

На рисунку 4.8 наведено розподіл фактичних значень target у тестовій вибірці.

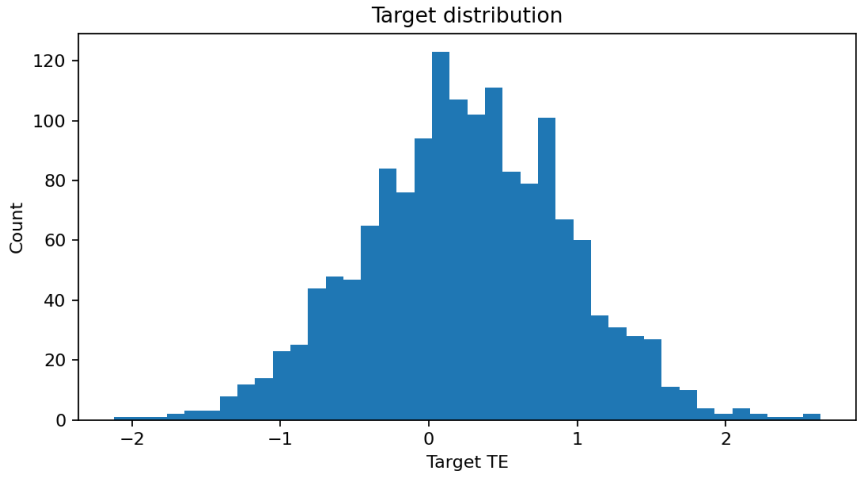


Рисунок 4.8 – Розподіл фактичних значень target у тестовій вибірці

Розподіл target на рисунку 4.8 є важливим для інтерпретації похибки. Якщо значення target мають широкий діапазон і містять крайні спостереження, то RMSE може бути чутливим до окремих складних прикладів. Саме тому в роботі додатково використовується MAE, який є менш чутливим до великих поодиноких похибок, а також кореляційні метрики Pearson і Spearman.

Отримані результати підтверджують, що фінальна мультимодальна модель має достатню узагальнювальну здатність на незалежній тестовій

вибірці. Вона зберегла якість, близьку до валідаційної, та показала RMSE 0.4692 при Pearson 0.7308 і Spearman 0.7232. Це свідчить про те, що запропонований підхід здатний прогнозувати ефективність трансляції мРНК не лише на даних, використаних під час налаштування моделі, а й на відкладеному тестовому наборі.

Фінальне тестове оцінювання також показує, що об’єднання повної послідовності, регіональної інформації та структурно-статистичних ознак є доцільним для поставленої задачі. Модель не обмежується аналізом окремої ділянки мРНК, а використовує повний транскрипт і додаткові інтерпретовані характеристики. Це відповідає головній ідеї роботи та підтверджує практичну придатність розробленого методу для прогнозування translation efficiency.

4.8 Порівняння моделей та аналіз похибок

Після фінального тестового оцінювання мультимодальної моделі було виконано узагальнене порівняння всіх основних моделей, використаних у роботі. Метою цього порівняння було визначити, чи забезпечує запропонований мультимодальний підхід покращення порівняно з класичними моделями машинного навчання, які використовують лише структурно-статистичні числові ознаки. Таке порівняння дозволяє оцінити не лише абсолютну якість фінальної моделі, а й її місце серед інших підходів, реалізованих у межах експерименту. Крім того, воно дає змогу зрозуміти, чи є додавання послідовнісної та регіональної інформації доцільним з погляду підсумкових метрик.

До порівняння було включено Ridge, RandomForestRegressor, GradientBoostingRegressor та фінальну MultimodalTEModel. Усі значення в таблиці наведено для тестової вибірки, яка не використовувалася під час навчання моделей або вибору гіперпараметрів. Результати порівняння наведено в таблиці 4.19.

Таблиця 4.19 – Порівняння моделей на тестовій вибірці

Показник	Ridge	RandomForest-Regressor	GradientBoosting-Regressor	MultimodalTE-Model
Тип модель	Базова лінійна модель	Базова ансамблева модель	Базова ансамблева модель	Модель глибинного навчання
MAE	0.4052	0.3690	0.3642	0.3611
RMSE	0.5227	0.4801	0.4745	0.4692
Pearson	0.6496	0.7176	0.7259	0.7308
Spearman	0.6639	0.7125	0.7180	0.7232
R^2	0.4212	0.5117	0.5230	0.5337

З таблиці 4.19 видно, що найкращу якість на тестовій вибірці показала мультимодальна модель. Вона має найнижчі значення MAE і RMSE, а також найвищі значення Pearson, Spearman і R^2. Порівняно з найкращою базовою моделлю GradientBoostingRegressor, мультимодальна модель зменшила MAE з 0.3642 до 0.3611, RMSE з 0.4745 до 0.4692, а також збільшила Pearson з 0.7259 до 0.7308 і Spearman з 0.7180 до 0.7232.

