NATIONAL UNIVERSITY OF KYIV-MOHYLA ACADEMY
Faculty of Informatics
Department of Mathematics

BACHELOR'S THESIS

topic: **Hybrid LSTM-GARCH Architecture for Option Pricing**

Performed by:
Student of the 4th year
Specialty "Applied Mathematics"
Olena Pasternak

Supervisor:
Candidate of Physical and Mathematical Sciences, Associate Professor
N. Y. Shchestyuk

Kyiv – 2026

NATIONAL UNIVERSITY OF KYIV-MOHYLA ACADEMY

Faculty of Informatics
Department of Mathematics
Degree – Bachelor
Specialty 113 “Applied Mathematics”
Educational Program “Applied Mathematics”

APPROVED BY
Head of the Department ____________________
____
“___” __________ 2026

ASSIGNMENT
FOR THE QUALIFICATION WORK TO THE STUDENT
Olena Pasternak

1. Thesis Topic: “Hybrid LSTM-GARCH Architecture for Option Pricing”
Supervisor: *N. Y. Shchestyuk, Candidate of Phys-Math Sciences, Associate Professor*
approved by the university order from “___” ________ 2026 No. ______
2. Deadline for submission: May 17, 2026.
3. Work Plan:

1. Review of pricing methods (BSM vs. Machine Learning), mathematical description of financial derivatives and the hybrid LSTM-GARCH architecture.
2. Collection of AAPL historical quotes and Nasdaq data, development of the preprocessing pipeline, and neural network training in Python.
3. Comparative analysis of forecasting accuracy (RMSE, MAE), generation of graphic materials, and final formatting of the thesis.

Date of issue “___” ________ 2025 Supervisor ______________

Assignment received by ______________

Schedule of Preparation of the Qualification Work for Defense

Schedule approved on "______" ________________ 2025.

No.	Task Description	Deadline	Date of Supervisor's Review	Supervisor's Signature	Note
1.	Literature review and problem statement formulation.	15.10.2025			
2.	Mathematical description of the Black-Scholes and LSTM models.	4.11.2025			
3.	Data collection, cleaning, and normalization.	10.11.2025-10.12.2025			
4.	Submission of the calendar plan and individual assignment to the DistEdu system.	14.12.2025			
5.	Submission of the interim progress report to the DistEdu system.	31.01.2026			
6.	Software implementation of the model, training, and uploading the draft version of the work to DistEdu.	15.03.2026			
7.	Receiving the supervisor's feedback in the DistEdu system.	23.03.2026			
8.	Preliminary defense of the qualification work.	02-03.04.2026			
9.	Comparative analysis of results and formatting the explanatory note.	15.04.2026			
10.	Pre-defense and final corrections.	7-8.05.2026			
11.	Uploading the final version of the work to the DistEdu system (without presentation).	17.05.2026			
12.	Uploading the supervisor's feedback and review to the DistEdu system.	25.05.2026			
13.	Defense of the qualification work.	28-29.05.2026			

Supervisor ______________________

(N. Y. Shchestyuk)

Student ______________________

(O. O. Pasternak)

ABSTRACT

The bachelor's thesis compares the accuracy of a proposed dual-branch Long Short-Term Memory (LSTM) neural network, processing 14-day historical sequences and GARCH-derived volatility, with the classical Black-Scholes-Merton model for pricing European call options. Empirical testing on approximately 800,000 Apple Inc. (AAPL) option records demonstrates that the LSTM network efficiently captures complex non-linear dependencies and market dynamics. At the first forecast step, the model achieved a 63.8% reduction in Mean Absolute Error (MAE) compared to the BSM-GARCH variant, while maintaining high predictive stability over a five-day horizon.

Keywords: Option Pricing, Black-Scholes, Neural Networks, LSTM, Deep Learning, GARCH, AAPL.

Кваліфікаційна робота бакалавра присвячена порівнянню точності запропонованої двогілкової архітектури нейронної мережі довгої короткострокової пам'яті (LSTM), яка обробляє 14-денні історичні послідовності та волатильність за моделлю GARCH, із класичною моделлю Блека-Шоулза-Мертона для оцінювання європейських опціонів Call. Емпіричне тестування на базі близько 800 000 записів щодо опціонів Apple Inc. (AAPL) демонструє, що мережа LSTM ефективно вловлює складні нелінійні залежності та динаміку ринку. На першому кроці прогнозування запропонована модель досягла зниження середньої абсолютної похибки (MAE) на 63,8% порівняно з базовим варіантом BSM-GARCH, зберігаючи при цьому високу прогностичну стабільність на п'ятиденному горизонті.

Ключові слова: оцінювання опціонів, Блек-Шоулз, нейронні мережі, LSTM, глибоке навчання, GARCH, AAPL.

CONTENTS

INTRODUCTION

Financial derivatives, especially options, are widely used for risk management and trading strategies in modern financial markets [2]. The standard method for pricing European options is the Black-Scholes-Merton (BSM) model [3]. Although this model is mathematically elegant, it relies on strict assumptions like constant volatility, which often fail when applied to real market data [4].

A clear example of these market movements can be seen in recent Nasdaq data for Apple Inc. (AAPL). Between March 12 and March 13, 2026, the stock price shifted from $250.12 to $251.447. This change, along with shifts in market sentiment, caused option premiums to change across different strike prices. Standard parametric models often fail to capture such quick, non-linear changes. However, modern machine learning tools, such as Long Short-Term Memory (LSTM) networks, are well-suited for these tasks because they can process historical sequences and volatility patterns effectively [5, 4].

Based on these market complexities, the *object* of this research is the process of pricing European call options in the stock market. The *subject* includes the analytical formulas and machine learning models used for option valuation, focusing on the Black-Scholes formula and LSTM neural networks. The *main goal* of this thesis is to build an optimized LSTM model for option price forecasting and compare its performance against traditional benchmarks.

To achieve this goal, the following research tasks were formulated:

1. Analyze the Black-Scholes-Merton model, its theoretical drawbacks, and advanced parametric frameworks.

2. Select and justify the optimal neural network configuration for option pricing and prediction evaluation.

3. Choose and collect the comprehensive empirical market data required for the investigation.

4. Develop an option pricing calculator implemented as a user-friendly analytical web interface.

In terms of the methodological approach, this work relies on methods from probability theory, stochastic processes, contemporary machine learning paradigms—specific backpropagation through time—and statistical analysis.

