Київський національний університет імені Тараса Шевченка
Факультет комп'ютерних наук та кібернетики
Кафедра моделювання складних систем

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА
на тему:

Порівняльний аналіз рекурентних алгоритмів для прогнозування цін на фондовому ринку

студента 4 курсу кафедри МСС
ЗАВОРОТИНСЬКОГО Максиміліана

Науковий керівник:
доцент, кандидат технічних наук
Віктор КУЛЯН

Робота заслухана на засіданні кафедри моделювання складних систем та рекомендована до захисту в ЕК, протокол № 10 від 05.06.2023 р.

Завідувач кафедри МСС Дмитро ЧЕРНІЙ

Київ – 2023

АНОТАЦІЯ

Кваліфікаційна робота бакалавра: 51 сторінка, 13 рисунків, 3 таблиці, 2 додатки, 15 інформаційних джерел.

Актуальність роботи: Фондові ринки відіграють ключову роль у глобальній економіці, надаючи широкі можливості для інвестицій, торгівлі та розробки фінансових стратегій. Однією з найбільш важливих прикладих задач на цих ринках є прогнозування цін активів, включаючи акції, облігації та інші цінні папери. Точність прогнозів цін на фондовому ринку має визначальне значення для інвесторів та фінансових установ, оскільки від цього залежать прийняття рішень та отримання прибутку.

Об’єкт дослідження: Застосування рекурентних алгоритмів комп'ютерного моделювання для прогнозування цін на фондовому ринку.

Мета роботи: Проведення порівняльного аналізу рекурентних алгоритмів для прогнозування цін на фондовому ринку, а саме фільтра Калмана на основі авторегресійної моделі та рекурентних нейронних мереж.

Предмет дослідження: Фільтр Калмана, авторегресійна модель та рекурентна нейронна мережа, як моделі для прогнозування цін на фондовому ринку.

Ключові слова: ФОНДОВИЙ РИНОК, РЕКУРЕНТНІ НЕЙРОННІ МЕРЕЖІ, LSTM, ФІЛЬТР КАЛМАНА, АВТОРЕГРЕСІЙНА МОДЕЛЬ.

ЗМІСТ

ВСТУП

У сучасному світі фондові ринки відіграють ключову роль у світовій економіці, надаючи можливості для інвестицій, торгівлі та створення фінансових стратегій. Одним із найважливіших аспектів фондових ринків є прогнозування цін на активи, такі як акції, облігації та інші цінні папери. Точність прогнозування цін на фондовому ринку має велике значення для інвесторів та фінансових установ, оскільки від цього залежить прийняття рішень та отримання прибутку.

З часом з'являються нові методи та алгоритми, що дозволяють підвищити точність прогнозування цін на фондовому ринку. Однією з сучасних тенденцій є використання рекурентних алгоритмів, які здатні враховувати залежності та послідовності даних, що особливо важливо під час аналізу фінансових часових рядів.

Мета цього дослідження полягає у проведенні порівняльного аналізу рекурентних алгоритмів для прогнозування цін на фондовому ринку. В рамках дослідження розглянуто два основні методи прогнозування: фільтр Калмана з використанням авторегресійної моделі, а також рекурентні нейронні мережі з довгою короткостроковою пам'яттю. Рекурентна нейронна мережа була обрана, оскільки є широко застосовуваним та актуальним підходом до прогнозування фінансових часових рядів. Фільтр Калмана є більш простим методом, що може бути корисно в деяких умовах, але цей алгоритм використовується не так часто в цій галузі, тому є потреба в порівняльному аналізі його точності, швидкості та інших параметрів.

Актуальність цього дослідження зумовлена необхідністю оцінки точності прогнозування цін на фондовому ринку, щоб забезпечити успішні фінансові рішення та мінімізувати ризики. Це має особливе значення в умовах високої конкуренції на фінансових ринків і економічних умов, що

швидко змінюються. Точніші прогнози цін дозволять інвесторам приймати рішення, засновані на більш надійних прогнозах майбутніх цінових рухів.

Крім того, це дослідження може мати практичну значущість для певних осіб або установ, які прагнуть автоматизувати процеси прогнозування та оптимізувати свої торгові стратегії. Успішне застосування рекурентних алгоритмів, які будуть кращими за інші в певних умовах, може призвести до покращення результатів у галузі фінансового прогнозування і збільшення прибутку та конкурентоспроможності.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ РЕКУРЕНТНИХ АЛГОРИТМІВ ТА ЇХ ЗАСТОСУВАННЯ У ПРОГНОЗУВАННІ ЦІН НА ФОНДОВОМУ РИНКУ

1.1. Постановка завдання та методи прогнозування цін на фондовому ринку

Постановка завдання та цілі дослідження.

Розглянемо портфель інвестицій, що складається з N акцій. Загальна задача оптимізації ризикованих інвестицій є двокритеріальною. Одним із критеріїв є максимізація очікуваного прибутку, а іншим – мінімізація ризику. Одним із методів розв'язання такої задачі є розбиття її на дві незалежні однокритеріальні задачі. Одна з таких задач полягає у максимізації очікуваного прибутку для заданого рівня ризику, а друга - у мінімізації ризику портфеля для заданого рівня очікуваного прибутку. Обидві задачі є задачами нелінійного програмування і для їх аналітичного розв'язання можна застосувати відповідні методи. Розглянемо другу постановку у формулюванні Г. Марковиця:

$$\begin{cases} \sum_{i=1}^{N} x_i r_i \to \max\limits_{x}, \\ \sum_{i=1}^{N}\sum_{j=1}^{N} x_i x_j \sigma_{ij} = \tau, \\ \sum_{i=1}^{N} x_i = 1, \\ x_i \geq 0, \end{cases}$$

де x_i – частка акцій і-го типу у портфелі, r_i – очікувана прибутковість акцій і-ого типу, $\sigma_{ij} = p_{ij}\sigma_i\sigma_j$ – коваріація між i-м та j-м цінним папером, $\rho_{ij} \in [-1, 1]$ – коефіцієнт кореляції, τ – фіксований рівень ризику. Для розв'язання цієї задачі необхідно побудувати прогноз для отримання очікуваних прибутковостей акцій. Тому основне завдання цього дослідження полягає у виконанні порівняльного аналізу ефективності прогнозування цін акцій з використанням фільтра Калмана з авторегресійною моделлю та рекурентних нейронних мереж з довгою коротко строковою пам'яттю (LSTM). Для досягнення цієї мети ставляться такі задачі:

- дослідити та описати теоретичні основи побудови та застосування фільтра Калмана, авторегресійної моделі та рекурентних нейронних мереж для прогнозування часових рядів;

- розробити ефективні моделі прогнозування цін акцій на основі фільтра Калмана та рекурентних нейронних мереж, використовуючи обрані метрики;

- провести обчислювальні експерименти, щоб порівняти ефективність прогнозування за допомогою розроблених моделей;

- проаналізувати результати експериментів та зробити висновки про переваги та недоліки кожного підходу.

У рамках даного дослідження зосередимося на використанні лише цін закриття акцій для побудови моделей прогнозування. Це рішення обумовлено такими причинами:

- ціна закриття акцій є найбільш важливим та інформативним показником, оскільки вона відображає кінцевий результат торгового дня та більшою мірою враховує загальну тенденцію ринку;
- спрощення моделей: використання тільки цін закриття спрощує моделі прогнозування, що сприяє зниженню ризику перенавчання та прискорення навчання;

- порівнянність результатів: обмеження даних до одного показника (ціни закриття) дозволяє більш об'єктивно порівнювати результати на основі моделей, заснованих на фільтрі Калмана та рекурентних нейромереж, оскільки вплив інших факторів буде виключено.

Але слід зазначити, що в реальних умовах торгівлі на фондовому ринку можуть бути використані додаткові показники, такі як ціни відкриття, максимальні та мінімальні ціни, обсяги торгів та інші фінансові показники. Однак у цьому дослідженні вони виключені з розгляду з метою зосередитись на порівняльному аналізі ефективності прогнозування з використанням фільтра Калмана та рекурентних нейронних мереж.

У ході експерименту використано історичні дані про ціни закриття акцій, обраних компаній на певний період часу. Ці дані будуть поділені на навчальну, валідаційну та тестову вибірки для побудови та оцінювання моделей прогнозування.

Фільтр Калмана та рекурентні нейромережі як рекурентні алгоритми.

Фільтр Калмана та рекурентні нейромережі обрано у даному дослідженні, оскільки вони є найбільш ефективними та поширеними рекурентними алгоритмами для прогнозування часових рядів. Фільтр Калмана заснований на алгоритмі побудови оптимальної оцінки стану динамічної системи шляхом послідовного коригування прогнозів та оцінок помилок. Основними перевагами фільтра Калмана є його здатність адаптуватися до змін у даних, швидке навчання та невелика кількість параметрів.

Рекурентні нейронні мережі мають властивість "пам'яті", яка дозволяє зберігати інформацію про попередні стани та враховувати їх при прогнозуванні. LSTM (Long Short-Term Memory, з англ. довга короткострокова пам’ять) – це різновид рекурентних нейронних мереж, розроблений для вирішення проблеми згасання градієнтів, властивої

традиційним рекурентних нейронним мережам. LSTM характеризується складною архітектурою, що складається з вентилів, що контролюють потік інформації. Це дозволяє LSTM навчатися на довгих послідовностях та вловлювати довгострокові залежності між даними. LSTM успішно застосовується для прогнозування цін на фондовому ринку завдяки своїй здатності враховувати складні залежності та адаптуватися до змін.

Порівняння фільтра Калмана та нейромереж між собою повинно виявити переваги, недоліки, галузь можливого застосування та доцільність використання. Фільтр Калмана повинен забезпечити швидку та адаптивну оцінку стану системи з невеликою кількістю параметрів, у той час як LSTM здатна виявляти складні залежності та адаптуватися до змін на довгих послідовностях даних. Таким чином, дослідження ефективності прогнозування цін акцій із використанням цих двох алгоритмів дозволить оптимізувати процедуру прийняття рішень при інвестуванні у ризиковані цінні папери.

1.2. Фільтр Калмана та авторегресійна модель: принципи побудови та застосування при прогнозуванні

Фільтр Калмана: теоретичні основи та алгоритм роботи

Фільтр Калмана – це алгоритм побудови оптимальної оцінки стану динамічної системи на основі послідовних вимірів із шумами. Він використовується для оцінювання стану систем, які не можуть бути безпосередньо спостережними і для уточнення оцінок стану, отриманих з неповних або даних з шумом. Фільтр Калмана часто використовують у таких галузях, як навігація, автоматичне керування, обробка сигналів та інше. У роботі наводяться теоретичні відомості про фільтр, а також адаптація його під прогнозування цін акцій та порівняння ефективності з іншими рекурентними алгоритмами.

Далі в процесі опису алгоритму використовуються поняття шуму та фільтра. Під шумом розуміють випадкові, чи умовно випадкові коливання, які можуть бути викликані різноманітними факторами та мати різну природу. Під фільтром розуміють алгоритм обробки вхідних даних, який повинен видаляти шуми.

Фільтр Калмана потребує задати початкові значення, вони можуть бути стандартними, або є можливість задати апріорну інформацію про характер спостережуваної системи, що може покращити прогнозування моделі [1]. Цими початковими даними є:

F – матриця процесу (матриця моделі), яка описує динамічну систему (тобто таку систему стан якої змінюється з часом). Саме за допомогою цієї моделі фільтр Калмана дозволяє прогнозувати стан системи у наступний момент часу та видаляти шуми, які не спостерігаються напряму. Фільтр Калмана також має певні обмеження щодо використовуваних моделей. Ці моделі повинні бути дискретні та лінійні. Оскільки розглядається задача моделювання динаміки цінних паперів на фондових ринках і існує ціна, яка спостерігається за певний період часу дискретно, то необхідно також розглядати дискретну модель.

H – матриця спостережень [1]. У найбільш простому випадку є лише одна змінна, яку ми спостерігаємо, це вартість закриття акції.