Абсолютне покращення порівняно з GradientBoostingRegressor є помірним, але стабільним за всіма метриками. Зменшення RMSE становить приблизно 0.0053, що відповідає приблизно 1.12% від RMSE базової моделі. Зменшення MAE становить приблизно 0.0030, тобто приблизно 0.84%. Водночас збільшення R^2 з 0.5230 до 0.5337 свідчить про те, що мультимодальна модель краще пояснює варіативність цільової змінної на тестовій вибірці.

Важливо, що покращення досягнуто не за рахунок використання лише більш складної моделі, а завдяки поєднанню різних типів інформації. Базові моделі працювали тільки з числовими ознаками, тоді як MultimodalTEModel додатково використовувала повну нуклеотидну послідовність і регіональне представлення позицій мРНК. Таким чином, фінальний результат

підтверджує основну гіпотезу роботи: прогнозування translation efficiency може бути покращене шляхом мультимодального об’єднання послідовнісної, позиційної та структурно-статистичної інформації.

Окремий інтерес становить порівняння Ridge і ансамблевих моделей. Ridge має найгірші показники серед усіх моделей, що вказує на недостатність лінійного опису залежності між ознаками та target. RandomForestRegressor і GradientBoostingRegressor показали значно кращі результати, що підтверджує наявність нелінійних залежностей у даних. Проте навіть найкраща базова ансамблева модель поступилася мультимодальній нейронній архітектурі.

Для аналізу характеру помилок фінальної моделі було додатково розглянуто залишки прогнозування. У програмній реалізації залишок визначався як різниця між прогнозованим і фактичним значенням target. Додатне середнє значення залишку означає, що модель у середньому завищує прогноз, а від’ємне – що модель у середньому занижує прогноз.

Спочатку було проаналізовано похибку залежно від довжини повної послідовності мРНК. Результати наведено в таблиці 4.20.

Таблиця 4.20 – Аналіз похибок мультимодальної моделі за довжиною послідовності

Показник	451–2064	2065–3264	3268–5033	5036–15159
Кількість записів	386	385	385	386
Мінімальна довжина	451	2065	3268	5036
Максимальна довжина	2064	3264	5033	15159
Mean residual	0.0217	-0.0042	-0.0182	-0.0212
MAE	0.3529	0.3665	0.3777	0.3474
RMSE	0.4617	0.4722	0.4919	0.4499

З таблиці 4.20 видно, що похибка не зростає монотонно зі збільшенням довжини повної послідовності. Найвищий RMSE

спостерігається у третьому інтервалі довжин, тоді як для найдовших послідовностей RMSE є найнижчим серед наведених груп. Це важливий результат, оскільки він показує, що використання full-sequence режиму та dynamic padding не призвело до явного погіршення якості на найдовших транскриптах.

Водночас середній залишок для довших груп має від'ємний знак, що може свідчити про незначну тенденцію до заниження прогнозу для частини довших транскриптів. Однак абсолютні значення mean residual є невеликими, тому не можна говорити про сильне систематичне зміщення лише за довжиною повної послідовності.

На рисунку 4.9 наведено графічне подання похибки залежно від довжини послідовності.

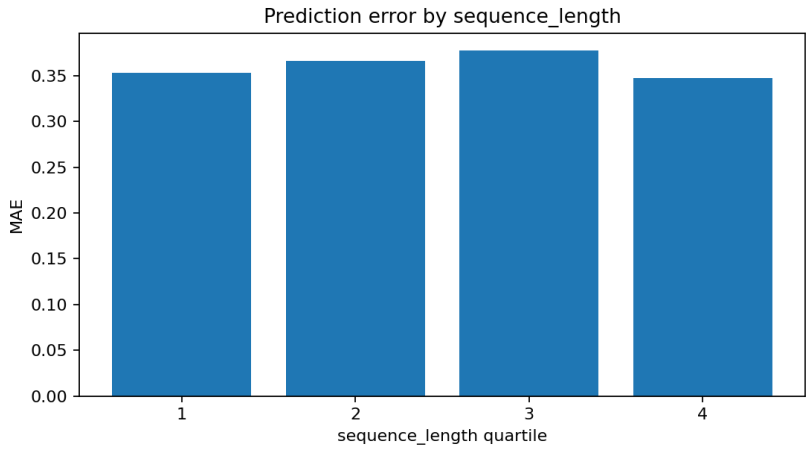


Рисунок 4.9 – Похибка прогнозування залежно від довжини послідовності мРНК

Графік на рисунку 4.9 доповнює результати таблиці 4.20 і показує, що модель зберігає відносно стабільну якість у різних діапазонах довжини послідовності. Це підтверджує доцільність обраного режиму обробки повної мРНК, оскільки довгі послідовності не були джерелом різкого зростання похибки.