The scientific novelty of the study lies in using a dual-branch LSTM structure to model option prices under highly variable market volatility. This configuration allows for the explicit incorporation of long-term temporal dependencies and conditional volatility features that are inherently oversimplified in classical parametric models. From a practical standpoint, the proposed framework offers a reliable tool for asset valuation and market risk assessment.

The theoretical and empirical findings of this research were formally validated at two scientific conferences. The main results were presented and discussed at the XIV All-Ukrainian Scientific Conference of Young Mathematicians (Kyiv, 2026) [1] and the 12th International Conference on Mathematical and Statistical Methods for Actuarial Sciences and Finance (MAF 2026). Structurally, the thesis consists of an introduction, three main sections, conclusions, and a list of references.

1. THEORETICAL FOUNDATIONS OF OPTION PRICING AND NEURAL NETWORKS

1.1 Concept of Options and Mathematical Context

Financial derivatives are contracts that get their value from an underlying asset, such as stocks, stock indices, currencies, or commodities. Among these contracts, options are widely used because they are flexible tools for managing financial risk, hedging, and trading.

In financial markets, we divide options into two basic types depending on the trader's goal:

- **Call Option:** A contract that gives the buyer the right to buy the underlying asset at a set price within a specific timeframe.
- **Put Option:** A contract that gives the buyer the right to sell the underlying asset at a set price within a specific timeframe.

Definition 1.1. A **European call option** is a financial contract that gives the holder the right, but not the obligation, to buy an underlying asset S at a specified strike price K strictly at the expiration time T.

Mathematically, the final value (payoff) of a European call option at expiration T depends on the difference between the final asset price and the strike price. I can write this condition simply as:

$$\max(S_T - K, 0), \tag{1.1}$$

where S_T is the price of the underlying asset at expiration.

Another core concept in option theory is put-call parity. This relationship links the prices of European call and put options on the same asset, assuming they have the same strike price and expiration date. At any time step t, I write this link as:

$$C_t - P_t = S_t - Ke^{-r(T-t)}, \tag{1.2}$$

where C_t is the current price of the call option, P_t is the price of the put option, S_t is the current price of the asset, K is the strike price, r is the constant risk-free interest rate, and $T - t$ is the time left until expiration in years. Any noticeable gap in this equation creates immediate riskless profit opportunities (arbitrage) in the market.

1.2 The Black-Scholes-Merton Model

Fischer Black and Myron Scholes introduced the classic approach to option pricing in their famous 1973 paper [3]. Their model offers an exact formula for option pricing by tracking how the asset price moves over time. The model assumes a market with at least one risky asset (like a stock) and one riskless asset (like cash or a bond). The continuous movements of the stock price follow a geometric Brownian motion (GBM), described by this differential equation:

$$dS_t = \mu S_t dt + \sigma S_t dW_t, \tag{1.3}$$

where S_t is the asset price at time t, μ is the constant drift parameter reflecting the expected return, σ is the constant volatility of the asset returns, and W_t is a standard Wiener process (Brownian motion).

1.2.1 Mathematical Formulation and Parameter Notation

The Black-Scholes-Merton framework provides a classic formula to calculate the price of a European call option:

$$C = SN(d_1) - Ke^{-rT}N(d_2) \tag{1.4}$$

where the variables d_1 and d_2 act as adjustments for risk and probability, defined as:

$$d_1 = \frac{\ln \frac{S}{K} + (r + \frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}}, \quad d_2 = d_1 - \sigma\sqrt{T} \tag{1.5}$$

To understand this equation, I define its parameters in simple financial terms:

- C is the fair price of the European call option at the starting time $t = 0$.
- S is the current price of the underlying asset.
- K is the strike price where the option holder can choose to buy the asset at expiration.
- r is the annualized risk-free interest rate used to discount future cash values.
- T is the time left until expiration, measured as a fraction of a year.

- σ is the asset volatility, which measures how much and how fast the asset price fluctuates.
- $N(\cdot)$ is the cumulative distribution function of a standard normal distribution. Financially, $N(d_2)$ shows the probability that the option will expire in-the-money ($S_T > K$) so the holder will execute it. Meanwhile, $SN(d_1)$ represents the expected value of the asset position using a delta-hedging strategy.

1.2.2 Asset-Specific Hypotheses

The Black-Scholes formula relies on several strict assumptions about how assets behave. First, it assumes that the risk-free interest rate is completely stable and stays the same over time. Second, the stock price must follow a smooth geometric Brownian motion with constant drift and constant volatility. If volatility changes over time, the formula needs modifications, but the volatility still cannot be random. Finally, the model assumes that the stock pays no dividends during the life of the option.

1.2.3 Market Mechanism Hypotheses

The model also assumes an ideal trading environment. It requires that the market is perfectly efficient with no arbitrage opportunities, meaning traders cannot make free riskless profits. Traders must also have the ability to borrow and lend any amount of cash at the risk-free rate, and buy or sell any amount of stock (including short selling), even in fractions. Furthermore, trading must be completely frictionless, meaning there are no transaction fees, taxes, or broker costs. Finally, trading must happen continuously without any pauses. Under these ideal conditions, traders can maintain a perfectly hedged portfolio that removes all risk, which leads directly to the Black-Scholes formula.

1.2.4 The Volatility Smile and Distributional Mismatch

When we compare these assumptions with real market data, we see clear mismatches. This makes the classic formula less reliable in the real world. The biggest problem is assuming that volatility is constant. In reality, market volatility changes all the time and tends to cluster together (high-volatility days follow other high-volatility days). Because the formula forces a single constant volatility σ on all contracts, it fails to capture real return behaviors. When we try to match the model with real market prices, it creates a pattern called the "volatility smile" or "volatility skew" [6]. This pattern proves that constant volatility does not hold true across different strikes and maturities.

1.2.5 Replication Failure and Market Frictions

The assumption of continuous trading with zero costs is another major limitation. Real trading happens at specific times, and adjusting a portfolio always costs money due to fees, taxes, and spreads. Trying to adjust a portfolio continuously as the model suggests would create infinite transaction costs, which would quickly wipe out any theoretical profits. Therefore, traders adjust their positions at discrete intervals, which introduces tracking errors and financial risks that the formula cannot measure.