Q – коваріаційна матриця похибки моделі [1]. Відображає похибку моделі, на основі якої побудовано прогноз стану, використовуючи реальні значення.

R – коваріаційна матриця похибки вимірювань [1]. У випадку коли розглядається одна спостережувана величина матриця має розмірність 1x1. Чим ближче значення до 0, тим більше впевненості у правильності значень вимірювань, у випадку спостережень цін акцій, значення буде близьким до нуля.

x – початковий стан [1], який будемо брати з конкретного набору даних.

P – коваріаційна матриця для початкового стану [1]. Як і у випадку з R, тим ближче до нуля, чим більше впевненості у правильності обраного початкового стану.

Також у стандартному фільтрі Калмана є такі змінні, як B – матриця застосування керуючого впливу, u – керуючий вплив. Проте, оскільки керуючого впливу у нашої моделі немає, то $B = 0$. Тому далі у розрахунках ці дві змінні будуть відкинуті.

Фільтр Калмана використовує динамічну модель системи, зазвичай це є якийсь механічний закон руху, але не в нашому випадку. Наша динамічна модель не є точною, в той час як спостереження є точними. Алгоритм фільтра Калмана складається з двох фаз: прогнозування і коригування. На першому етапі ми робимо прогноз стану системи на наступний момент часу з урахуванням неточності нашої моделі, а також прогнозуємо похибку коваріації, тобто робимо такі дії:

$$x_i = F * x_{i-1}, \tag{1.1}$$

$$P_i = F * P_{i-1} * F^T + Q. \tag{1.2}$$

Формула (1.1) дозволяє обчислити наступний стан системи, де x_i – новий стан системи, x_{i-1} – попередній стан системи, F – матриця процесу (матриця динамічної моделі системи). Формула (1.2) дозволяє обчислити наступну похибку коваріації, де P_i – прогнозоване значення похибки, P_{i-1} – похибка в попередній момент часу, Q – коваріаційна матриця похибки моделі.

На другому етапі нова інформація, отримана в результаті спостереження, коригує прогнозоване значення з урахуванням шуму і неточності цієї інформації, це відбувається наступним чином:

$$K_i = P_i * H^T * \left(H * P_i * H^T + R\right)^{-1}, \quad (1.3)$$

$$x_i = x_i + K_i(z_i - H * x_i), \quad (1.4)$$

$$P_i = (I - K_i * H) * P_i. \quad (1.5)$$

Формула (1.3) дає можливість обчислити посилення Калмана (Kalman Gain) [1], яке необхідно для наступних обчислень. H та R це матриця спостережень та коваріаційна матриця похибки вимірювань відповідно. Формула (1.4) використовує значення, що отримано на попередньому кроці. На цьому етапі коригується оцінка стану з урахуванням нових спостережень (позначаються z_i). Формула (1.5) коригує похибку коваріації, де I – одинична матриця.

Також існує модифікація цього алгоритму, яка називається розширений фільтр Калмана. В даній роботі він не розглядається, оскільки береться лінійний випадок авторегресійної системи, і не має потреби в ускладнені основного алгоритму, проте це може бути корисне для подальшого розвинення теми та покращення прогнозу за рахунок побудови більш складної системи досліджень.

Авторегресійна модель: визначення та властивості

Авторегресійна модель (AR) є одним із ключових інструментів у часових рядах для аналізу та прогнозування. AR модель використовує власні попередні значення часового ряду для прогнозування майбутніх значень. Визначення авторегресійної моделі n-го порядку можна записати так:

$$Y_t = c + \Phi_1 * Y_{t-1} + \Phi_2 * Y_{t-2} + \ldots + \Phi_n * Y_{t-n} + \epsilon_t \quad (1.6)$$

де Y_t – значення часового ряду в момент часу t, c – вільний член і константа, Φ_i – коефіцієнт авторегресії, ϵ_t – випадкова величина. Характерною особливістю даної авторегресійної моделі є її лінійність, а отже її можна буде використовувати у фільтрі Калмана, про що буде детальніше написано у наступному пункті. Для побудови оцінки параметрів авторегресійної моделі

(коефіцієнтів авторегресії та константи c) можна використовувати різні методи, такі як метод найменших квадратів (МНК), максимальна правдоподібність [8] та інформаційний критерій Акаїки (AIC) [6] або інформаційний критерій Байєса (BIC) [7].

Метод найменших квадратів (МНК): Цей метод мінімізує суму квадратів різниць між фактичними та прогнозованими значеннями часового ряду. МНК є одним із найбільш популярних методів оцінювання параметрів у лінійних моделях.

Максимальна правдоподібність: Цей метод дозволяє визначати параметри, що максимізують функцію правдоподібності, яка дає можливість оцінити ймовірність отримання спостережених даних при заданих параметрах моделі. Метод максимальної правдоподібності має деякі переваги порівняно з МНК, такі як здатність оцінювати параметри більш складних моделей.

Інформаційний критерій Байєса (BIC): BIC є критерієм для вибору оптимальної моделі з урахуванням її складності та здатності описати дані. Чим нижче значення BIC, тим краща модель. BIC допомагає уникнути перенавчання моделі. У задачі, яка поставлена в цій роботі, буде використовуватися AIC або BIC, обґрунтування та теоретичні відомості будуть детальніше написані у параграфі 2.1.

Також для авторегресійної моделі важливо, щоб часовий ряд з яким вона буде працювати був стаціонарний. Стаціонарність це така властивість ряду, яка вказує на стабільність його статистичних характеристик (середнє значення, дисперсія, автокореляція) протягом часу [2] і це сприяє більш точному прогнозуванню. Для перевірки ряду на стаціонарність буде використовуватися тест Дікі-Фуллера. У цьому тесті ми маємо перевірити нульову гіпотезу [3]. Цей тест проводиться для авторегресійної моделі першого порядку, виду:

$$Y_t = c + \Phi_1 * Y_{t-1} + \epsilon_t. \quad (1.7)$$

У формулі (1.7) перевіряється значення коефіцієнта Φ_1. Якщо $\Phi_1 = 1$, то процес має одиничний корінь, в такому випадку ряд Y_t не стаціонарний. Якщо $|\Phi_1| < 1$, то ряд стаціонарний. Це перевіряється так: спочатку переписують рівняння (1.7) у вигляд (1.8).

$$\Delta Y_t = b * Y_{t-1} + \epsilon_t, \quad (1.8)$$

де $\Delta Y_t = Y_t - Y_{t-1}$, а $b = \Phi_1 - 1$. У вигляді (1.8) нульова гіпотеза приймає вид $b = 0$. Для перевірки цієї гіпотези використовується t-статистика [3], яка порівнюється з критичним значенням обраного рівня значимості. Детальніше про цю статистику можна дізнатися за джерелом [3]. Також варто відмітити, що існує розширений тест Дікі-Фуллера [3], який дозволяє проводити тест для авторегресійних моделей вищого порядку.

Після проведення тесту, якщо виявилось, що ряд не стаціонарний то його потрібно перетворити на стаціонарний. Для перетворення нестаціонарних часових рядів у стаціонарні можна використовувати різні методи, такі як диференціювання чи логарифмування. У цій роботі буде використовуватися логарифмування ряду, а саме - перетворення Бокса-Кокса [10]. Перетворення має такий вигляд:

$$y = \begin{cases} \frac{y^\lambda - 1}{\lambda}, \ \lambda = 0 \\ \ln(y), \ \lambda \neq 0 \end{cases} \quad (1.9)$$

і це накладає певні обмеження на у. Тобто при $\lambda \neq 0$, ряд y має бути додатнім.

Комбінований підхід з використанням авторегресійної моделі та фільтра Калмана

Для застосування авторегресійної моделі у фільтрі Калмана потрібно побудувати таку матрицю процесу (F), щоб вона відповідала формулі (1.6). Для цього спочатку необхідно визначити яким буде стан системи x. Він має

включати n-коефіцієнтів авторегресії з рівняння (1.6) та вільний член c. Тому в цьому випадку стан системи буде мати вигляд:

$$x_t = (Y_t, Y_{t-1}, ..., Y_{t-n}, 1).$$

Також важлива і матриця спостережень H, у випадку який буде розглядатися у цій роботі, тобто спостереження за однією величиною матриця буде мати такий вигляд:

$$H = (1, 0, ..0, 0).$$

Це значить що ми спостерігаємо лише за змінною Y_t. Враховуючи це, можна побудувати матрицю F і вона матиме вигляд:

$$F = \begin{pmatrix} \Phi_1 & \Phi_2 & \cdots & \Phi_n & c \\ 1 & 0 & ... & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & ... & 0 & 1 \end{pmatrix}.$$

Далі потрібно визначити матрицю похибки моделі Q. У даному випадку будемо вважати що, похибка розподіляється порівну між кожним вимірюванням. А отже, матриця набуде такого виду:

$$Q = \frac{\sigma}{n} * \begin{pmatrix} 1 & 0 & ... & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

де n – порядок авторегресійної моделі, σ – дисперсія між прогнозованими значеннями та реальними. Множення відбувається на одиничну матрицю, але останній рядок нульовий, оскільки він відповідає за вільний член, згідно нашої моделі, в якому немає похибки.

1.3. Рекурентні нейронні мережі та довга короткострокова пам'ять: основи, архітектура та застосування при прогнозуванні

Основи та архітектура рекурентних нейронних мереж.

Рекурентні нейронні мережі – це клас штучних нейронних мереж, що мають здатність обробляти послідовні дані, такі як часові ряди або текст. Основна особливість рекурентних нейронних мереж полягає в наявності рекурентних зв'язків, що дозволяють передавати інформацію між прихованими станами на різних кроках. Основна структура рекурентних нейронних мереж складається з трьох шарів: вхідного шару, прихованого шару та вихідного шару. Вхідний шар є матрицею вхідних даних X, де кожен рядок відповідає одному тимчасовому кроку, а стовпці – ознакам. Прихований шар має рекурентні зв'язки, які забезпечують передачу інформації між прихованими станами на різних кроках. Вихідний шар передбачає цільові значення Y на основі прихованих станів.

Зворотне поширення помилки в часі (BPTT) – це алгоритм навчання рекурентних нейронних мереж, який ґрунтується на зворотному розповсюдженні помилки для звичайних нейронних мереж [12]. У BPTT обчислюються градієнти помилки по відношенню до ваги мережі на кожному тимчасовому кроці, після чого виконується оновлення ваг.

Для рекурентного шару обчислення прихованого стану h на тимчасовому кроці t здійснюється за допомогою наступної формули:

$$h_t = f\left(W_x * x_t + W_h * h_{t-1} + b_h\right),$$

де f – функція активації, W_x – вагова матриця , x_t – вхідні дані, W_h - вагова матриця для рекурентних зв’язків, h_{t-1} – попередній прихований стан, b_h - зсув для прихованого шару. У процесі навчання рекурентних нейронних

мереж з використанням BPTT можуть виникнути проблеми із загасаючими та вибуховими градієнтами. Загасаючі градієнти виникають, коли градієнти помилки стають занадто маленькими, що може призвести до недонавчання. Вибухові градієнти виникають, коли градієнти помилки стають занадто великими, що може призвести до нестабільності навчання. Ці проблеми пов'язані з довгостроковими залежностями у послідовних даних.

Довга короткострокова пам'ять та її архітектура.

LSTM – це різновид рекурентних нейронних мереж, розроблений спеціально для боротьби з проблемами загасаючих та вибухових градієнтів [11]. LSTM є архітектурою нейронної мережі з використанням спеціальних модулів пам'яті, які дозволяють зберігати та передавати інформацію на великі відстані в часі.