Далі було проаналізовано похибки залежно від довжини CDS, оскільки в попередніх підрозділах було показано, що CDS length є однією з найбільш інформативних ознак. Результати наведено в таблиці 4.21.

Таблиця 4.21 – Аналіз похибок мультимодальної моделі за довжиною CDS

Показник	165–870	873–1398	1401–2103	2109–10011
Кількість записів	387	388	381	386
Мінімальна довжина CDS	165	873	1401	2109
Максимальна довжина CDS	870	1398	2103	10011
Mean residual	0.0306	-0.0086	-0.0306	-0.0137
MAE	0.3733	0.3606	0.3451	0.3653
RMSE	0.4826	0.4827	0.4516	0.4585

З таблиці 4.21 видно, що модель також не демонструє різкого погіршення якості для транскриптів із найдовшими CDS. Найменше значення RMSE спостерігається для третього інтервалу, а найбільші значення RMSE мають перший і другий інтервали. Це означає, що помилки моделі не можна пояснити лише довжиною кодуючої ділянки.

Разом з тим довжина CDS залишається важливою ознакою для прогнозування, оскільки вона має високий зв’язок із target і входить до переліку найважливіших числових характеристик. Тому результати таблиці 4.21 слід інтерпретувати не як відсутність значення CDS, а як відсутність прямого монотонного зростання похибки залежно від довжини CDS.

На рисунку 4.10 наведено графічне подання похибки залежно від довжини CDS.

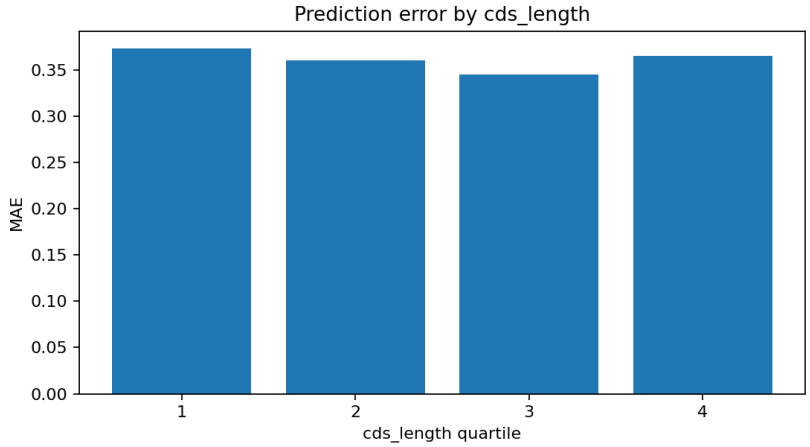


Рисунок 4.10 – Похибка прогнозування залежно від довжини CDS

Рисунок 4.10 показує, що модель зберігає приблизно однаковий рівень похибки для різних груп CDS length, що свідчить про використання не лише довжинних характеристик, а й інших типів інформації, зокрема GC-вмісту, 3-мерних частот, MFE-ознак та послідовнісного представлення.

Найбільш виражена закономірність була отримана під час аналізу похибок за значенням target. Результати наведено в таблиці 4.22.

Таблиця 4.22 – Аналіз похибок мультимодальної моделі за значенням target

Показник	-2.1227 – -(0.2098)	-0.2093 – 0.2473	0.2498 – 0.7340	0.7345 – 2.6388
Кількість записів	386	385	385	386
Мінімальне target	-2.1227	-0.2093	0.2498	0.7345
Максимальне target	-0.2098	0.2473	0.7340	2.6388
Mean residual	0.4115	0.0923	-0.1200	-0.4057
MAE	0.4642	0.2746	0.2591	0.4461
RMSE	0.5919	0.3411	0.3290	0.5523

З таблиці 4.22 видно, що найбільші похибки спостерігаються для крайніх значень target. Для найнижчого інтервалу target середній залишок є додатним, тобто модель у середньому завищує прогноз. Для найвищого інтервалу target середній залишок є від'ємним, тобто модель у середньому занижує прогноз. Це означає, що модель має тенденцію до згладжування крайніх значень і частково наближає прогнози до центральної ділянки розподілу.

Така поведінка є типовою для регресійних моделей у задачах із нерівномірним розподілом цільової змінної. Оскільки більшість записів розташована ближче до центральних значень target, модель краще прогнозує середній діапазон і гірше відтворює крайні випадки. Це підтверджується нижчими значеннями MAE і RMSE у двох центральних інтервалах target.

На рисунку 4.11 наведено графічне подання похибки залежно від значення target.



Рисунок 4.11 – Похибка прогнозування залежно від значення target

Рисунок 4.11 візуально підтверджує, що найбільші помилки виникають на краях розподілу target. Це важливе обмеження фінальної моделі, оскільки саме транскрипти з дуже низькою або дуже високою ефективністю трансляції можуть бути найбільш цікавими з біологічної точки зору. У подальших дослідженнях якість для крайніх значень можна

покращувати за допомогою додаткового балансування навчальної вибірки, спеціалізованих функцій втрат або врахування додаткового клітинного контексту.