1.2.6 Liquidity Constraints and Discrete Asset Jumps

Finally, real stock prices do not move as smoothly as geometric Brownian motion suggests. Real assets experience sudden price jumps, trading gaps, and periods of low liquidity, especially during market crises or major news releases. Since the classic formula assumes perfectly smooth price lines, it cannot handle sudden price gaps. This limitation causes the model to underestimate the value of deep out-of-the-money options, where real-world tail risks are much higher than a smooth distribution predicts. These persistent issues highlight why we need advanced econometric tools or flexible, data-driven methods like Long Short-Term Memory (LSTM) neural networks.

1.3 Advanced Pricing Approaches and Neural Network Evolution

Since recent market events have challenged the core assumptions of the Black-Scholes model, quantitative finance has moved toward more advanced methods. Modern financial data clearly shows heavy tails, changing volatility, and low liquidity. As a result, researchers have developed newer analytical and data-driven approaches.

1.3.1 Parametric Modifications and Subdiffusion Frameworks

To fix the limits of geometric Brownian motion, researchers introduced several modifications. One path uses Fractional Brownian Motion, complex models like the Heston architecture, or autoregressive setups like GARCH. Researchers also used "fractal activity clock" setups to include memory patterns in returns [7]. Another approach uses subdiffusion models, which work well for pricing options in illiquid or changing markets. In these setups, option values are calculated using intense Monte Carlo simulations [8, 9] or by solving fractional differential equations, like the fractal Dupire equation [10, 11].

While these advanced models match real market features better, they bring heavy computational problems. They rarely have simple closed-form solutions, making them slow to run, requiring heavy manual tuning, and making them too slow for real-time risk management.

1.3.2 Non-Parametric Data-Driven Alternatives

Instead of using rigid mathematical rules, my research focuses on artificial neural networks as a flexible, data-driven alternative. Machine learning models can learn complex, non-linear relationships directly from historical market data without needing predefined assumptions about asset behavior.

Hutchinson, Lo, and Poggio [12] pioneered this direction by showing that simple networks could easily reconstruct option pricing surfaces and perform successful delta-hedging. Over the decades, these systems grew much more advanced. As Ruf and Wang [4] noted in their literature review, neural networks have evolved from basic feedforward structures into complex recurrent and deep networks capable of joint pricing and calibration.

Recent studies, like those by Amornwattana et al. [13] and Zhang et al. [14], show that hybrid network architectures which combine econometric models like GARCH with deep multi-input systems capture market changes with excellent accuracy, heavily outperforming traditional models in complex volatility environments.

2. DATA ACQUISITION AND PREPROCESSING PIPELINE

The predictive power and generalization capabilities of deep learning models in derivative pricing depend heavily on the quality, continuity, and structure of the training data. Unlike classic parametric models that only require market inputs from a single moment, recurrent neural networks require continuous historical sequences to learn hidden market dynamics. This section describes the multi-stage pipeline developed for data harvesting, professional API integration, volatility modeling, and feature preprocessing.

2.1 Data Sources and Descriptive Statistics

The baseline historical options data for Apple Inc. (AAPL) were primarily sourced from the OptionsDX platform. This dataset covers a dense observation period from September 1, 2020, through December 29, 2023, while the expiration dates for the included option contracts span from September 4, 2020, to January 16, 2026. This long timeframe provides a rich variety of market conditions, including low-volatility growth periods and high-volatility market drops.

To check the quantitative traits of the data before filtering, a basic exploratory data analysis (EDA) was performed. The statistical properties of the key financial fields in the raw dataset are documented in Table 2.1.

Table 2.1: Summary statistics of key fields for AAPL European Call options

Field	Minimum	Maximum	Mean	Volatility
Underlying Asset Price (S_t, \$)	106.50	198.14	151.23	22.39
Maturity Horizon ($T - t$, days)	0.00	1059.00	207.32	236.05
Call Option Premium (C_t, \$)	0.00	155.40	27.42	33.41
Strike Price (K, \$)	18.75	320.00	145.93	64.32

Furthermore, the internal structural composition of the aggregated option dataset was examined across different moneyness states. As illustrated in Figure 2.1, the dataset provides a comprehensive representation of option regimes.

Specifically, In-the-Money (ITM) options form the largest segment, accounting for 53.5% (414,205 observations) of the matrix. Out-of-the-Money (OTM) options represent 44.3% (342,677 observations), while At-the-Money (ATM) options comprise a smaller but highly critical subset of 2.2% (17,247 observations). This com-

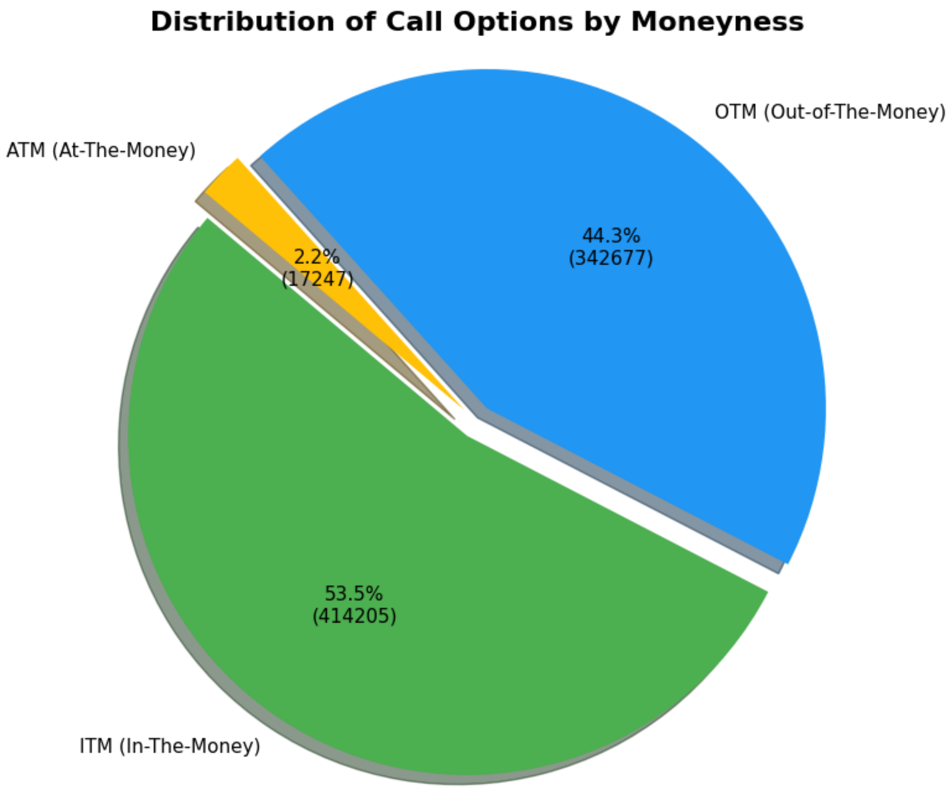

Figure 2.1: Distribution of the consolidated AAPL European Call options dataset by moneyness regimes (ITM, ATM, and OTM).

prehensive coverage ensures that the neural network encounters sufficient training examples to approximate the non-linear option pricing surface across all structural risk boundaries.