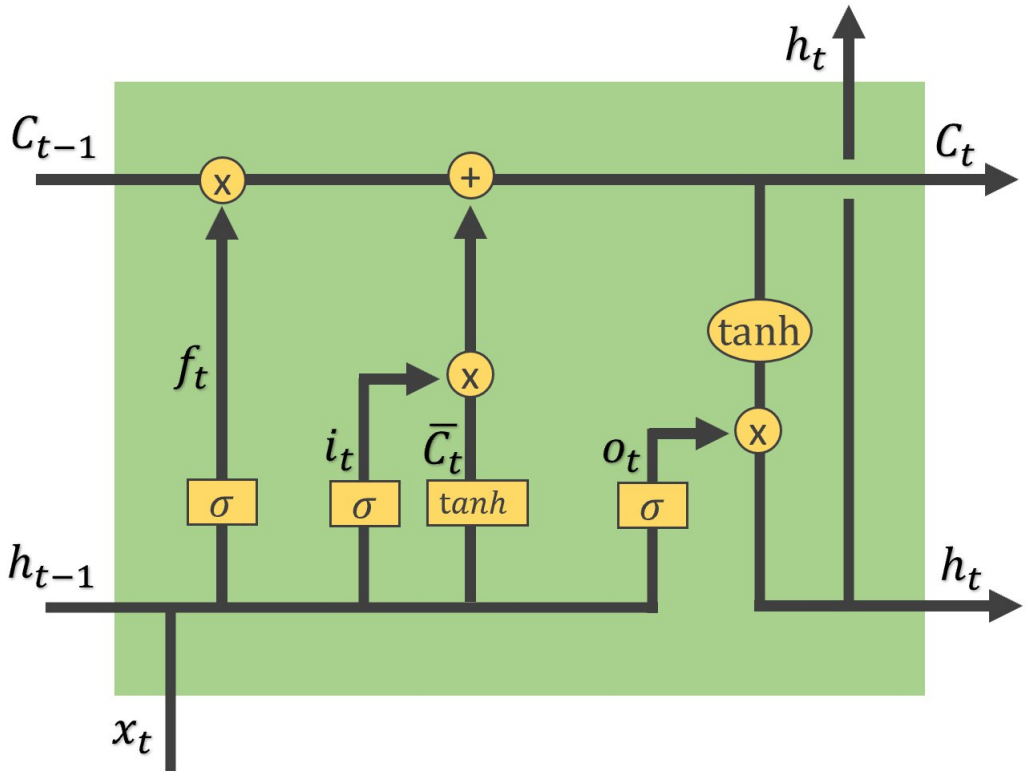


Рис. 1.1

Алгоритм роботи LSTM-комірки можна графічно відобразити на схемі (рисунок 1.1). Далі буде детально описано процес, що відображено на схемі,

де x_t – вхідний вектор в момент t, h_t – вектор прихованого стану (hidden state) на кроці t, C_t – комірка стану (cell state), $\overline{C_t}$ – нові значення-кандидати до комірки C_t. Також на схемі ще є f_t, i_t, o_t. Їх часто називають вентилями і вони є важливими елементами, тому нижче наведено описання кожного з них:

1. Вентиль забування (forget gate) – регулює рівень збереження інформації з попереднього стану осередку.
2. Вентиль входу (input gate) – визначає, яка інформація з поточного вхідного значення та прихованого стану має бути додана в комірку стану.
3. Вентиль виходу (output gate) – визначає, яка інформація з комірки стану має бути передана на наступний тимчасовий крок.

У LSTM-мережі кожен вентиль використовує сигмоїдальну або гіперболічну функцію активації та свій власний набір параметрів (ваг та зсувів). Вони позначаються відповідно σ та $tanh$. Також на рисунку (1.1) присутні кружечки в яких знаходяться знаки x, +, tanh вони означають поелементне відповідно множення, додавання та обрахування гіперболічного тангенсу.

Варто відмітити, що стрілочкою на цій схемі позначається напрям руху вектора, коли дві лінії з'єднуються це означає поєднання, тобто конкатенацію двох векторів, якщо пряма розходиться – це означає копіювання вектора.

Далі буде детально описано кожен крок роботи схеми LSTM-комірки [11].

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right). \tag{1.10}$$

Вентиль забування, який рахується за формулою (1.10), визначає, яка інформація із попереднього стану осередку буде "забута" при переході до поточного стану. Тут σ позначає сигмоїдальну функцію активації, W_f - ваги вентиля забування, $[h_{t-1},x_t]$ - конкатенація попереднього прихованого стану h_{t-1} і вхідного вектора x_t, а b_f - зміщення вентиля забування.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right), \tag{1.11}$$

$$\overline{C_t} = \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right). \tag{1.12}$$

Вхідний вентиль, у формулі (1.11), визначає, яку нову інформацію слід додати до стану комірки. Відповідно, W_i та b_i - ваги та зміщення вхідного вентиля. Оновлення стану визначається за формулою (1.12) - де W_C і b_C - ваги і зміщення цього вентиля.

Поточний стан комірки (1.13) обчислюється шляхом комбінування інформації від вентиля забування f_t та вхідного вентиля i_t.

$$C_t = f_t * C_{t-1} + i_t * \overline{C_t}. \tag{1.13}$$

Вентиль забування регулює ступінь збереження попереднього стану комірки C_{t-1}, а вхідний вентиль визначає, яку нову інформацію слід інтегрувати у стан комірки. Це досягається шляхом по елементного множення f_t C_{t-1} і i_t на $\overline{C_t}$, після чого результати складаються для отримання поточного стану осередку C_t.

Вихідний вентиль (1.14) визначає, яка інформація з поточного стану осередку C_t передаватиметься на вихід у вигляді поточного прихованого стану h_t, у рівнянні (1.15).

$$o_t = \sigma\left(W_o\,[h_{t-1},x_t] + b_o\right), \tag{1.14}$$

$$h_t = o_t * \tanh\left(C_t\right). \tag{1.15}$$

Тут W_o та b_o позначають ваги та зміщення вихідного вентиля відповідно. Вихідний вентиль обчислюється з використанням сигмоїдальної функції активації, а прихований поточний стан h_t виходить шляхом поелементного множення вихідного вентиля o_t на гіперболічний тангенс поточного стану осередку $tanh\left(C_t\right)$.

Модифікації LSTM та альтернативні архітектури.

Вентильний рекурентний вузол (з англ. Gated Recurrent Unit, далі GRU) – це альтернативна архітектура LSTM, яка також вирішує проблему загасаючих градієнтів [13]. GRU використовує меншу кількість вентилів та параметрів, що робить його більш ефективним для обчислень та потребує менше зусиль для навчання, однак це може призвести до деяких обмежень у якості моделі, коли даних достатньо. Загальний алгоритм схожий на описаний раніше LSTM, проте має деякі зміни. Загалом, його можна описати алгоритм наступним чином:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z),$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r),$$

$$\tilde{h_t} = \tanh(W_{xh} \cdot x_t + W_{hh} \cdot (r_t \otimes h_{t-1}) + b_h),$$

$$h_t = h_{t-1} \otimes (1 - z_t) + \tilde{h_t} \otimes z_t,$$

де z_t – аналог f_t, r_t – відповідає за забування «знань», W_z, W_r W_h - ваги, b_z, b_r b_h - зсуви, $\tilde{h_t}$ – відповідає за додавання нових «знань» до комірки.

Інші модифікації LSTM і GRU включають Peephole LSTM, де вентилі можуть "зазирнути" в комірку стану, та Layer Normalization LSTM, де застосовується нормалізація шару для стабілізації навчання.

Отже, підводячи підсумки, LSTM дає можливість розв’язати проблему загасаючих градієнтів за рахунок використання осередків стану та вентилів. Це дозволяє моделі передавати інформацію на великі інтервали часу без значного згасання градієнтів. В результаті LSTM може навчатися на даних із довгостроковими залежностями, що складно для традиційних рекурентних нейронних мереж. Також існує більш проста та швидка GRU, але вона може давати гірші результати, тому саме LSTM буде використовуватися для розв’язання поставленої задачі.

РОЗДІЛ 2. МЕТОДОЛОГІЯ ПОРІВНЯЛЬНОГО АНАЛІЗУ ТА РОЗРОБКА МОДЕЛЕЙ ПРОГНОЗУВАННЯ

2.1. Методи оцінювання ефективності та точності прогнозних моделей

Введення у метрики оцінювання прогнозних моделей.

У даній роботі з порівняльного аналізу важливим аспектом є побудова оцінки ефективності та точності прогнозних моделей. Це допомагає визначити яка з моделей краще справляється із задачею та оцінити, наскільки точні та корисні отримані прогнози. Також під час розробки та використання моделей для прогнозування даних важливо мати засіб для оцінювання якості цих моделей. У роботі це буде застосовуватися для оцінювання якості авторегресійних моделей. Побудова оцінки якості моделей дозволяє: визначити, яка з моделей працює найкраще в умовах даної задачі; виявити слабкі сторони моделей та ввести відповідні зміни; забезпечити порівняння між різними моделями та алгоритмами; оцінити ступінь довіри до отриманих результатів. Останніми буде розглянуто метрики для оцінювання якості побудови моделей машинного навчання, які допомагають знайти найкращу модель та уникнути типових проблем, такі як перенавчання або недонавчання.

Метрики для побудови оцінки точності прогнозів.

Далі буде детально розглянуто метрики для оцінки точності прогнозів побудованих моделей, які були обрані як основні.

Середня абсолютна помилка визначається як середнє значення абсолютних різниць між прогнозами моделі і фактичними значеннями [4]. Має місце такий вираз:

$$MAE = \frac{1}{n} * \sum_{i=1}^{n} |y_i - \overline{y_i}|,$$

де n – кількість спостережень, y_i – фактичні значення і-ого спостереження, $\overline{y_i}$ – прогнозоване значення і-ого спостереження. Середня абсолютна помилка є метрикою, що вимірює середнє відхилення прогнозованих значень від фактичних. Вона є середньою величиною помилки без урахування знака, що дозволяє інтерпретувати результат у термінах абсолютних значень. Тобто MAE може дорівнювати нулю або позитивному числу, де нуль означає ідеальний збіг прогнозних і фактичних значень, а великі значення вказують на велику помилку моделі. Тобто, можна вважати, що чим ближче значення до нуля, тим кращий прогноз дозволяє отримати модель.

Середньоквадратична помилка (MSE) визначається як середнє значення квадратів різниць між прогнозами моделі та фактичними значеннями [4]

$$MSE = \frac{1}{n} * \sum_{i=1}^{n} (y_i - \overline{y_i})^2,$$

де n – кількість спостережень, y_i – фактичні значення і-ого спостереження, $\overline{y_i}$ – прогнозоване значення і-ого спостереження. Середньоквадратична помилка є метрикою, що вимірює середнє відхилення прогнозних значень від фактичних у квадратичному вигляді. Це означає, що більші помилки мають більшу вагу, ніж менші помилки [15]. MSE завжди невід'ємна, і значення, що дорівнює нулю, вказує на ідеальний збіг прогнозних і фактичних значень.

Також для більш простої інтерпретації результатів використовують метрику RMSE [4], це корінь із середньоквадратичної помилки. Розраховується він так:

$$RMSE = \sqrt{\frac{1}{n} * \sum_{y=1}^{n} (y_i - \overline{y_i})^2} .$$

RMSE також дає невід'ємні значення, і чим ближче вони до нуля, тим краще модель прогнозує.

Середня абсолютна відсоткова помилка (MAPE) визначається як середнє значення абсолютних відсоткових різниць між прогнозами моделі та фактичними значеннями [4] і розраховується за такою формулою:

$$MAPE = \frac{1}{n} * \sum_{i=1}^{n} \left| \frac{y_i - \overline{y_i}}{y_i} \right| * 100.$$

Позначення відповідають стандартним, що були описані раніше. Середня абсолютна відсоткова помилка є метрикою, яка вимірює середнє відхилення прогнозних значень від фактичних у відсотковому вираженні. Це означає, що MAPE дозволяє оцінити помилки моделі як відсотки від фактичних значень, що полегшує порівняння між різними наборами даних і різними одиницями виміру. MAPE завжди дорівнює позитивному числу, де близьке до нуля значення означає майже ідеальний збіг прогнозних і фактичних значень, а великі значення вказують на велику помилку моделі.