Загалом аналіз похибок показав, що фінальна мультимодальна модель має стабільну якість для різних груп довжини послідовності та CDS, але гірше прогнозує крайні значення target. Це свідчить про те, що основним джерелом складності є не лише довжина мРНК, а й біологічна неоднорідність транскриптів з дуже низькою або дуже високою ефективністю трансляції.

Порівняння моделей і аналіз залишків дозволяють зробити декілька висновків. По-перше, мультимодальна модель показала найкращий результат серед усіх розглянутих моделей на тестовій вибірці. По-друге, покращення порівняно з GradientBoostingRegressor є помірним, але стабільним за всіма метриками. По-третє, найбільш проблемними для прогнозування залишаються крайні значення target, тоді як залежність похибки від довжини послідовності або CDS не має вираженого монотонного характеру.

Таким чином, результати підтверджують ефективність запропонованого мультимодального підходу, але також показують його обмеження. Модель добре відтворює загальну залежність між ознаками мРНК і translation efficiency, однак не повністю вирішує проблему точного прогнозування крайніх значень цільової змінної. Це створює підґрунтя для подальшого аналізу практичного використання моделі та обмежень проведеного експерименту.

4.9 Обмеження проведеного експерименту

Після порівняння моделей та аналізу похибок доцільно окремо розглянути обмеження проведеного експерименту. Це необхідно для коректної інтерпретації отриманих результатів і визначення напрямів

подальшого розвитку розробленого методу. Незважаючи на те, що мультимодальна модель показала найкращі результати серед розглянутих підходів, отримані метрики не слід трактувати як остаточне розв’язання задачі прогнозування ефективності трансляції мРНК.

Першим обмеженням є використання одного основного набору даних. У роботі застосовано набір morrislab/translation-efficiency-human, який містить послідовності людських транскриптів і значення target для задачі регресійного прогнозування. Такий набір є придатним для побудови й оцінювання запропонованого методу, однак він не дозволяє повністю перевірити здатність моделі узагальнювати результати на інші експериментальні платформи, інші типи клітин або інші організми. Тому отримані результати насамперед характеризують якість моделі в межах використаного набору даних.

Другим обмеженням є відсутність повноцінного клітинного контексту. Ефективність трансляції мРНК залежить не лише від самої послідовності, а й від стану клітини, типу тканини, концентрації регуляторних білків, активності сигнальних шляхів, умов експерименту та інших факторів. У межах використаного набору даних основними доступними вхідними даними були послідовність, CDS, splice, gene, chromosome і target. Тому мультимодальна модель враховувала повну послідовність і структурно-статистичні ознаки, але не могла явно моделювати клітинний стан.

Третім обмеженням є спосіб представлення вторинної структури мРНК. У роботі використовувалися локальні MFE-ознаки, зокрема для 5′UTR, ділянки навколо старт-кодона та першого фрагмента CDS. Такий підхід є практичним і обчислювально прийнятним, однак він не описує повну вторинну структуру всієї молекули мРНК. Повна структурна інформація може бути значно складнішою, а локальні MFE-характеристики відображають лише окремі її аспекти. Крім того, MFE є наближеним

термодинамічним показником і не враховує всіх процесів, що відбуваються в клітині під час трансляції.

Четвертим обмеженням є архітектура послідовнісної гілки. У роботі було реалізовано згорткову нейронну мережу, яка дозволяє виділяти локальні патерни в нуклеотидній послідовності. Така архітектура є відносно стабільною, зрозумілою та придатною для бакалаврської роботи, однак вона має обмежену здатність моделювати дуже далекі залежності між позиціями в довгій послідовності мРНК. Для повнішого врахування далеких взаємодій у майбутньому можна дослідити складніші архітектури, наприклад моделі з attention-механізмами або спеціалізовані архітектури для довгих біологічних послідовностей.

П’ятим обмеженням є нерівномірність розподілу цільової змінної. Аналіз похибок у підрозділі 4.8 показав, що модель краще прогнозує центральну частину розподілу target і гірше працює з крайніми значеннями. Для транскриптів із дуже низькою ефективністю трансляції модель має тенденцію завищувати прогноз, а для транскриптів із дуже високою ефективністю трансляції – занижувати його. Це означає, що модель частково згладжує крайні значення і тяжіє до центральної ділянки розподілу. Така поведінка є важливою, оскільки саме крайні випадки можуть мати найбільший біологічний інтерес.