An analysis of this historical distribution also revealed that major corporate events—especially large stock splits—and sharp macroeconomic shifts in early 2023 created structural breaks in the pricing data. Specifically, the relationship between the stock price and option premiums showed unusual anomalies. Leaving these uncorrected would introduce a heavy bias into the neural network's loss function. To keep the model accurate and relevant, the data architecture was engineered to focus on a dynamic post-2023 framework.

2.2 Automated Data Collection and API Integration

To keep the data fresh and capture market shifts in real time, an automated collection pipeline was engineered. This sub-system utilizes the `yfinance` library to pull daily end-of-day snapshots of the full Apple Inc. options chain directly from the Nasdaq exchange. This script ensures a steady flow of current market views to expand the historical core.

Since the historical options records from the OptionsDX platform terminate on December 29, 2023, a decision was made to obtain fresh, up-to-date market data through daily parsing via a professional API integration. To achieve superior historical depth, backtest multi-step horizons, and fill potential gaps in the 2025 trading intervals, high-frequency options snapshots were collected from the Massive platform (formerly Polygon.io) using the specialized `RESTClient` library. This setup allowed the retrieval of raw, uncompressed market states at the exact moment of trade execution. The basic initialization script for the developed API layer is shown below:

```
from polygon import RESTClient
client = RESTClient(api_key="YOUR_API_KEY")
```

Combining these collection streams yielded a large master dataset with about 800,000 unique European Call option rows. This dataset covers a wide range of moneyness levels, including deep In-the-Money (ITM), At-the-Money (ATM), and deep Out-of-the-Money (OTM) contracts. For each observation, a vector of core financial metrics was mapped: the current stock spot price (S_t), the contract strike price (K), the exact time left to expiration ($T-t$), the bid-ask spread margins with the last trade price, and the annualized risk-free interest rate (r) taken from US Treasury yield curves.

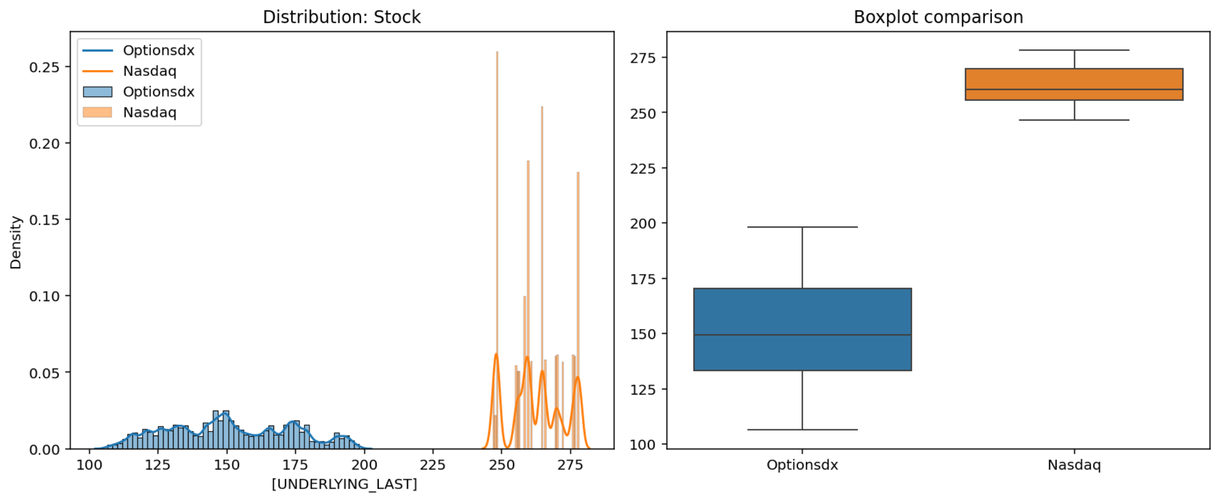

Figure 2.2: Distribution and boxplot comparison of the underlying asset price (S_t) between the historical OptionsDX dataset and the freshly collected Nasdaq API data.

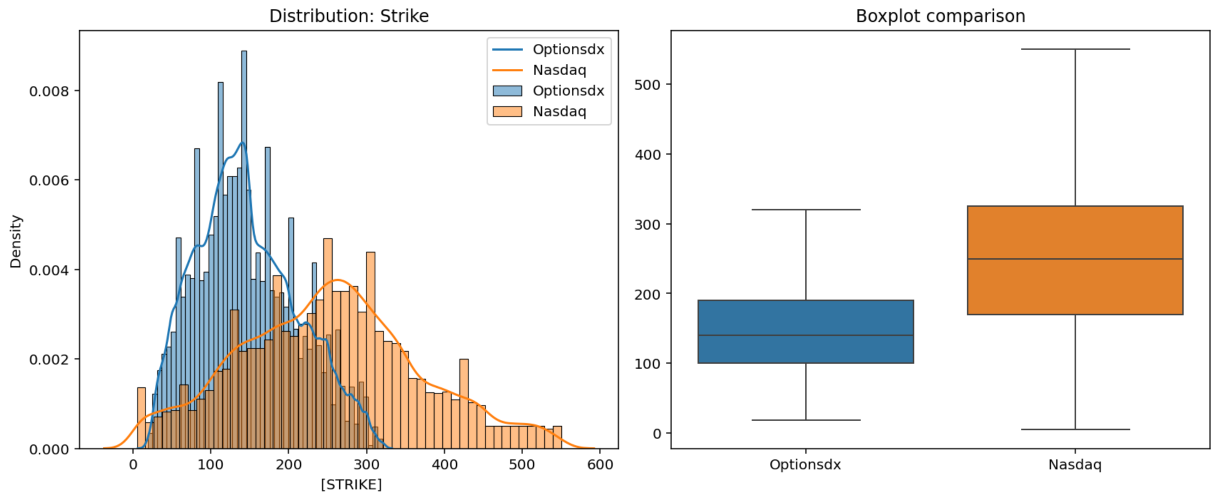

Figure 2.3: Distribution and boxplot comparison of contract strike prices (K) demonstrating the expanded strike boundaries in the active Nasdaq data stream.