Отже, кожна з вище наведених метрик має свої переваги та недоліки, тому, підсумуємо:

MAE: вимірює похибку в початкових одиницях, стійка до викидів, але слабо чутлива до більших помилок.

MSE: вимірює похибку у квадратах початкових одиниць, чутлива до більших помилок, чутлива до викидів.

RMSE: вимірює похибку в початкових одиницях, чутлива до більших помилок, чутлива до викидів.

MAPE: вимірює похибку у відсотках від фактичних значень і полегшує порівняння похибки між різними наборами даних, чутлива до похибок менших значень, дуже близьких до нуля.

Оскільки кожна з метрик має свої переваги і вказує на похибки, то для аналізу будуть використовуватися усі наведені метрики, окрім MSE. Замість неї буде використовуватися RMSE.

Метрики оцінювання ефективності прогнозних моделей.

Першим розглянемо коефіцієнт детермінації (R^2) [5], він є важливою статистичною метрикою, яка використовується для оцінювання точності та ефективності прогнозних моделей, а у даному випадку для завдання регресії. Коефіцієнт детермінації (R^2) вимірює частку поясненої варіації залежної змінної (відгуку) моделі регресії щодо загальної варіації залежної змінної [5]. Значення R^2 знаходяться в діапазоні від 0 до 1, і вищі значення вказують на більшу точність та ефективність моделі. Коефіцієнт детермінації обраховується за такою формулою:

$$R^2 = 1 - \frac{\sigma^2}{\sigma_y^2},$$

де σ^2 – умовна за фактором дисперсія залежної змінної (дисперсія помилки моделі), σ_y^2 – дисперсія випадкової величини. Значення R^2 можна інтерпретувати так:

- при $R^2 = 0$: модель не пояснює жодної варіації залежної змінної. Це вказує на те, що модель не придатна для цього завдання;
- при $0 < R^2 < 1$: модель пояснює деяку частину варіації залежної змінної. Чим ближче значення R^2 до 1, тим краще модель прогнозує дані;
- при $R^2 = 1$: модель пояснює всю варіацію залежної змінної, що може вказувати на ідеальну відповідність моделі.

Коефіцієнт детермінації може збільшуватись, якщо додавати до моделі нові змінні, навіть які не є статистично значимі, а також можуть взагалі не мати причино-наслідкового зв'язку, в таких випадках використовується модифікований коефіцієнт детермінації R^2, який знижує якість моделі за її складність, але у випадку який розглянутий в даній роботі не має сенсу використовувати модифікації, так як в будь-якому випадку у моделі буде лише одна змінна.

Інформаційний критерій Акаїки (AIC) [6] є широко використовуваним статистичним інструментом для порівняння та відбору статистичних моделей. AIC є критерієм, заснованим на імовірнісній оцінці якості моделі з урахуванням кількості параметрів моделі та загальної кількості спостережень. AIC допомагає порівнювати та обирати моделі. В загальному випадку AIC:

$$AIC = 2k - 2 * \ln(L),$$

де k – число параметрів у статистичної моделі, L – максимізоване значення функції правдоподібності моделі. Значення AIC інтерпретуються так: чим нижче значення, тим краще модель. AIC не має абсолютного значення і використовується виключно для порівняння різних моделей між собою, тобто цей критерій не говорить нічого про якість моделі, а лише про те яка з даних моделей краще пристосовується до даних серед інших. Також AIC передбачає, що ці моделі правильно специфіковані і мають однакову кількість спостережень. Якщо ця умова не виконується, то використовують інші критерії, наприклад BIC, про яку далі піде мова.

Байєсівський інформаційний критерій (BIC) [7] дозволяє враховувати якість пристосування моделі до даних та її складність, але робить це з використанням байєсівського підходу. BIC є критерієм, заснованим на байєсівському аналізі для побудови оцінки якості моделі з урахуванням кількості параметрів моделі та загальної кількості спостережень. На відміну

від AIC, BIC включає штраф за складність моделі, який залежить від кількості спостережень, що робить його більш строгим критерієм при порівнянні моделей. BIC розраховується за формулою:

$$BIC = k * \ln(n) - 2 * \ln(L),$$

де n – кількість спостережень в даних, k – кількість параметрів моделі, L – максимізоване значення функції правдоподібності моделі. Інтерпретація значень критерія точно така сама як у AIC і була зазначена вище.

Крос-валідація (Cross-Validation, далі CV) – це метод оцінювання ефективності моделей машинного навчання, заснований на розподілі доступних даних на кілька підмножин, подальшому навчанні на частини цих підмножин та валідації. Цей метод дозволяє отримати надійну та об'єктивну оцінку продуктивності моделі та уникнути перенавчання. Крос-валідація проводиться шляхом розподілу всієї вибірки на k підмножинах, що не перетинаються. Потім модель навчається на k-1 підмножинах і оцінюється на підмножині, що залишилася. Процес повторюється k разів, і щоразу валідаційна підмножина змінюється. Після проведення всіх ітерацій результати усереднюються, і виходить узагальнена оцінка прогнозів моделі. Існує багато типів та модифікацій цього методу, але в роботі буде використано метод, який був описаний вище.

2.2. Розробка моделі прогнозування на основі фільтра Калмана та авторегресійної моделі

Побудова авторегресійної моделі для прогнозування цін акцій.

Процес підготовки до побудови авторегресійної моделі передбачає насамперед підготувати сам ряд. Першою стане перевірка стаціонарності за допомогою тесту Дікі-Фуллера. Як нульову гіпотезу встановимо стандартне для подібних тестів максимальне значення p-value в 0.05. Тобто, якщо в результаті тесту отримаємо більше значення, то ряд не стаціонарний, якщо менше або дорівнює то стаціонарний. У більшості випадків при проведені експерименту ряд не був стаціонарний і його необхідно було зводити до стаціонарного. Для цього буде використано, як описано у параграфі 1.2, перетворення Бокса-Кокса. При не нульових λ, на значення які можуть набувати члени ряду, накладаються обмеження, а саме вони повинні бути додатніми, оскільки у рівнянні (1.9) використовується процедура логарифмування. Також у параграфі 3.1 буде обґрунтована потреба нормалізувати дані, це буде відбуватися за формулою (3.1). У такому випадку маємо, що значення ряду будуть коливатися на проміжку [0, 1]. Але, як було сказано вище, 0 не входить до області допустимих значень для перетворення Бокса-Кокса з ненульовою λ. Існує декілька підходів до розв'язання цієї проблеми. Перший підхід полягає у тому, що при використанні першої формули у рівнянні (1.9), коли λ близька до нуля і коли вона додатна, можна розглянути границю цієї функції

$$\lim_{y\to 0} \frac{y^{\lambda}-1}{\lambda} = -\frac{1}{\lambda}.$$

У такому випадку нульове значення в даному ряді можна буде перетворити на $-\frac{1}{\lambda}$. Але цей підхід має значний недолік, він вимагає додатні та близькі до нуля значення λ, хоча на практиці інколи зустрічаються від'ємні значення λ.

Тому у проведеному обчислювальному експерименті, використовувався інший підхід, а саме зсув вправо всього вхідного ряду перед проходженням перетворення Бокса-Кокса на значення дуже близьке до нуля. Конкретне значення яке краще обрати, буде залежати від апаратних можливостей певної машини на якій виконується програма, тому це буде описано у ході експерименту. Після того як авторегресійна модель побудує прогноз на основі даного їй ряду, результати будуть перетворені у зворотному напрямку до початкових.

Наступним важливим кроком для побудови авторегресійної моделі є знаходження оптимальної кількості коефіцієнтів авторегресії та їх конкретних значень. Підбір кількості коефіцієнтів буде відбуватися шляхом перебору авторегресійних моделей від першого порядку до двадцятого, і перевірятися на якість інформаційним критерієм BIC, який описано у параграфі 2.1. Перебирати авторегресійні моделі вище двадцятого порядку не має сенсу, тому що на практиці майже ніколи не зустрічається ефективна модель з такими високими порядками. Варто відмітити, що це тому що BIC штрафує модель за її складність. Інформаційний критерій у ході експерименту буде використовуватися з декількох міркувань. По-перше з теоретичних міркувань, а саме це те що BIC враховує кількість спостережуваних значень і порядок авторегресійної моделі, що захищає від перенавчання моделі, оскільки в експерименті для кожного набору даних, буде підбиратися своя, оптимальна модель яка більше всього ефективна саме на цій вибірці. Друга причина, це те що в даному експерименті, на даних які були відібрані для тестування (про це детальніше у параграфі 3.1) модель відібрана за критерієм BIC майже завжди показувала кращі результати ніж AIC.

Коефіцієнти авторегресійної моделі шукають за допомогою умовного методу максимальної правдоподібності. Цей метод полягає у знаходженні таких значень параметрів, які максимізують логарифмічну функцію

правдоподібності, детальніше про яку можна дізнатися з джерела [8]. Для розв'язання цієї задачі оптимізації застосуємо стандартний метод Ньютона-Рафсона. Обґрунтування цього методу можна знайти у джерелі [9].

Визначення початкового стану системи для фільтру Калмана.

Визначення матриці процесу (F) будується відповідно до кожної авторегресійної моделі. Для кожного набору даних обирається найкраща модель, за допомогою метрик, які описані вище. Побудова відбувається, згідно з теоретичними викладками, описаними у параграфі 1.2. Там описано як обиралися матриці спостережень H та коваріаційні матриці похибки моделі Q, а також початковий стан x. Залишилось обрати коваріаційні матриці похибки вимірювань та початкового стану. Враховуючи те, що авторегресійна модель побудована на минулих значеннях, які були виміряні, і початковий стан також є виміряними початковими значеннями, то ці коваріаційні матриці будуть мати однакове значення. Тепер потрібно обрати конкретні значення елементів цих матриць. Оскільки наші вимірювання дуже точні то значення повинно бути дуже близьким до нуля. Але не нуль, інакше фільтр буде ігнорувати побудовану авторегресійну модель. Оскільки точно оцінити оптимальне значення для кожного набору є важкою та неоднозначною задачею, було проведено експеримент, для пошуку оптимального значення для максимізації точності прогнозів за допомогою фільтра Калмана на основі авторегресійної моделі. Для експерименту обрано понад 2000 випадкових наборів даних. Відповідні метрики описані у параграфі 2.1. Нижче на рисунках (2.1) та (2.2) показано результати цього експерименту. Для обрання найкращої моделі потрібно знайти мінімальне значення в кожній із метрик. В даному випадки всі метрики були найменшими у точці 10^{-4}. Вона і буде обрана в якості значення коваріаційної матриці похибки вимірювань та початкового стану.