Шостим обмеженням є використання сформованих вручну структурно-статистичних ознак. У роботі було обчислено довжини ділянок, GC-вміст, upstream ATG, 3-мерні частоти та MFE-характеристики. Ці ознаки є біологічно інтерпретованими й корисними для прогнозування, однак вони не охоплюють усі можливі регуляторні сигнали. Наприклад, у роботі не виконувалося повне моделювання RNA-binding protein мотивів, мікроРНК-сайтів, модифікацій мРНК, експериментально визначеної структури або контексту рибосомного сканування.

Сьомим обмеженням є те, що фінальне оцінювання виконувалося в межах одного розбиття train, validation і test. Розбиття було зроблено

коректно, з урахуванням gene-груп, що запобігало потраплянню транскриптів одного гена до різних вибірок. Проте для ще надійнішої оцінки якості в майбутньому можна застосувати повторювану крос-валідацію за групами або кілька незалежних train/test розбиттів. Це дозволило б оцінити варіативність метрик залежно від конкретного складу вибірок.

Восьмим обмеженням є відсутність зовнішньої незалежної перевірки на іншому наборі даних. У межах цієї роботи модель оцінювалася на відкладеній тестовій частині основного набору. Такий підхід є коректним для внутрішнього експериментального дослідження, однак він не гарантує, що модель покаже таку саму якість на даних, отриманих в іншій лабораторії або за іншою експериментальною методикою. Для практичного використання моделі бажано виконати додаткове зовнішнє тестування.

Дев’ятим обмеженням є те, що розроблена модель прогнозує числове значення target, але не пояснює повністю причинно-наслідкові механізми трансляційної регуляції. Значення важливості ознак і аналіз похибок дають певну інтерпретацію, однак вони не є доказом прямого біологічного причинного впливу окремої ознаки на ефективність трансляції. Наприклад, висока важливість cds_end або cds_length може відображати як прямий зв’язок із властивостями транскрипту, так і непрямі статистичні закономірності в наборі даних.

Наведені обмеження не зменшують значення отриманих результатів, але визначають межі їх коректного застосування. У межах поставленої задачі було розроблено стабільний і відтворюваний програмний пайплайн, сформовано мультимодальне представлення мРНК, навчено нейронну модель і показано її перевагу над базовими підходами. Водночас подальше підвищення якості прогнозування потребує розширення вхідних даних, використання додаткових біологічних контекстів, зовнішньої валідації та дослідження складніших архітектур для довгих послідовностей.

4.10 Узагальнення результатів експериментального дослідження

У четвертому розділі було виконано експериментальне дослідження розробленого методу мультимодального прогнозування ефективності трансляції мРНК. Основна увага була зосереджена на аналізі вихідного набору даних, результатах попередньої обробки, сформованих структурно-статистичних ознаках, порівнянні базових моделей машинного навчання та моделей глибинного навчання, а також на фінальному тестовому оцінюванні мультимодальної моделі.

На першому етапі було проаналізовано вихідний набір даних morrislab/translation-efficiency-human. Було встановлено, що початковий набір містив 10281 запис і 7 основних колонок: transcript_id, gene, chromosome, sequence, cds, splice і target. Цільова змінна target є числовою, тому задача була сформульована як задача регресійного прогнозування. Аналіз довжин показав, що мРНК мають значну варіативність: повна довжина послідовності змінювалася від 303 до 17497 нуклеотидів, а середнє значення становило 3899.09 нуклеотида. Це підтвердило доцільність роботи саме з повними послідовностями мРНК, а не лише з окремими ділянками.

Під час попередньої обробки було виконано очищення послідовностей, перевірку target, аналіз CDS-треку та формування функціональних ділянок 5′UTR, CDS і 3′UTR. Після вилучення записів із некоректним CDS-треком у підготовленому наборі залишилося 10275 записів. Дані було розділено на навчальну, валідаційну та тестову вибірки з урахуванням gene-груп. Навчальна вибірка містила 7192 записи, валідаційна – 1541 запис, тестова – 1542 записи. Перетин gene між вибірками був відсутній, що забезпечило більш коректне оцінювання узагальнювальної здатності моделей.

На етапі формування ознак було створено 83 числові структурно-статистичні характеристики. До них увійшли довжинні ознаки, межі CDS, GC-вміст для повної послідовності та окремих функціональних ділянок,

кількість upstream ATG, 3-мерні частоти, MFE-ознаки та службові контрольні характеристики. Аналіз кореляцій і важливості ознак показав, що найбільш інформативними були ознаки, пов’язані з CDS, upstream ATG, локальними MFE-характеристиками та окремими 3-мерними мотивами. Це підтвердило доцільність включення структурно-статистичної модальності до моделі.

Базові моделі машинного навчання використовували лише сформовані числові ознаки. Серед Ridge, RandomForestRegressor і GradientBoostingRegressor найкращий результат показала GradientBoostingRegressor. На тестовій вибірці вона отримала MAE 0.3642, RMSE 0.4745, Pearson 0.7259 і Spearman 0.7180. Це свідчить про те, що сформовані числові ознаки є інформативними, а нелінійні ансамблеві моделі здатні ефективно використовувати їх для прогнозування translation efficiency.