To analyze the structural shifts introduced by combining these data sources, an exploratory statistical comparison was conducted. As visualized in Figure 2.2, a clear structural jump is observed in the underlying asset price distribution. The historical OptionsDX data is heavily concentrated within the lower \$100 to \$200 range, whereas the newly parsed Nasdaq API stream reflects a significantly higher price regime, clustering tightly between \$245 and \$280.

Similarly, Figure 2.3 illustrates the distributional differences in the contract strike prices (K) across the two subsets. While the historical OptionsDX baseline mostly contains lower strike prices skewed towards the \$100–\$200 zone, the active Nasdaq stream expands into significantly higher strike regimes, reaching up to \$550. This expansion was necessary to accommodate the massive long-term appreciation of the underlying equity and ensure that the neural network learns to generalize across expanded financial scales without introducing target distortion.

2.3 Modeling of Volatility Dynamics

A primary vulnerability of traditional pricing models is their reliance on constant or simple volatility values. Real financial markets exhibit volatility clustering, where large price shocks follow large shocks, and small changes follow small changes. To capture this behavior and give the neural network a better measure of risk, a Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model was added to the feature engineering steps.

The return changes and conditional variance of the stock are modeled using a standard $GARCH(1,1)$ framework, which follows these equations: $\mathrm{r}_t = \mu + \varepsilon_t, \quad \varepsilon_t = \sigma_t z_t, \quad z_t \sim \mathcal{N}(0,1)$ $\sigma_t^2 = \omega + \alpha\varepsilon_{t-1}^2 + \beta\sigma_{t-1}^2$ where r_t is the log return of the stock price at time step t, μ is the average return, and ε_t is the random shock term scaled by the conditional standard deviation σ_t. The variance equation uses three key parameters: the baseline constant variance ω, the ARCH coefficient α (how fast the model reacts to recent shocks), and the GARCH coefficient β (how long volatility shocks persist over time).

Empirical tests confirm that a $GARCH(1,1)$ setup is sufficient for financial stock returns. It captures volatility clusters well without the risk of overfitting that comes with more complex parameters. This conditional volatility metric σ_t is dynamically generated for each sequence and passed straight into the input layer of the recurrent neural network. This allows the architecture to map the links between shifting volatility and option premiums. Also, instead of training the network on raw nominal prices, the pipeline converts asset values into a continuous moneyness ratio (S_t/K). This abstraction decouples the network weights from absolute price scales, preventing overfitting and helping the model generalize better across different price levels.

2.4 Data Preprocessing, Cleaning, and Filtering

Before training the network, the raw data was cleaned and filtered. This step removes market noise and ensures that the historical sequences are consistent.

2.4.1 Liquidity Constraints and Microstructural Filtering

Options chains often contain inactive or illiquid rows, especially for strike prices far from the current stock price. To remove these uninformative points, an automated liquidity filter was used. Option records with a last trade price (C_{last}) of \$0.10 or less, or those with zero trading volume, were deleted from the dataset. This step prevents tiny price fractions from disrupting the training process, helping the model learn faster and more stably.

2.4.2 Temporal Alignment and Sequential Grouping

To create the sequential inputs for the network, quote timestamps and contract expiration dates were matched. This alignment allowed the calculation of the exact Days to Expiration (DTE) and the removal of rows with zero or negative time left.

Next, the data was split into separate sequences, sorted by date and grouped by strike price (K) and expiration date (T). For the model to process these segments smoothly, we required each contract group to have at least 19 consecutive daily rows. This number matches the sum of the 14-day look-back window and the 5-day forecast horizon. To keep the training stable, only contract groups with 50 or more continuous historical days were kept.

2.4.3 Feature Scaling and Normalization

The final step in the data pipeline scales the inputs. Neural networks are highly sensitive to the scale of input features. To ensure stable calculations, all features were mapped into a $(0, 1)$ range using a `MinMaxScaler`. This prevents large values, such as the absolute Apple stock price, from overshadowing smaller features during training. The transformation applied to each feature component x is written as:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{2.1}$$

where x_{min} and x_{max} are the minimum and maximum values of the feature in the training set. This scaling ensures that all inputs contribute fairly to the network updates.

3. ADVANCED LSTM ARCHITECTURE AND FINAL RESULTS

To evaluate data-driven option pricing in practice, a transition from standard sequential neural networks to an optimized multi-input topology was executed. Financial time series contain completely different types of data: historical non-linear price trajectories, autoregressive volatility parameters, and fixed future contract constraints. To combine these features, a multi-horizon neural network was constructed utilizing the Keras Functional API. This setup tracks historical sequences and future dependencies inside a single execution graph.

3.1 Multi-Input LSTM Architecture and Network Topology

It should be emphasized that the final configuration of the proposed neural network represents the culmination of a highly iterative development process. The optimization lifecycle involved benchmarking several intermediate setups, including baseline feedforward networks and single-input sequential models. While a deep analysis of those early training runs falls outside the scope of this work, their empirical feedback helped isolate the optimal network components. Ultimately, the final consolidated architecture was engineered as a dual-branch system with two independent input tracks that merge just before the final output layer. This configuration keeps the network flexible while keeping it strictly tied to the actual contract expiration limits.

3.1.1 Temporal Input Layer and Feature Set

The primary branch of the architecture is designed to ingest a historical sequence matrix. If w is the length of the rolling look-back window, the main temporal sequence is formally defined as:

$$X_t = \{x_{t-w+1}, \ldots, x_t\}, \quad x_t \in \mathbb{R}^d, \tag{3.1}$$

where the historical window size is set to exactly fourteen trading days ($w = 14$). The parameter $d = 5$ is the number of features inside each time step.

The engineered feature set includes five metrics: the historical option price series (C_t), the time left to maturity in years (τ_t), and the moneyness ratio (S_t/K_t), which maps the stock price against the strike price to capture the non-linear shape of option payoffs. Additionally, the feature set integrates the conditional volatility (σ_t) calculated from the GARCH(1,1) model, and a normalized time encoding feature that rescales the timeline inside each sliding window between 0 and 1 to stabilize the recurrent pattern recognition.