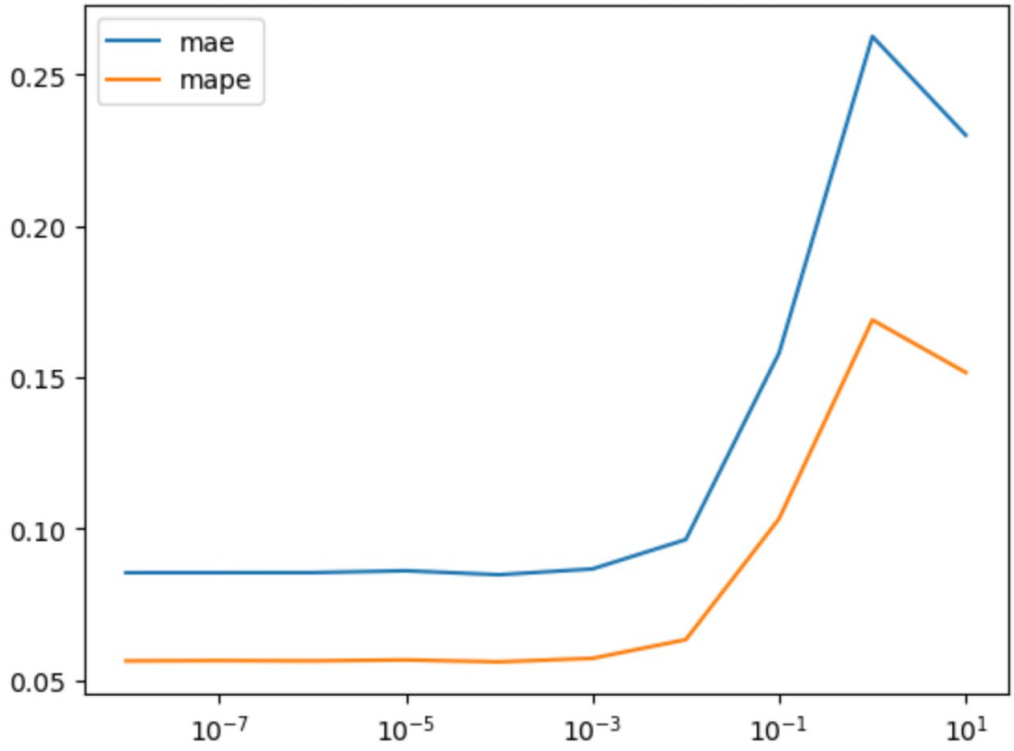


Рис. 2.1 – MAE та MAPE метрики для побудованих моделей

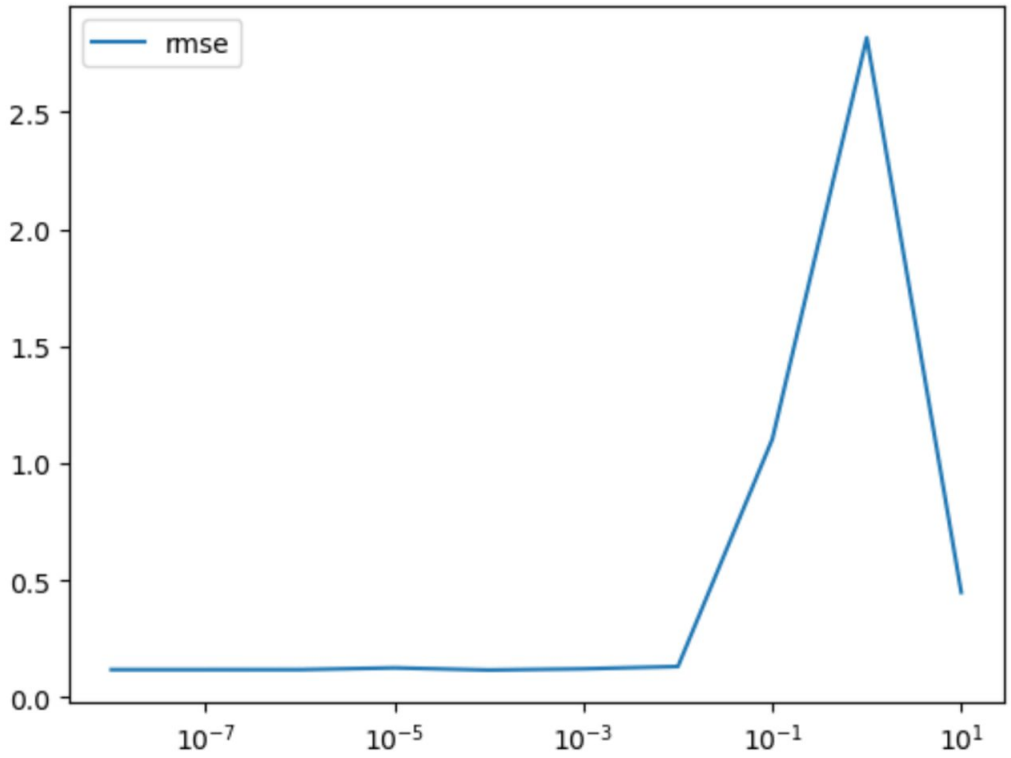


Рис. 2.2 – RMSE метрика для побудованих моделей

2.3. Розробка моделі прогнозування на основі рекурентних нейронних мереж та LSTM

Загальний підхід до створення ефективної моделі на основі LSTM

З технічного аналізу відомо, що існують певні шаблони за якими розвиваються ціни на фондовому ринку, також відомо, що компанії мають свої особливі шаблони, це стосується великих компаній з довгою історією. Для того щоб врахувати це, необхідно щоб нейрона мережа навчалась на історії певної компанії для якої в подальшому нейрона мережа повинна буде робити прогноз. Але більшість компаній має не достатньо довгу історію коливання цін, що не достатньо для навчання нейронної мережі, оскільки збільшення кількості епох призведе до перенавчання і в подальшому поганої адаптації до нових даних. З іншого боку, якщо зробити більш просту модель, яка буде швидше навчатися, то ця модель теж буде погано адаптуватися до нових даних, оскільки «запам’ятає» не достатньо шаблонів поведінки цін. Тому для покращення прогнозування навчання відбувалося у декілька кроків. На першому кроці навчання, була обрана велика кількість випадкових історій різних компаній і навчання з маленькою кількістю епох. На другому кроці навчання, брались ваги з першого кроку та частина історії однією конкретної компанії на якій в подальшому ця нейронна мережа повинна буде робити прогноз. На другому етапі використовувалась менша швидкість навчання та більша кількість епох. Таким чином, точність прогнозів у рамках експерименту була максимальною.

Методи запобігання перенавчанню та недонавчанню.

Перенавчання та недонавчання є поширеними проблемами під час навчання глибоких нейронних мереж. Для боротьби з цими проблемами у вибраній моделі використовується шар виключення (з англ. Dropout). В даному випадку шар виключення застосовувався після кожного шару LSTM з коефіцієнтом відсікання 0.05. Метод виключення дозволяє випадково обрати виходи нейронів під час навчання та робить їх нульовими. Така процедура

дозволяє знизити вплив окремих нейронів та покращити узагальнюючу здатність моделі. Це допомагає запобігти перенавчанню, коли модель занадто точно запам'ятовує навчальні дані та не може добре узагальнювати нові дані.

Застосування шару виключення також сприяє усуненню недонавчання. Недонавчання виникає у випадку коли модель не змогла отримати достатньо інформації з навчальних даних і не може досягти високої продуктивності на тестових даних. Метод виключення запобігає недонавчання, урізноманітнивши навчання та покращивши здатність моделі до узагальнення. Шляхом використання шару виключення після кожного LSTM шару з невеликим коефіцієнтом відсікання 0.05, дана модель забезпечує баланс між вилученням інформації з даних та регуляризацією, що допомагає усунути перенавчання та недонавчання. Регулювання ваг, зазвичай, відбувається шляхом додавання штрафу до функції втрат, який збільшується разом з величиною ваг. Це допомагає запобігти ситуації, коли ваги моделі стають надто великими і модель стає надмірно складною. Разом з тим використовується рання зупинка, яка полягає у припиненні навчання, коли продуктивність на валідаційному наборі даних перестає покращуватись.

Архітектура побудованої моделі

Архітектура моделі включає послідовне з'єднання декількох шарів, де кожен шар виконує певні обчислення. За фінальну модель (найбільш успішну) обрано таку архітектуру:

1. Перший шар LSTM(150): Цей шар LSTM містить 150 нейронів і використовується як перший шар моделі. Він приймає вхідні дані за певною формою та повертає на виході деяку послідовність. Це дозволяє передавати інформацію між тимчасовими кроками моделі.
2. Шар виключення: Після першого шару LSTM застосовується шар виключення з коефіцієнтом відсікання 0.05.

3. Шар LSTM(75): Другий шар LSTM містить 75 нейронів та використовується для подальшої обробки даних. Він приймає на вхід послідовності, сформовані попереднім шаром LSTM, і повертає останній вихідний елемент.
4. Шар виключення: Після другого шару LSTM застосовується ще один шар виключення з коефіцієнтом відсікання 0.05 для подальшого зменшення перенавчання.
5. Повнозв'язний шар: Останній шар моделі є повнозв'язним шаром (з англ. Dense) з одним нейроном і лінійною активацією. Цей шар генерує прогнозоване значення на основі вхідних даних та попередніх шарів моделі.

Обрані параметри для навчання моделі

У процесі розробки моделі необхідно задати певні налаштування та гіперпараметри, які впливають на навчання та результати моделі.

Для цієї моделі вибрано оптимізатор Adam. Параметр швидкості навчання (з англ. learning rate) для першого кроку навчання був стандартний, а саме 0.001. Для другого кроку навчання значення становило менше, а саме 0.0005. Оптимізатор Adam є найбільш популярним вибором для навчання нейронних мереж, оскільки він ефективно оновлює ваги моделі.

Як функція втрат (з англ. loss function) використовується середньоквадратична помилка (MSE). Ця функція широко застосовується у завданнях регресії, оскільки дозволяє оцінити різницю між прогнозованими значеннями та істинними значеннями.

Додаткові гіперпараметри моделі включають:

- Розмір пакета: (з англ. batch size) встановлений на 32. Він визначає кількість зразків, що оброблюються за одну ітерацію навчання. Більші

значення можуть прискорити навчання з допомогою паралельної обробки, але потребують більше пам'яті.

- Епохи: Кількість епох для першого кроку навчання встановлено на 3. Для другого кроку навчання значення встановлено на 5. Епоха являє собою один прохід через весь набір даних навчання. Значення кількості епох вказує на кількість циклів навчання моделі за усім набором даних.
- Значення validation split встановлено на 0.1, що відповідає 10% загального обсягу даних і використовується для створення валідаційного набору даних. Валідаційний набір даних використовується для оцінювання продуктивності моделі під час навчання та допомагає контролювати перенавчання.

РОЗДІЛ 3. ОБЧИСЛЮВАЛЬНИЙ ЕКСПЕРИМЕНТ ТА АНАЛІЗ ЙОГО РЕЗУЛЬТАТІВ

3.1. Підготовка даних

У роботі використовувалися дані цін акцій підприємств, отримані з джерела [14]. Для першого етапу навчання взято компанії з такими тикерами: TSLA, IFRX, PLTR, F, DELL, HPE, BAC, T, SCHW, AMC, SOFI, PLUG, FRC, PFE, KEY, MARA, SWN, GOLD, CCL, WAL, VZ, BURU, SNAP, SIRI, MU, AAL, UBER, VTRS, FRGT, JNJ, CMCSA, HBAN, XOM, RIOT, PACW, LYFT, WFC, WBD, ETRN, LUMN, ELAN, NU, DLO, RIG, CSX, KO, BBAI, USB, IONQ, SHOP, KDP, FCEL, GRAB, SCLX, NEM, C, SQ, TFC, BAX, CHPT, SPCE, MRO, KMI, DKNG, LCID, GM, FTCH, PCG, PARA, MRVL, PYPL, COIN, FCX, ONON, ZIM. Для другого етапу навчання та для проведення експерименту з використанням фільтра Калмана та рекурентної нейромережі використано дані з джерела [14] з такими тикерами: AAPL, ACN, AMD, AMZN, CSCO, GOOGL, IBM, INTC, META, NTDOY, NVDA, ORCL, PCRFY, SONY, SFTBY, TM, MBIO. Для усіх подальших експериментів та аналізів використовувалися лише ціни закриття. Це пов'язано з тим, що ціна закриття акцій є кінцевим значенням дня і відображає консенсус між покупцями і продавцями після всіх торгових сесій в цей день. Крім того, ціна закриття часто використовується як базова точка для аналізу наступного торгового дня.

Для забезпечення якісного навчання моделей та перевірки їх ефективності дані поділено на навчальну вибірку, валідаційну та тестову.

Майже 80% усієї історії цін акцій було використано для навчання рекурентних нейронних мереж, застосовуючи метод крос-валідації. Цей підхід забезпечує більш точну оцінку продуктивності моделі, оскільки розглядає кілька різних розбитих даних та усереднює результати. Решта 20% даних, що залишилися, були використані для тестування нейронної мережі та фільтра Калмана, що дозволяє оцінити їх здатність до прогнозування на даних, які моделі раніше не використовували.