Порівняння моделей глибинного навчання показало перевагу мультимодального підходу над sequence-only моделлю. SequenceOnlyCNN, яка використовувала лише повну нуклеотидну послідовність, досягла на валідаційній вибірці RMSE 0.6646 і Pearson 0.4599. Натомість MultimodalTEModel, яка додатково враховувала регіональну інформацію та 83 числові ознаки, отримала RMSE 0.4816 і Pearson 0.7252. Це показало, що сама послідовність містить корисну інформацію, але її поєднання зі структурно-статистичними ознаками суттєво покращує якість прогнозування.

Після цього було виконано підбір гіперпараметрів мультимодальної моделі. Найкращою стала конфігурація з learning rate 0.0001, weight decay 0.0001, dropout 0.1, batch size 8, Smooth L1 loss, optimizer AdamW, embedding dim 32, region embedding dim 4, conv channels 128 і 64 та numeric hidden dim 128. Найкращий trial досяг на валідаційній вибірці MAE 0.3731, RMSE 0.4733, Pearson 0.7350 і Spearman 0.7229. Цей checkpoint було збережено як фінальну мультимодальну модель.

Фінальне тестове оцінювання показало, що мультимодальна модель зберегла якість на незалежній тестовій вибірці. На test вона отримала MAE 0.3611, RMSE 0.4692, Pearson 0.7308 і Spearman 0.7232. Порівняння з валідаційними результатами показало відсутність різкого перенавчання, оскільки тестові метрики залишилися близькими до validation. Це свідчить про достатню стабільність обраної конфігурації.

Підсумкове порівняння всіх моделей показало, що MultimodalTEModel перевищила найкращу базову модель GradientBoostingRegressor за всіма основними метриками. Порівняно з GradientBoostingRegressor мультимодальна модель зменшила RMSE з 0.4745 до 0.4692, MAE з 0.3642 до 0.3611, а також збільшила Pearson з 0.7259 до 0.7308 і Spearman з 0.7180 до 0.7232. Покращення є помірним, але стабільним за всіма показниками, що підтверджує ефективність запропонованого мультимодального представлення. Важливо, що перевага мультимодальної моделі спостерігалася не за однією окремою метрикою, а одночасно за показниками похибки та кореляції. Це свідчить про більш збалансований характер отриманого результату.

Аналіз похибок показав, що модель зберігає відносно стабільну якість для різних груп довжини повної послідовності та CDS. Водночас було виявлено, що найбільші помилки виникають для крайніх значень target. Для транскриптів із дуже низькою ефективністю трансляції модель має тенденцію завищувати прогноз, а для транскриптів із дуже високою ефективністю трансляції - занижувати його. Це вказує на часткове згладжування крайніх значень і є одним з основних обмежень отриманої моделі.

У розділі також було визначено основні обмеження проведеного експерименту. До них належать використання одного основного набору даних, відсутність повноцінного клітинного контексту, наближене представлення вторинної структури через локальні MFE-ознаки, обмеження згорткової архітектури для моделювання далеких залежностей,

нерівномірність розподілу target і відсутність зовнішньої незалежної валідації. Ці обмеження не скасовують отримані результати, але визначають межі їх застосування. Їх урахування є важливим для коректної інтерпретації результатів і планування подальшого розвитку методу.

Таким чином, у четвертому розділі було експериментально підтверджено працездатність розробленого методу. Отримані результати показали, що поєднання повної послідовності мРНК, регіональної інформації та структурно-статистичних ознак дозволяє підвищити якість прогнозування ефективності трансляції порівняно з базовими моделями машинного навчання та sequence-only нейронною моделлю. Розроблений підхід є відтворюваним, програмно реалізованим і може бути використаний як основа для подальших досліджень у сфері прогнозування трансляційної активності мРНК.

ВИСНОВКИ

У кваліфікаційній роботі розроблено та експериментально перевірено метод мультимодального прогнозування ефективності трансляції мРНК за повною послідовністю на основі глибинного навчання. Для досягнення поставленої мети проаналізовано предметну галузь, формалізовано задачу прогнозування як регресійну, сформовано мультимодальне представлення даних, реалізовано програмний пайплайн, навчено базові та нейронні моделі, виконано підбір гіперпараметрів і фінальне тестове оцінювання.

Аналіз предметної галузі показав, що ефективність трансляції залежить від сукупності послідовнісних, структурних і позиційних факторів. У роботі мРНК подано трьома взаємодоповнювальними модальностями: повною нуклеотидною послідовністю, інформацією про ділянки 5′UTR, CDS і 3′UTR та структурно-статистичними ознаками. До числового представлення включено довжинні характеристики, межі CDS, GC-вміст, кількість upstream ATG, частоти 3-мерів і локальні MFE-ознаки. Загалом сформовано 83 числові ознаки.