3.1.2 Recurrent Encoding Layer

To track temporal dependencies, volatility clustering, and short-term momentum from the sequence X_t, the matrix is passed directly to a Long Short-Term Memory (LSTM) layer configured with 32 hidden units. The hidden state transformation is written as:

$$h_t = \text{LSTM}(X_t), \quad h_t \in \mathbb{R}^{32}. \tag{3.2}$$

Since recurrent models easily overfit on noisy financial data, a regularization layer was applied right after the LSTM block. A Dropout layer with a 0.3 rate drops random weights during training, creating the regularized state vector $\tilde{h}_t$:

$$\tilde{h}_t = \text{Dropout}(h_t). \tag{3.3}$$

3.1.3 Contextual Branch and Multi-Horizon Output

Simultaneously, the secondary branch of the architecture processes a future-oriented vector containing the exact expiration paths across the forecast horizon. This conditioning sequence is a vector of normalized time values for the next $H = 5$ trading days:

$$D = (\tau_{t+1}, \tau_{t+2}, \ldots, \tau_{t+5}). \tag{3.4}$$

The vector D is flattened and routed to a concatenation layer, where it merges with the regularized historical hidden state vector $\tilde{h}_t$. This step builds a combined feature matrix with a size of $32 + 5 = 37$:

$$X_{\text{combined}} = [\tilde{h}_t \parallel D]. \tag{3.5}$$

This structure forces the network to calculate the time decay ($Theta$) for each individual prediction step. Finally, the combined vector X_{combined} goes through a fully connected dense layer with a linear activation to output the final multi-step price forecast:

$$\hat{C} = (\hat{C}_{t+1}, \hat{C}_{t+2}, \ldots, \hat{C}_{t+5}), \tag{3.6}$$

where $\hat{C}_{t+i}$ is the predicted call option price at the future step $t+i$.

3.2 Model Training and Convergence Dynamics

The network parameters were optimized by minimizing the Mean Squared Error (MSE) loss function across all forecast steps at the same time:

$$\mathcal{L} = \frac{1}{H}\sum_{i=1}^{H}(C_{t+i} - \hat{C}_{t+i})^2, \tag{3.7}$$

where C_{t+i} is the actual market price of the option contract at step $t+i$. The training routine utilized the Adam optimizer, which updates individual learning rates using the gradient moments.

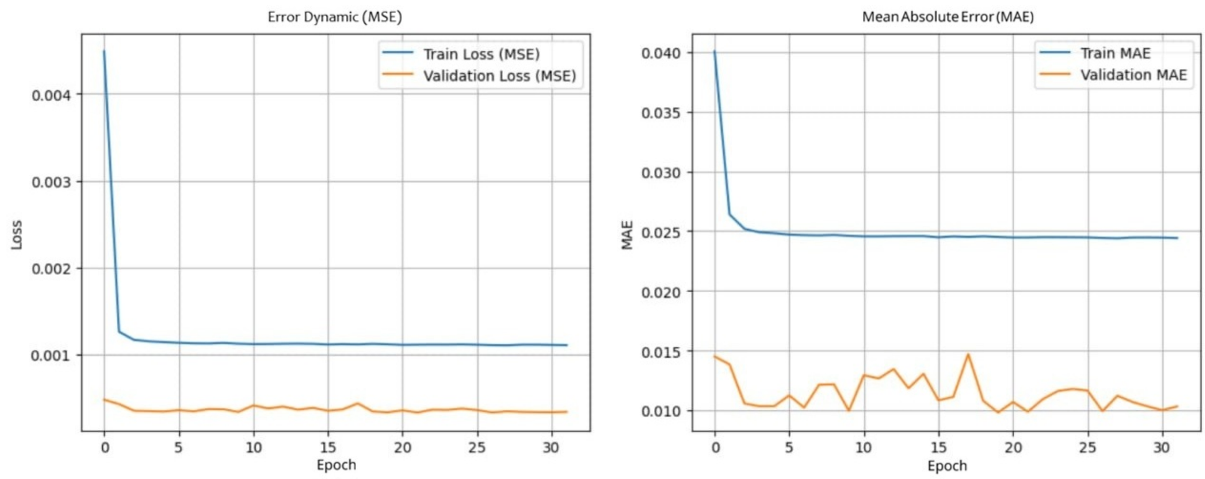

Figure 3.1: Training and Validation convergence trajectories across optimization epochs for Mean Squared Error (MSE) and Mean Absolute Error (MAE).

The training limit was set to 100 epochs, augmented by an early stopping rule based on validation loss stabilization. As shown in Figure 3.1, stable numerical convergence was achieved at exactly the 32nd epoch. The loss lines show a steady exponential drop without any unstable behavior. Notably, the validation errors stay slightly below the training errors (validation MSE stops near 0.0003, while training MSE reaches 0.0011). This occurs because the dropout layer only filters weights during training runs. This convergence path shows excellent generalization and confirms that the setup does not suffer from overfitting.

3.3 Comparative Performance Analysis and Multi-step Forecasting

To evaluate the predictive accuracy of this optimized multi-input LSTM network, its performance was benchmarked against three implementations of the Black-Scholes-Merton model: a classic setup using 20-day historical volatility, an advanced version using GARCH(1,1) conditional volatility, and an ideal version using full realized volatility. The final metrics compiled across the 5-day forecasting horizon (t_1 to t_5) are detailed in Table 3.1.

Table 3.1: Predictive Performance Metrics Across the 5-Day Forecast Horizon

Step	Model	MAE	RMSE	MAE rel	RMSE rel
t_1	LSTM	0.8196	1.6153	0.0206	0.0405
	BS	2.2638	4.4377	0.0568	0.1113
t_2	LSTM	1.3112	2.3120	0.0328	0.0579
	BS	2.2590	4.4295	0.0566	0.1109
t_3	LSTM	1.5562	2.7890	0.0389	0.0697
	BS	2.2546	4.4236	0.0564	0.1106
t_4	LSTM	1.7773	3.2183	0.0444	0.0803
	BS	2.2476	4.4173	0.0561	0.1102
t_5	LSTM	2.0259	3.6206	0.0505	0.0902
	BS	2.2465	4.4179	0.0560	0.1101

3.3.1 Analysis of Short-Term Forecasting Performance

At the first prediction step (t_1), the recurrent network shows a clear advantage over the analytical baseline. The optimized LSTM model reaches a Mean Absolute Error (MAE) of \$0.8196 and an RMSE of \$1.6153. Meanwhile, the Black-Scholes model with GARCH volatility gives an MAE of \$2.2638 and an RMSE of \$4.4377. This means the proposed model reduces the pricing error immediately by 63.8%. This performance gap can be attributed to the LSTM's ability to learn latent microstructural traits and complex pricing surfaces straight from real asset data. On the other hand, the Black-Scholes formula cannot escape its rigid log-normal assumptions, even for short-term predictions.