Масштабування даних також є важливим етапом підготовки даних. У даному випадку дані масштабовано так, щоб їх кінцеві значення лежали в діапазоні від 0 до 1. Це відбувалося за такою формулою:

$$z_i = \frac{y_i - \min(y)}{\max(y) - \min(y)} . \qquad (3.1)$$

Цей підхід використовується для покращення продуктивності та стабільності процесу навчання, тобто щоб нейронна мережа була здатна підтримувати якість даних прогнозів незалежно від величини даних. Крім того, такі алгоритми як рекурентні нейронні мережі та особливо довга короткострокова пам'ять часто навчаються швидше та працюють краще з нормалізованими даними. Масштабовані дані також використано і для фільтру Калмана, аби полегшати в подальшому порівняння результатів цих методів.

Для навчання моделей обрано вікно (діапазон індексів) 50 значень, де 49 значень подаються на вхід моделі (нейронної мережі або фільтру Калмана), а одне значення вони повинні передбачити. Вибір розмірності вікна обумовлений кількома чинниками. По-перше, включення достатньої кількості спостережень дозволяє моделі вловити більш довгострокові залежності даних. По-друге, розмір вікна не повинен бути занадто великим, щоб уникнути проблем із перенавчанням та спростити навчання моделі. Також цей розмір вікна дозволяє моделі прогнозувати наступне значення на основі достатньої кількості попередніх спостережень, що відповідає реальним умовам прогнозування на фондовому ринку.

3.2. Хід експерименту з використанням рекурентних методів прогнозу

В ході експерименту з використанням фільтра Калмана та авторегресійної моделі отримано результати приведені у табличці 3.1, де кількість прогнозів це кількість вікон (діапазон індексів), які могла проаналізувати модель, а час прогнозів - це час, який знадобився моделі щоб побудувати прогнози для усіх вікон.

Тикер	Кількість прогнозів	Час прогнозів	MAE	RMSE	MAPE
IBM	3028	121.30422	0.0959009	0.1431639	0.0676371
SONY	2472	100.98156	0.0907343	0.1252366	0.0596405
AMD	2115	85.530893	0.1001445	0.2214034	0.0666793
INTC	2115	84.697520	0.1032790	0.4143212	0.0710523
PCRFY	2115	85.040604	0.0894628	0.1285872	0.0625570
AAPL	2077	83.770623	0.0886300	0.1210997	0.0585407
ORCL	1812	72.814800	0.1028171	0.1486002	0.0692087
CSCO	1613	65.291626	0.0987580	0.1455773	0.0685238
NTDOY	1272	50.680040	0.1021409	0.1479604	0.0704638
AMZN	1247	50.338834	0.0957637	0.1302930	0.0640662
NVDA	1162	46.358647	0.0935178	0.1369635	0.0615269
ACN	1036	41.653254	0.0920981	0.1283399	0.0600756
GOOGL	882	35.425724	0.1067370	0.1463835	0.0694025
META	491	19.674090	0.0970611	0.1354402	0.0704288
MBIO	234	9.9410045	0.0959356	0.1288059	0.0743962

Табл. 3.1 – результати ефективності прогнозування фільтра Калмана та авто регресійного моделі

Результати експерименту для нейронної мережі приведені в таблиці 3.2, де стовпчик «дані для навчання» відображає кількість вікон з історії компанії, які брали участь у навчанні рекурентної нейронної мережі на другому етапі. Варто також відмітити, що під час навчання на першому етапі всього використано 378600 вікон, з компаній, які мають відповідні тикери зазначені у 3.1.

Тикер	Дані для навчання	Кількість прогнозів	Час прогнозів	MAE	RMSE	MAPE
IBM	12310	3028	144.46745	0.0748242	0.1057857	0.0534060
SONY	10087	2472	118.06104	0.0699473	0.0948797	0.0463410
AMD	8658	2115	101.19234	0.0690291	0.0981316	0.0469724
INTC	8658	2115	100.00027	0.0745589	0.1056757	0.0530054
PCRFY	8658	2115	99.670283	0.0694960	0.0970644	0.0493928
AAPL	8508	2077	102.93946	0.0691315	0.0957290	0.0463727
ORCL	7448	1812	86.363679	0.0766865	0.1160100	0.0533399
CSCO	6651	1613	77.337085	0.0749977	0.1100679	0.0522970
NTDOY	5284	1272	60.166689	0.0798413	0.1085988	0.0555356
AMZN	5186	1247	61.098280	0.0740570	0.0994337	0.0507459
NVDA	4846	1162	57.714678	0.0724748	0.0976888	0.0478375
ACN	4344	1036	52.784957	0.0715995	0.0977901	0.0482565
GOOGL	3725	882	42.630993	0.0777304	0.1087904	0.0525753
META	2164	491	23.362783	0.0762982	0.1059343	0.0564711
MBIO	1132	234	11.498573	0.0761939	0.1048176	0.0578161

Табл. 3.2 – результати ефективності прогнозування нейронної мережі з використанням LSTM

У ході експерименту для демонстрації переваги в точності прогнозів фільтра Калмана на авторегресійної моделі над рекурентною нейронною мережею в умовах, коли відсутня велика кількість даних проведено експеримент з навчанням нейронної мережі виключно на історії цін акцій однієї з компанії. Для прикладу взято компанії під тикером META, яка всього має часовий ряд у 2655 значень цін. Архітектура нейронної мережі розглянута, як і у основної нейронної мережі, що описана у 2.3. Точні значення метрики прогнозу фільтра Калмана зазначено у таблиці 3.1. А порівняння метрик зазначено у додатку А. Відповідно, в додатку А.1 побудовано графік залежності значень метрики оцінки MAE для прогнозів моделі від кількості даних, на яких навчалась нейронна мережа. В додатку А.2 для значень метрики RMSE, а в додатку А.3 для MAPE.

3.3. Аналіз отриманих результатів

Для зручності аналізу дані з таблиці 3.1 та 3.2 буде зображено у вигляді графіка для кожної з метрик. Розглянемо MAE метрику. Відповідний графік зображений на рисунку 3.1. На осі ординат записані значення метрик, а на осі абсцис зображена кількість прогнозів, які побудувала кожна з моделей. Як зазначено у 2.1 в метриці MAE чим ближче результат до нуля тим точніший прогноз дає модель. На графіку 3.1 видно, що з цією задачею рекурентна нейронна мережа справилася краще на 0.0230743, що визначено шляхом обчислення різниці середніх значень для двох результатів. Також на графіку можна побачити, що форма графіку дуже схожа, але для нейронної мережі має менший розмах (різниця між найбільшим та найменшим значенням) на 0.0072948, що свідчить про спроможність нейронної мережі давати прогнози з більш стабільною точністю.

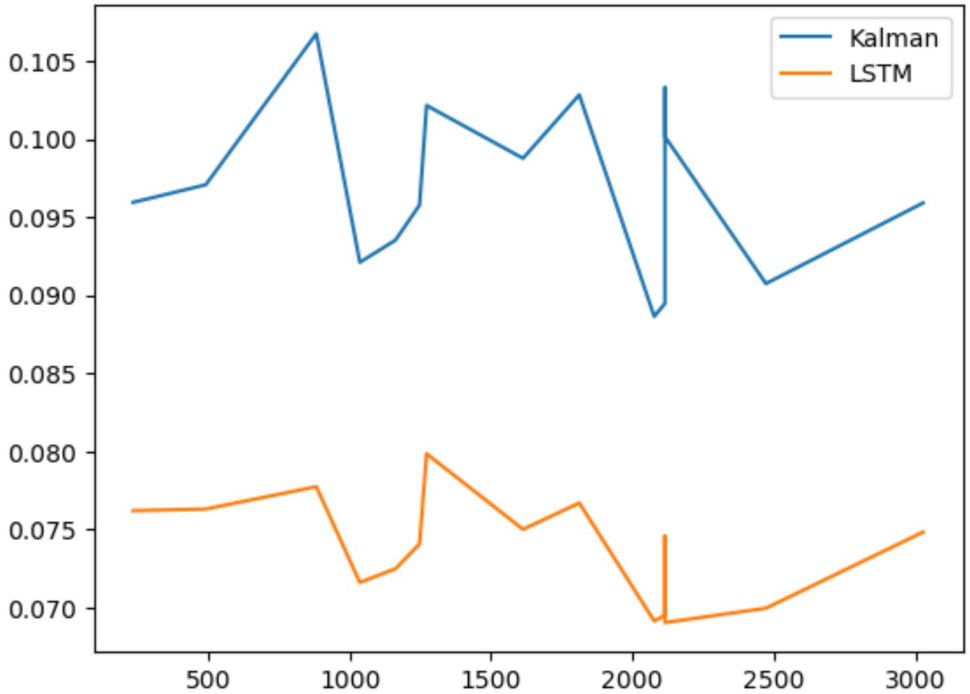


Рис. 3.1 – порівняння MAE для побудованих моделей

Далі, на рисунку 3.2 зображено порівняння метрик RMSE для двох моделей. Тут теж нейронна мережа дозволила побудувати прогноз в

середньому кращий на 0.05705188. Але одразу можна на графіку помітити значення, яке різко відрізняється від більшості інших. Це відбулося при прогнозуванні цін акцій компанії за тикером INTC.

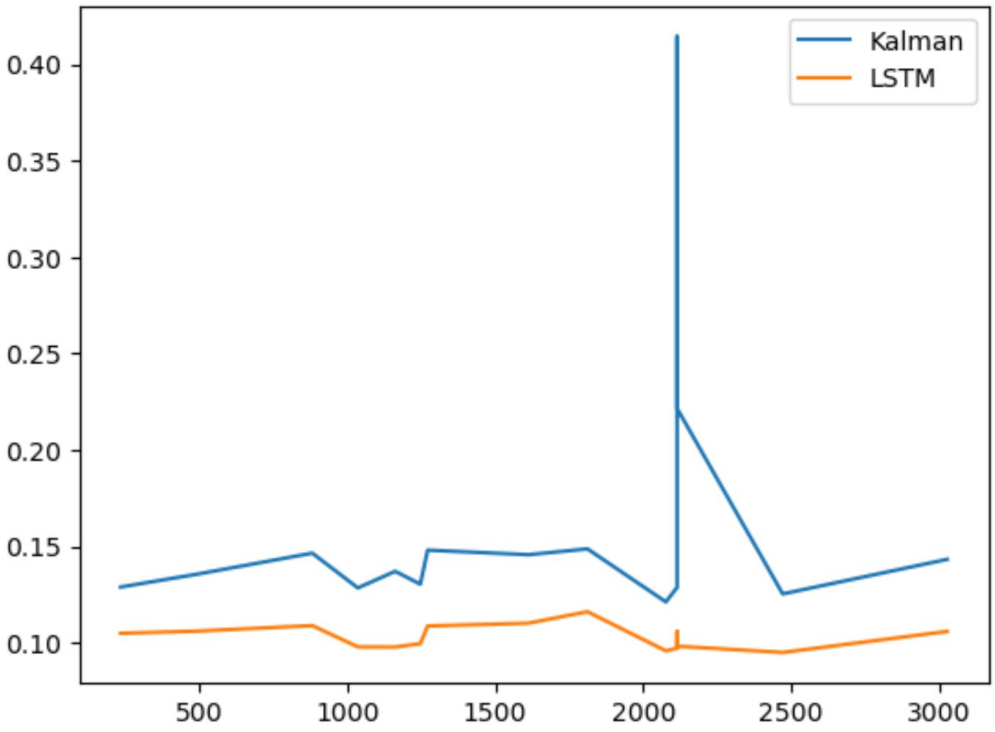


Рис. 3.2 – порівняння RMSE для побудованих моделей

Причиною цього могло статися те, що в історії цін акцій цієї компанії було багато моментів коли значення правильного прогнозу могли різко відрізнятися від їх попереднього стану. Наприклад, якщо порахувати розмах (різниця максимального та мінімального значення) на множені, яку було визначено шляхом різниці останнього (це те значення, яке модель буде прогнозувати) та перед останнього значення, то буде видно, що в історії цін акцій компанії за тикером INTC вище зазначений розмах буде найвищим. Приклади значень для сусідніх тикерів зазначені в таблиці 3.3.