Програмну реалізацію виконано у вигляді відтворюваного Python-проєкту. Розроблений пайплайн охоплює завантаження та очищення даних, визначення функціональних ділянок, формування ознак, підготовку даних для PyTorch, навчання моделей, підбір гіперпараметрів, оцінювання якості та збереження результатів. Відтворюваність забезпечено фіксацією seed, використанням конфігураційного файлу та збереженням розбиття даних, метрик, прогнозів, графіків і checkpoint моделей.

Експериментальне дослідження проведено на наборі morrislab/translation-efficiency-human. Після попередньої обробки залишилося 10275 коректних записів. Розбиття на навчальну, валідаційну та тестову вибірки виконано з урахуванням gene-груп, що запобігло потраплянню транскриптів одного гена до різних вибірок.

Серед базових моделей найкращий результат показала GradientBoostingRegressor, яка на тестовій вибірці отримала MAE 0.3642, RMSE 0.4745, Pearson 0.7259 і Spearman 0.7180. Фінальна MultimodalTEModel досягла MAE 0.3611, RMSE 0.4692, Pearson 0.7308 і Spearman 0.7232. Порівняно з найкращою базовою моделлю мультимодальний підхід покращив усі основні метрики. Хоча покращення є помірним, його стабільність підтверджує доцільність поєднання повної послідовності, регіональної інформації та структурно-статистичних характеристик.

Порівняння з SequenceOnlyCNN також показало перевагу мультимодального підходу. Використання лише нуклеотидної послідовності забезпечувало нижчу якість. Аналіз похибок показав, що модель зберігає відносно стабільну якість для транскриптів різної довжини, однак має тенденцію завищувати дуже низькі та занижувати дуже високі значення target.

Наукова новизна роботи полягає в подальшому розвитку методу прогнозування ефективності трансляції мРНК шляхом мультимодального поєднання повної послідовності, функціональних ділянок і структурно-статистичних ознак. Практичне значення полягає у створенні відтворюваного програмного пайплайну, який може бути використаний у подальших біоінформатичних дослідженнях, для попереднього оцінювання трансляційного потенціалу мРНК та в навчальному процесі. Приклади роботи демонстраційного інтерфейсу наведено в додатку А.

Під час підготовки кваліфікаційної роботи інструменти генеративного ШІ використовувалися для пошуку та систематизації джерел інформації, а також для редагування граматики тексту. Змістові рішення, програмна реалізація, проведення експериментів і аналіз результатів виконувалися автором. Таким чином, мету кваліфікаційної роботи досягнуто, а працездатність запропонованого мультимодального методу підтверджено експериментально.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Vogel C., Marcotte E. M. Insights into the regulation of protein abundance from proteomic and transcriptomic analyses. *Nature reviews genetics*. 2012. Vol. 13, no. 4. P. 227–232. URL: https://doi.org/10.1038/nrg3185 (date of access: 15.04.2026).

2. Genome-wide analysis in vivo of translation with nucleotide resolution using ribosome profiling / N. T. Ingolia et al. *Science*. 2009. Vol. 324, no. 5924. P. 218–223. URL: https://doi.org/10.1126/science.1168978 (date of access: 16.04.2026).

3. Current limitations in predicting mRNA translation with deep learning models / N. Schlusser et al. *Genome biology*. 2024. Vol. 25. Art. 227. URL: https://doi.org/10.1186/s13059-024-03369-6 (date of access: 19.04.2026).

4. Cooper G. M. Translation of mRNA. The cell: a molecular approach. 2nd ed. Sunderland (MA) : Sinauer Associates, 2000. URL: https://www.ncbi.nlm.nih.gov/books/NBK9849/ (date of access: 18.04.2026).

5. Blanchet S., Ranjan N. Translation phases in eukaryotes. Ribosome biogenesis: methods and protocols / ed. K.-D. Entian. New York : Humana, 2022. Vol. 2533. P. 217–228. URL: https://doi.org/10.1007/978-1-0716-2501-9_13 (date of access: 20.04.2026).

6. Wilkie G. S., Dickson K. S., Gray N. K. Regulation of mRNA translation by 5′- and 3′-UTR-binding factors. *Trends in biochemical sciences*. 2003. Vol. 28, no. 4. P. 182-188. URL: https://doi.org/10.1016/S0968-0004(03)00051-3 (date of access: 21.04.2026).

7. Mayr C. Regulation by 3′-untranslated regions. *Annual review of genetics*. 2017. Vol. 51. P. 171-194. URL: https://doi.org/10.1146/annurev-genet-120116-024704 (date of access: 22.04.2026).