3.3.2 Error Progression and Long-Term Predictive Stability

As the forecast horizon expands from t_2 through t_5, a clear divergence in error propagation mechanics can be observed between the two modeling setups. Across the 5-day horizon, the Black-Scholes model shows a flat error profile, where its MAE hovers rigidly around the \$2.25 mark (2.2638 at t_1 and 2.2465 at t_5). This behavior is mathematically consistent with the status of the BSM model as a static analytical framework; since it evaluates prices using raw daily inputs at each specific point instead of building on past forecasts, it does not accumulate tracking errors over time.

In contrast, the optimized LSTM network exhibits a gradual, expected rise in its error metrics. The MAE moves from \$$1.3112$ at t_2 to \$$2.0259$ on the fifth day (t_5). This increase is a natural consequence of the cumulative accumulation of stochastic noise inherent in multi-step ahead financial forecasting. Crucially, however, the rate of error propagation remains linear and tightly controlled. This proves the model's structural stability and confirms that the network does not suffer from the sudden exponential error spikes common in recurrent models. Even at the furthest forecast boundary (t_5), where the network faces its maximum uncertainty (MAE = \$2.0259), the LSTM still completely outperforms the Black-Scholes results at any point of the testing window. Throughout the entire cycle, the relative error (MAE_{rel}) of the LSTM remains constrained under 5.05$\%$, confirming that the dual-branch configuration generalizes successfully.

3.4 Web Interface Deployment and Future Extensions

To display the practical utility of the trained deep learning framework, a prototype web interface was developed in Python utilizing the Streamlit framework and deployed. This interactive environment allows for the immediate visualization of option pricing trajectories, historical factual backtesting, and model comparison graphs. The graphical user interface layout is illustrated in Figure 3.2.

The current interface works by processing local files and pre-calculated sequences to render pricing curves smoothly. However, it is critical to note that building a functional real-time option calculator requires a fundamentally different architecture. A production-grade real-time system relies on a continuous client-server configuration, where user-driven inputs trigger automated API parsing and immedi-

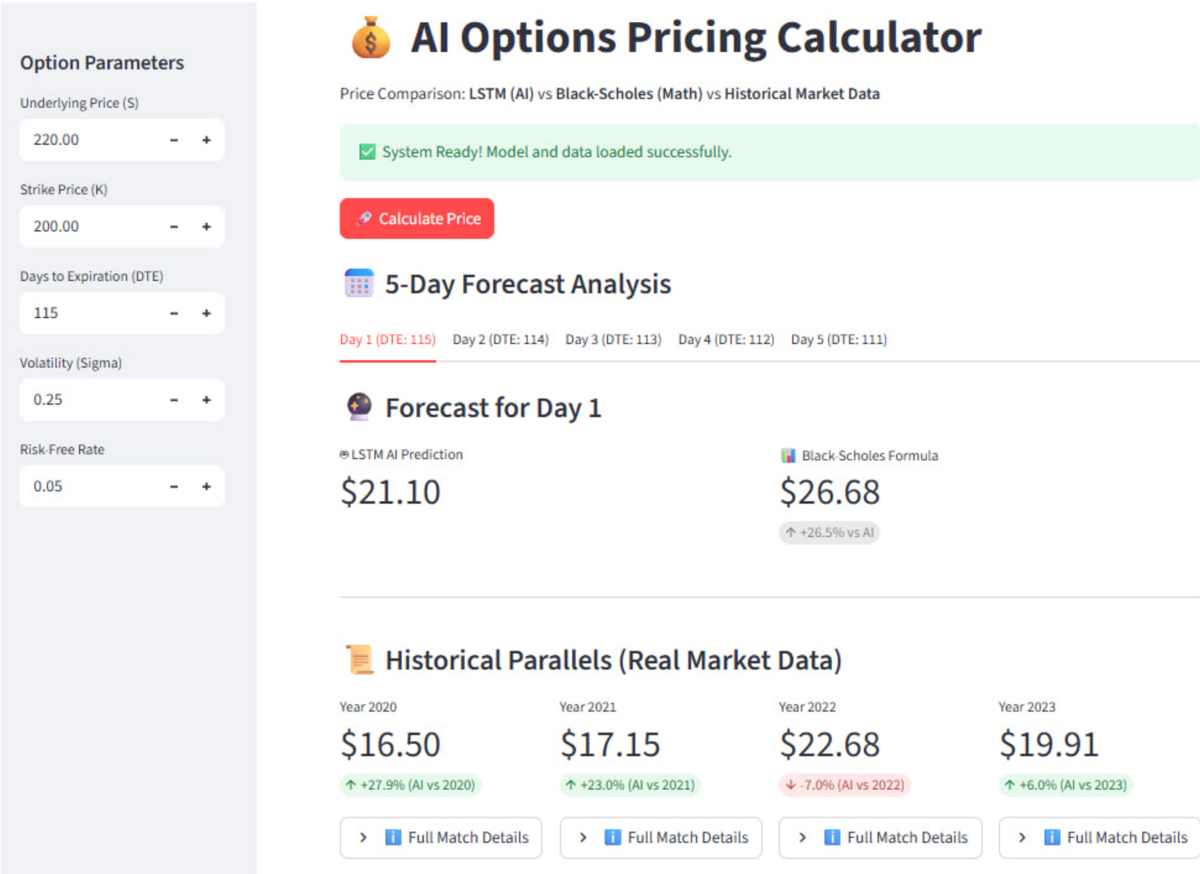

Figure 3.2: Layout of the analytical prototype dashboard showcasing multi-step option price forecasting and model comparison visualization.

ate server requests. These asynchronous requests must be processed by a dedicated remote server hosting the trained Keras execution graph to yield low-latency updates. Consequently, the transition from a local visualization setup to an active, server-driven real-time calculator is designated as a key direction for future research.

CONCLUSIONS

This thesis provides a comprehensive empirical and theoretical investigation into data-driven derivative pricing frameworks, evaluating the performance of non-parametric architectures against classical financial models under non-stationary market regimes. The analytical and quantitative outcomes of this study confirm that the main research goal has been successfully achieved by building an optimized multi-input Long Short-Term Memory (LSTM) network and validating its forecasting superiority over traditional benchmarks.