Усі значення, які були обраховані в таблиці 3.3 були попередньо нормовані відповідно до формули 3.1.

Тикер	RMSE	Розмах

SONY	0.1252366	1.436379
PCRFY	0.1285872	1.424155
INTC	0.4143212	1.630607
AAPL	0.1210997	1.179358

Табл. 3.3 – порівняння розмахів для сусідніх тикерів

Оскільки метрика RMSE чутлива до великих похибок то можна зробити висновок, що прогноз рекурентного фільтру Калмана та авторегресійної моделі доволі сильно відхилявся від правильного значення, через те що в самій історії цін акцій було багато різких падінь та навпаки різких підйомів. Також видно, що в цьому випадку рекурентна нейронна мережа досить добре справилася з прогнозом, не маючи великих відхилень. Проте, в цілому алгоритм рекурентного фільтра Калмана справився не набагато гірше.

Наступною розглянутою метрикою для порівняння є MAPE, що зображена на рисунку 3.3. Жодних викидів на графіку не помічено, що свідчить про те, що обидва алгоритми стабільно прогнозували значення.

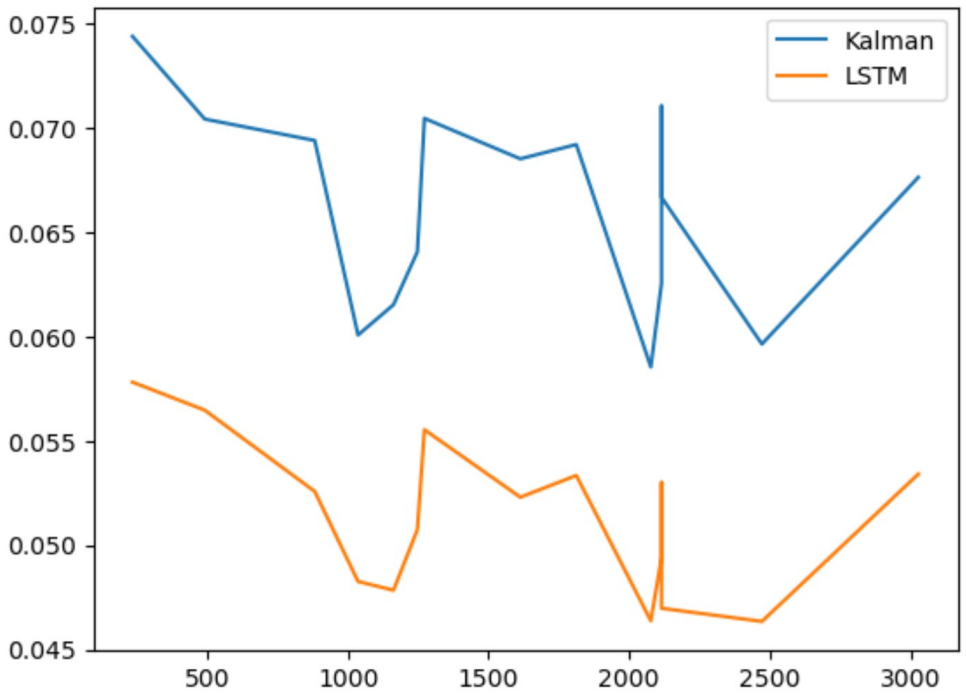


Рис. 3.3 – порівняння MAPE для побудованих моделей

Середня різниця значень для цих двох алгоритмів становить 0.01492228, що є не великою і прийнятною різницею в точності. Різниця розмаху значень

теж є прийнятною, а саме 0.0043804, що свідчить про схожу стабільність прогнозів фільтра Калмана та нейронної мережі. Проте остання дозволяє будувати більш точні та стабільні прогнози.

Останнє порівняння стосується часу, який потрібен для того, щоб модель побудувала прогноз. Графік залежності часу від кількості прогнозів зображено на рисунку 3.4. Він є лінійним та досить схожим для обидвох алгоритмів, проте фільтр Калмана швидше в середньому на 12.385674 секунд ніж рекурентна нейронна мережа.

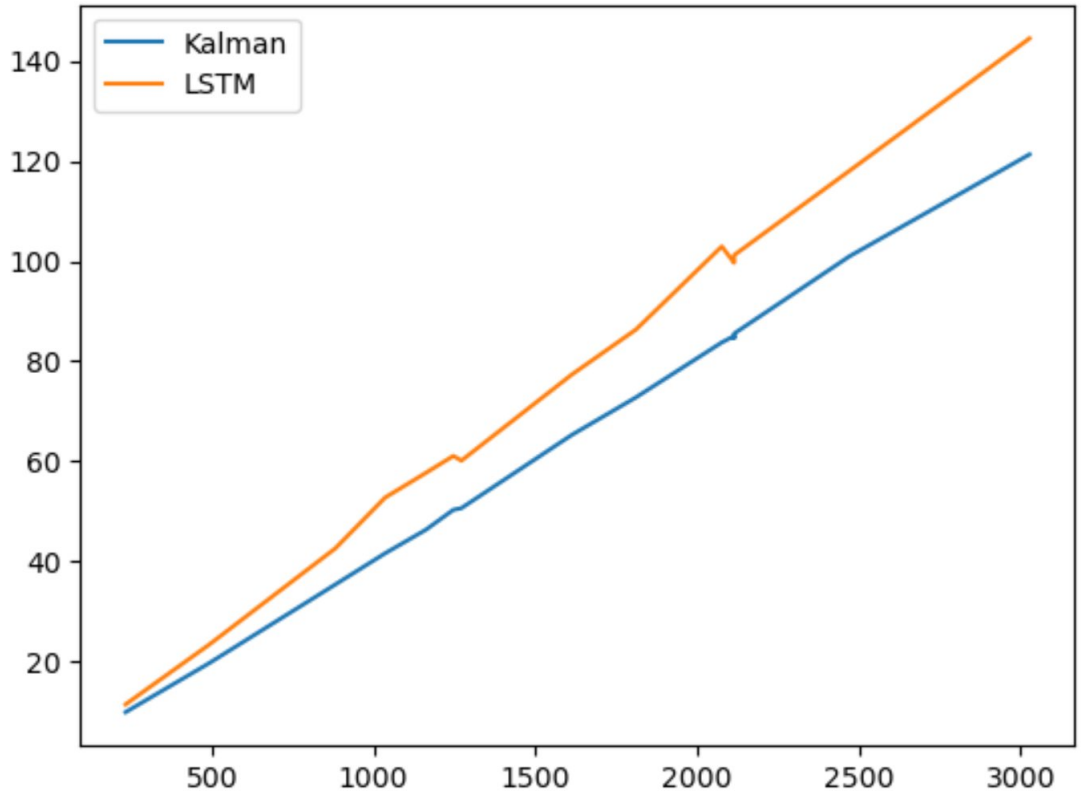


Рис. 3.4 – порівняння часу прогнозування для побудованих моделей

Варто відмітити, що, якщо на початку графіку при значенні кількості прогнозів менше 500, середня різниця швидкості становить 4.15051017, а в кінці, при значені більше 2400 середня різниця швидкості становить вже 20.121355 секунд. Це стається через те, що фільтр Калмана дає прогноз за передбачуваний час, який коливається 0.03-0.05 секунд, приклад приведений для компанії під тикером IBM у додатку Б.1. А графік, що демонструє час роботи рекурентної нейронної мережі наведено у додатку Б.2, де можна

спостерігати викиди під час прогнозу. І чим більша кількість прогнозів тим більше таких викидів трапляється. Наприклад, у компанії під тикером META, для якої потрібно більше ніж в 6 разів менша кількість прогнозів, цих викидів майже немає і час прогнозу коливається від 0.04 до 0.05 секунд (графік наведено у додатку Б.3.)

ВИСНОВКИ

У даній кваліфікаційній роботі проведено порівняльний аналіз рекурентних алгоритмів для прогнозування цін на фондовому ринку. Основна мета роботи полягала у вивченні та порівнянні ефективності двох рекурентних моделей прогнозування: моделі, заснованої на фільтрі Калмана та авторегресійної моделі, та моделі, заснованої на рекурентних нейронних мережах, а саме довгій короткостроковій пам'яті.

На основі проведеного дослідження можна зробити такі висновки:

- Розроблена рекурентна нейронна мережа на основі довгої короткострокової пам’яті в умовах доступності до великої кількості історичних даних дозволяє отримати більш точні прогнози.
- Для того, щоб рекурентна нейронна мережа на основі довгої короткострокової пам’яті даозволяла отримати більш точні прогнози, потрібні велика вибірка даних історії цін акцій в певному секторі фондового ринку та історія акцій цін окремої компанії.
- В цілому рекурентна нейронна мережа на основі довгої короткострокової пам’яті працює більш повільно ніж фільтр Калмана на основі авторегресійної моделі, що може бути суттєво в певних умовах, наприклад в торгівлі на дуже маленьких відрізках часу.
- Рекурентний алгоритм фільтра Калмана на основі авторегресійної моделі хоч і поступається в точності прогнозів, але точність залишається все ще на тому рівні, що дозволяє використовувати цей алгоритм для реальних прогнозів, в умовах коли недостатньо даних за попередні сесії торгівлі, оскільки даний алгоритм не потребує будь яких довгих попередніх історичних даних.
- Також виявлено, що в умовах різких, сильних змін цін, рекурентна нейронна мережа спроможна давати результати з стабільною точністю,

на відміну від фільтра Калмана. Тому у таких випадках, можливо, не доцільно буде використовувати наведений алгоритм фільтра Калмана.

- Алгоритм на основі фільтра Калмана та авторегресійної моделі показав більш точні прогнози у випадку, коли рекурентна нейронна мережа не має достатньо великого набору історичних даних для навчання.

Отже, наведений алгоритм побудови короткострокового прогнозу ринкових цін акцій з використанням фільтра Калмана має достатню точність для реального застосування фондовому ринку та перевагу у швидкості виконання. Варто зауважити, що цей алгоритм має переваги і у точності в умовах недостатньої кількості даних. Проте, у випаду коли дані для навчання доступні у дуже великих кількостях, наведений алгоритм з використанням рекурентних нейронних мереж з LSTM дає більш точні результати.