8. Documentation. *Translation efficiency dynamic database (TEDD)*. URL: https://ngdc.cncb.ac.cn/tedd/documentation/ (date of access: 31.05.2026).

9. Analysis of translation using polysome profiling / H. Chassé et al. *Nucleic acids research*. 2017. Vol. 45, no. 3. P. e15. URL: https://doi.org/10.1093/nar/gkw907 (date of access: 31.05.2026).

10. Kozak M. Point mutations define a sequence flanking the AUG initiator codon that modulates translation by eukaryotic ribosomes. *Cell*. 1986. Vol. 44, no. 2. P. 283–292. URL: https://doi.org/10.1016/0092-8674(86)90762-2 (date of access: 02.06.2026).

11. Human 5′ UTR design and variant effect prediction from a massively parallel translation assay / P. J. Sample et al. *Nature biotechnology*. 2019. Vol. 37, no. 7. P. 803-809. URL: https://doi.org/10.1038/s41587-019-0164-5 (date of access: 02.06.2026).

12. Revealing determinants of translation efficiency via whole-gene codon randomization and machine learning / T. Nieuwkoop et al. *Nucleic acids research*. 2023. Vol. 51, no. 5. P. 2363–2376. URL: https://doi.org/10.1093/nar/gkad035 (date of access: 02.06.2026).

13. ViennaRNA package 2.0 / R. Lorenz et al. *Algorithms for molecular biology*. 2011. Vol. 6. Art. 26. URL: https://doi.org/10.1186/1748-7188-6-26 (date of access: 02.06.2026).

14. Optimizing 5′UTRs for mRNA-delivered gene editing using deep learning / S. Castillo-Hair et al. *Nature communications*. 2024. Vol. 15. Art. 5284. URL: https://doi.org/10.1038/s41467-024-49508-2 (date of access: 02.06.2026).

15. Predicting the translation efficiency of messenger RNA in mammalian cells / D. Zheng et al. *Nature biotechnology*. 2026. Vol. 44, no. 5. P. 783-796. URL: https://doi.org/10.1038/s41587-025-02712-x (date of access: 02.06.2026).

16. Full-length ribosome density prediction by a multi-input and multi-output model / T. Tian et al. *PLOS computational biology*. 2021. Vol. 17, no. 3. Art. e1008842. URL: https://doi.org/10.1371/journal.pcbi.1008842 (date of access: 02.06.2026).

17. Бібічков І. Є., Сокол В. В., Шевченко О. Ю. Розробка методу генерації веб-контенту на основі онтологічного підходу. *Technology audit*

and production reserves. 2019. Т. 5, № 2(49). С. 4–10. URL: https://doi.org/10.15587/2312-8372.2019.183201 (дата звернення: 03.06.2026).

18. Baltrušaitis T., Ahuja C., Morency L.-P. Multimodal machine learning: a survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*. 2019. Vol. 41, no. 2. P. 423–443. URL: https://doi.org/10.1109/TPAMI.2018.2798607 (date of access: 03.06.2026).

19. Kingma D. P., Ba J. Adam: a method for stochastic optimization. *arXiv*. 2014. URL: https://arxiv.org/abs/1412.6980 (date of access: 03.06.2026).

20. Hodson T. O. Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not. *Geoscientific model development*. 2022. Vol. 15, P. 5481–5487. URL: https://doi.org/10.5194/gmd-15-5481-2022 (date of access: 03.06.2026).

21. mRNABench: a curated benchmark for mature mRNA property and function prediction / R. I. Shi et al. *bioRxiv*. 2025. URL: https://doi.org/10.1101/2025.07.05.662870 (date of access: 16.06.2026).

22. Hugging Face. Load. *Hugging Face datasets documentation*. URL: https://huggingface.co/docs/datasets/en/loading (date of access: 16.06.2026).

23. The pandas development team. pandas.read_parquet. *pandas API reference*. URL: https://pandas.pydata.org/docs/reference/api/pandas.read_parquet.html (date of access: 16.06.2026).

24. scikit-learn developers. sklearn.model_selection.GroupShuffleSplit. *scikit-learn API reference*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GroupShuffleSplit.html (date of access: 16.06.2026).

25. PyTorch Contributors. torch.utils.data. *PyTorch documentation*. URL: https://docs.pytorch.org/docs/stable/data.html (date of access: 16.06.2026).

26. PyTorch Contributors. SmoothL1Loss. *PyTorch documentation*. URL: https://docs.pytorch.org/docs/stable/generated/torch.nn.SmoothL1Loss.html (date of access: 16.06.2026).

27. PyTorch Contributors. ReduceLROnPlateau. *PyTorch documentation*. URL: https://docs.pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html (date of access: 16.06.2026).

28. PyTorch Contributors. Save and load the model. *PyTorch tutorials*. URL: https://docs.pytorch.org/tutorials/beginner/basics/saveloadrun_tutorial.html (date of access: 16.06.2026).