Based on the initial theoretical tasks of the research, the investigations detailed in Section 1 confirmed that while the analytical Black-Scholes equation remains an elegant mathematical baseline, its strict reliance on constant asset volatility and log-normal return distributions fails to capture real-world option characteristics. This limitation manifests empirically as the volatility smile or volatility skew. To address these drawbacks, the selection and justification of a non-parametric recurrent architecture were covered in Subsection 1.3, proving its capacity to map complex, high-dimensional pricing surfaces directly from continuous market data flows without requiring restrictive stochastic assumptions.

To bridge this theoretical gap with empirical data, a robust data harvesting and feature engineering infrastructure was constructed, as described in Section 2. Utilizing a master dataset of approximately 800,000 Apple Inc. (AAPL) option observations, microstructural noise and illiquidity artifacts were eliminated through the strict filtering rules outlined in Subsection 2.4. A central component of this data pipeline is the econometric layer developed in Subsection **??**, where a $GARCH(1, 1)$ framework was integrated to model time-varying variance clustering. Feeding this conditional volatility metric alongside a normalized moneyness ratio (S_t/K) into the temporal branch of the final network architecture prevented target scale overfitting and allowed the model to adapt dynamically to shifting volatility regimes.

The empirical benchmarking conducted in Subsection 3.3 yields strong quantitative validation of the hybrid network's multi-step predictive capabilities. At the immediate short-term forecasting horizon (t_1), the optimized multi-input LSTM model achieved a Mean Absolute Error (MAE) of \$0.8196 and a Root Mean Squared Error (RMSE) of \$1.6153. Compared to the most competitive analytical variant—the

Black-Scholes model operating with GARCH volatility (\$2.2638)—the proposed architecture achieved an immediate 63.8% reduction in pricing error. Furthermore, over an extended five-day horizon, the rolling-window iterative prediction loop demonstrated excellent structural stability. Although the accumulation of stochastic noise caused the LSTM's error to rise linearly to an MAE of \$2.0259 at t_5, its predictive accuracy completely outperformed the static Black-Scholes calculations at every tested forecasting step.

Finally, to fulfill the practical software objectives of the study, the trained predictive model was transitioned into a functional application environment, as detailed in Subsection 3.4. A user-friendly analytical option pricing calculator was successfully developed in Python using the Streamlit framework and deployed. This web interface allows for immediate option chain visualization, historical backtesting, and interactive comparative analysis of pricing errors, effectively bridging complex machine learning outputs with operational risk management tools.

In conclusion, the predictive framework developed in this study represents a reliable and mathematically sound tool for derivatives pricing under realistic, volatile market conditions. The ability of the recurrent topology to organically reconstruct the latent volatility surface from sequential data blocks provides a significant advantage for modern financial engineering and asset risk assessment.

REFERENCES

[1] Pasternak O. O. *Neural Network for Option Pricing* // Materials of the XIV All-Ukrainian Scientific Conference of Young Mathematicians. — Kyiv, 2026. — P. 152.

[2] Culkin R., Das S.R. *Machine Learning in Finance: The Case of Deep Learning for Option Pricing* // Journal of Investment Management. — 2017. — Vol. 15, No. 4. — P. 92–100.

[3] Black F., Scholes M. *The pricing of options and corporate liabilities* // Journal of Political Economy. — 1973. — Vol. 81, No. 3. — P. 637–654.

[4] Ruf J., Wang W. *Neural networks for option pricing and hedging: a literature review* // Journal of Computational Finance. — 2020. — Vol. 24, No. 1. — P. 1–46.

[5] Hochreiter S., Schmidhuber J. *Long short-term memory* // Neural Computation. — 1997. — Vol. 9, No. 8. — P. 1735–1780.

[6] Malliaris M., Salchenberger L. *A Neural Network Model for Estimating Option Prices* // Journal of Applied Intelligence. — 1993. — Vol. 3. — P. 193–206.

[7] Casteli F., Leonenko N., Shchestyuk N. *Student-like models for risky asset with dependence* // Stochastic Analysis and Applications. — 2017. — Vol. 35, No. 4. — P. 452–464.

[8] Shchestyuk N., Drin S., Tyshchenko S. *Risk evaluating for subdiffusive option price model with gamma subordinator* // Mathematical and Statistical Methods for Actuarial Sciences and Finance (MAF 2024). — 2024. — P. 286–291.

[9] Shchestyuk N., Tyshchenko S. *Subdiffusive option price model with inverse Gaussian subordinator* // Modern Stochastics: Theory and Applications. — 2025. — Vol. 12, No. 2. — P. 136–152.

[10] Donatien H., Leonenko N. N. *Option pricing in illiquid markets: A fractional jump–diffusion approach* // Journal of Computational and Applied Mathematics. — 2021. — Vol. 381. — Art. 112995.

[11] Shchestyuk N., Sluchynskyi D., Serduk F. *Subdiffusive models with local volatility for illiquid markets* // Bulletin of Taras Shevchenko National University of Kyiv. Series: Physics & Mathematics. — 2025. — (in press).

[12] Hutchinson J. M., Lo A. W., Poggio T. *A Nonparametric Approach to Pricing and Hedging Derivative Securities via Learning Networks* // The Journal of Finance. — 1994. — Vol. 49, No. 3. — P. 851–889.

[13] Amornwattana S., Enke D., Dagli C. H. *A hybrid option pricing model using a neural network for estimating volatility* // Proceedings of the Engineering Management Conference. — 2002. — Vol. 1. — P. 507–511.

[14] Zhang Z., Chen Y., Li X. *Pricing options with a new hybrid neural network model* // Expert Systems with Applications. — 2024. — Vol. 251. — Art. 123979.

[15] Pimentel R., Risstad M. et al. *Option pricing with deep learning: a long short-term memory approach* // Decisions in Economics and Finance. — 2025. — Vol. 48. — P. 1–30.

[16] Li W. *Application of Machine Learning in Option Pricing: A Review* // Advances in Economics, Business and Management Research. — 2022. — Vol. 215. — P. 209–214.

[17] Yao J., Li Y., Tan C.L. *Option price forecasting using neural networks* // Omega. — 2000. — Vol. 28, No. 4. — P. 455–466.

[18] Shchestyuk N., Tyshchenko S. *Option Pricing and Stochastic Optimization* // Stochastic Processes, Statistical Methods, and Engineering Mathematics. SPAS 2019. Springer Proceedings in Mathematics and Statistics. — 2022. — Vol. 408. — P. 651–669.