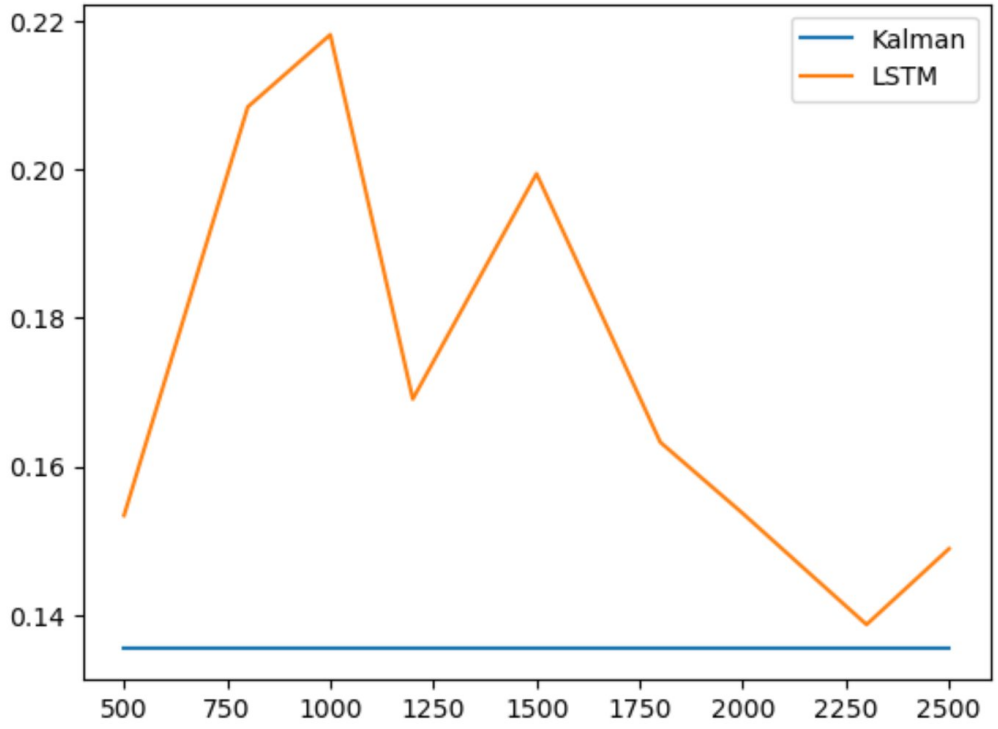
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Roger R., Labbe Jr., 2020. Kalman and Bayesian Filters in Python.
2. Stationary process - Wikipedia
   https://en.wikipedia.org/wiki/Stationary_process
3. Dickey–Fuller test - Wikipedia
   https://en.wikipedia.org/wiki/Stationary_process
4. Peter Bruce and Andrew Bruce 2018. Practical Statistics for Data Scientists.
5. Coefficient of determination - Wikipedia
   https://en.wikipedia.org/wiki/Coefficient_of_determination
6. Akaike information criterion - Wikipedia
   https://en.wikipedia.org/wiki/Coefficient_of_determination
7. Bayesian information criterion - Wikipedia
   https://en.wikipedia.org/wiki/Coefficient_of_determination
8. Maximum likelihood estimation - Wikipedia
   https://en.wikipedia.org/wiki/Coefficient_of_determination
9. Newton's method - Wikipedia
   https://en.wikipedia.org/wiki/Coefficient_of_determination
10. Box-Cox transformation - Encyclopedia of Mathematics
    https://encyclopediaofmath.org/wiki/Box-Cox_transformation
11. Understanding LSTM Networks -- colah's blog
    http://colah.github.io/posts/2015-08-Understanding-LSTMs/
12. Backpropagation Through Time for Recurrent Neural Network
    https://mmuratarat.github.io/2019-02-07/bptt-of-rnn
13. Gated recurrent unit - Wikipedia
    https://en.wikipedia.org/wiki/Gated_recurrent_unit
14. Yahoo Finance - Stock Market Live, Quotes, Business & Finance News
    https://finance.yahoo.com/

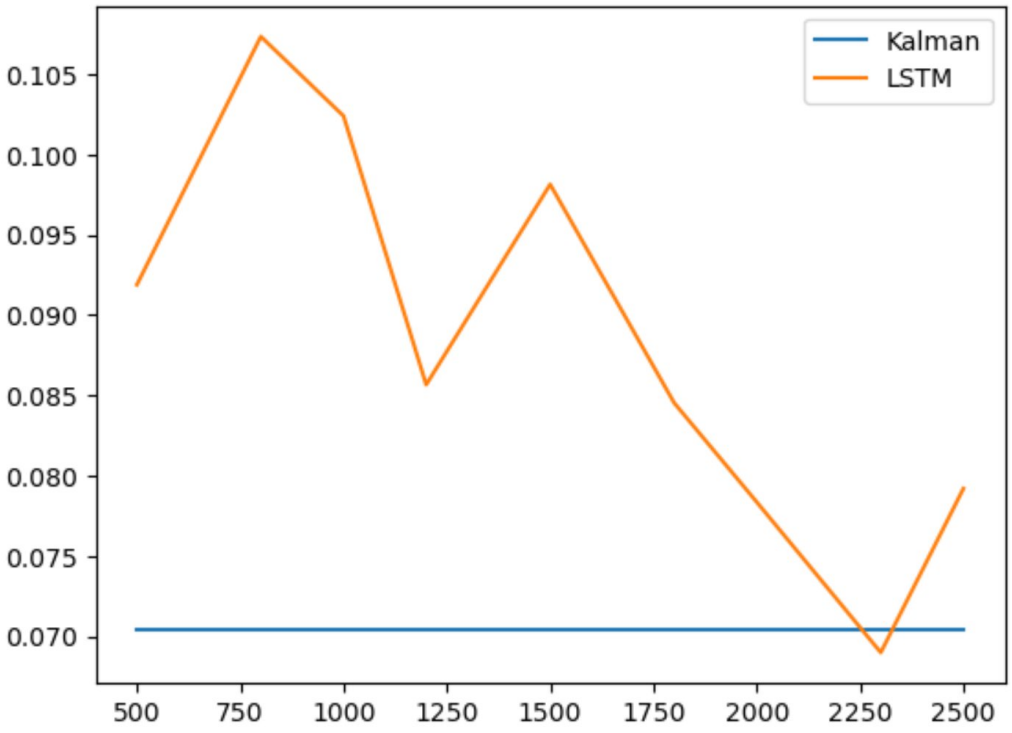
15. Root Mean Squared Error Versus Mean Absolute Error (jmlb.github.io)
https://jmlb.github.io/flashcards/2018/07/01/mae_vs_rmse/

Додаток А. Порівняння точності в умовах малої кількості даних.

Додаток А.1

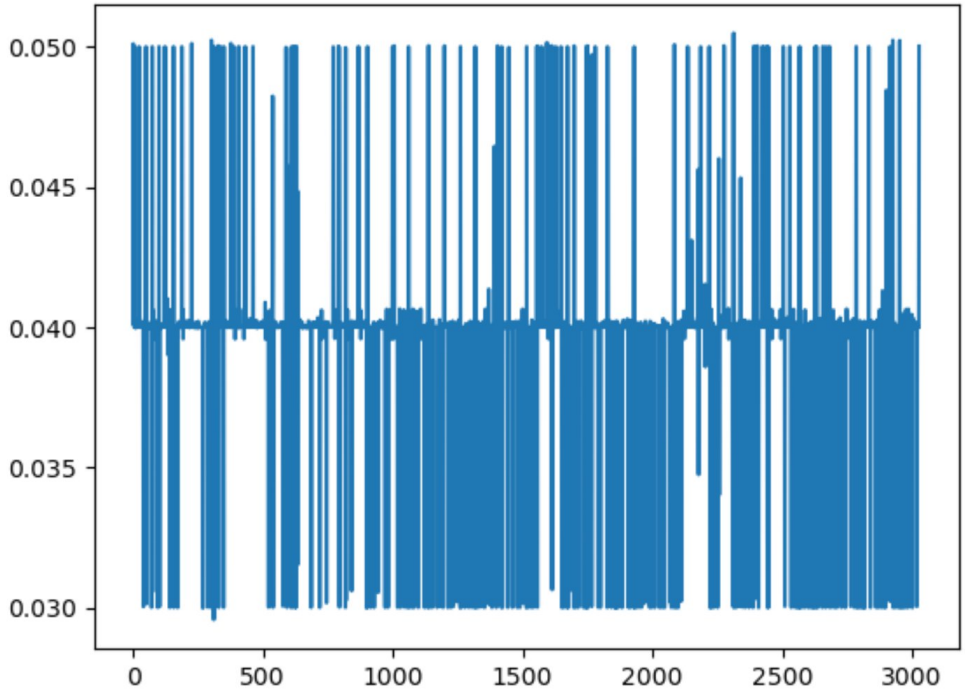


Додаток А.2

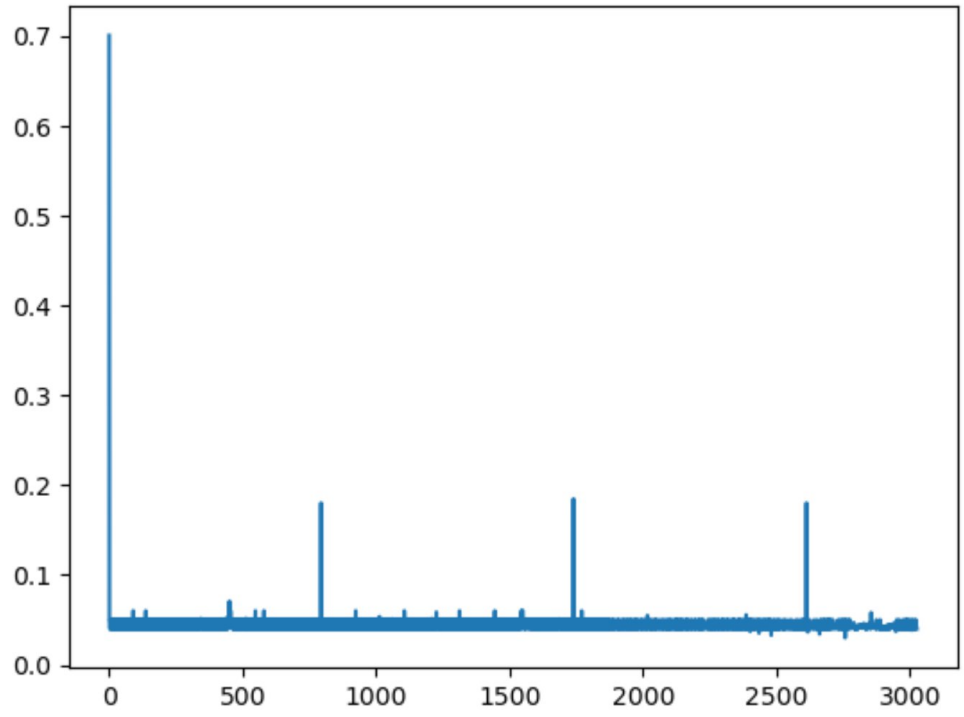

Kalman
LSTM
0.105
0.100
0.095
0.090
0.085
0.080
0.075
0.070
500
750
1000
1250
1500
1750
2000
2250
2500

Додаток А.3

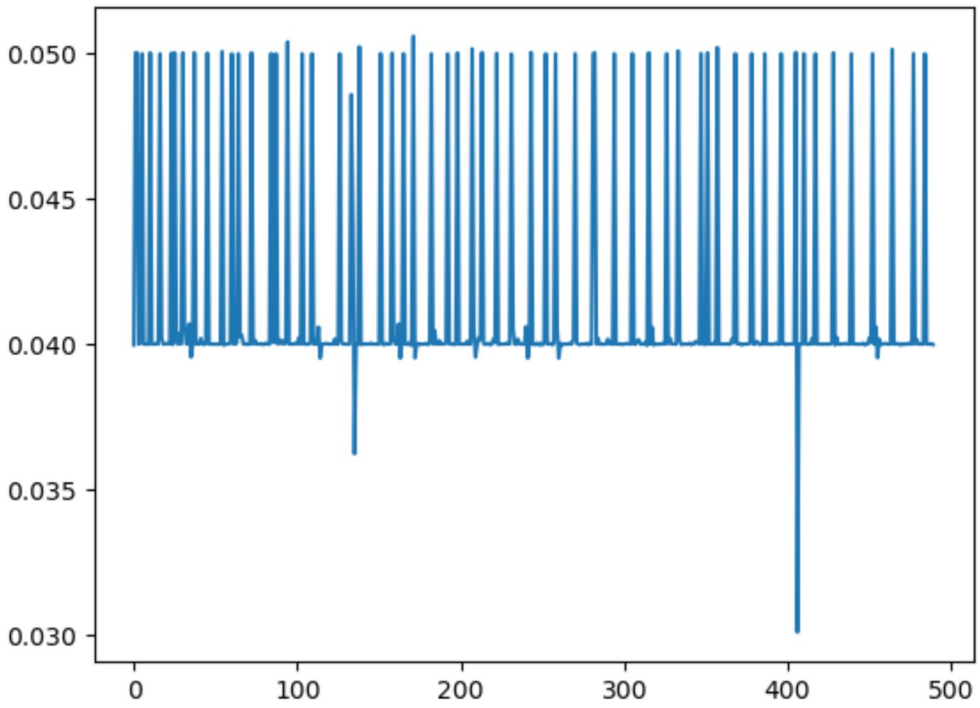
Додаток Б. Порівняння часу, потрібного для побудови прогнозу.



Додаток Б.1



Додаток Б.2



Додаток Б.3