ENSA
École Nationale des Sciences Appliquées
Khouribga

Report

End-of-year project
1st year Master's Big Data and Decision Support

DEEP LEARNING IN FINANCE

Prepared by :
HARRACH RAS Salma
BELLA ABDELOUAHAB

Supervised by :
Pr. EL HADFI Youssef

Defended on 09/09/2024, before the jury :
Pr. HAFIDI Imad
Pr. JBEL Mouad
Pr. JABRANE Mourad

Abstract

Deep learning in finance is transforming financial data analysis by enabling the processing of large and complex datasets. It is widely used in areas such as price prediction, algorithmic trading, fraud detection, and risk management. Deep neural networks, particularly RNNs and LSTMs, are highly effective in forecasting financial time series, while autoencoders and generative adversarial networks (GANs) are used for anomaly detection. Although deep learning offers significant advantages, such as increased accuracy and the ability to model complex relationships, it also presents challenges, including model interpretability and the need for massive datasets.
Keywords: Deep Learning-Finance Time Series Forecasting-Option Pricing-Sentiment Analysis-Risk Assessment-Recurrent Neural Networks (RNNs)-Long Short-Term Memory (LSTM)-Convolutional Neural Networks (CNNs)-Portfolio Management .

Résumé

Le deep learning en finance révolutionne l'analyse des données financières en permettant de traiter de vastes ensembles de données complexes. Il est largement utilisé dans divers domaines tels que la prévision des prix, le trading algorithmique, la détection de fraude, et la gestion des risques. Les réseaux de neurones profonds, notamment les RNN et les LSTM, sont particulièrement efficaces pour la prévision des séries temporelles financières, tandis que les autoencodeurs et les réseaux adverses génératifs (GANs) sont utilisés pour la détection des anomalies. Bien que le deep learning offre des avantages significatifs, tels qu'une précision accrue et la capacité à modéliser des relations complexes, il présente aussi des défis, notamment en matière d'interprétabilité des modèles et de besoins en données massives.

Mots clés : Apprentissage profond-Finance-Prévision des séries temporelles-Évaluation des options-Analyse des sentiments-Évaluation des risques-Réseaux de neurones récurrents (RNN)-Mémoire à long court terme (LSTM)-Réseaux de neurones convolutifs (CNN)-Gestion de portefeuille.

ملخص

تقنية Deep Learning في المجال المالي تُحدث ثورة في تحليل البيانات المالية من خلال تمكين معالجة مجموعات كبيرة ومعقدة من البيانات. تُستخدم بشكل واسع في مجالات متنوعة مثل التنبؤ بالأسعار، التداول الآلي، اكتشاف الاحتيال، وإدارة المخاطر. تُعد الشبكات العصبية العميقة، خاصة شبكات RNN و LSTM، فعالة للغاية في التنبؤ بالسلاسل الزمنية المالية، بينما تُستخدم الشبكات العصبية التلقائية (Autoencoders) والشبكات الخصومة التوليدية (GANs) لاكتشاف الشذوذ. على الرغم من أن Deep Learning يوفر فوائد كبيرة، مثل دقة أعلى وقدرة على نمذجة العلاقات المعقدة، إلا أنه يواجه تحديات أيضاً، بما في ذلك صعوبة تفسير النماذج والحاجة إلى كميات ضخمة من البيانات.
الكلمات المفتاحية: Deep Learning-المالية-التنبؤ بالسلاسل الزمنية-تسعير الخيارات-تحليل المشاعر-تقييم المخاطر-الشبكات العصبية التكرارية (RNN)-ذاكرة طويلة قصيرة المدى (LSTM)-الشبكات العصبية الالتفافية (CNN)-إدارة المحافظ.

Contents

List of Figures

List of Tables

General Introduction

In recent years, the financial sector has experienced profound transformations driven by advances in artificial intelligence (AI) and machine learning (ML). Among the most impactful developments is the rise of deep learning, a subset of machine learning that enables the modeling of highly complex, non-linear relationships within large datasets. Financial institutions, traders, and analysts are increasingly leveraging these advanced techniques to address challenges such as market volatility, risk management, and investment decision-making.

This report focuses on the application of deep learning in finance, with particular emphasis on its use in time series forecasting, option pricing, sentiment analysis, and risk assessment. These areas represent critical components of financial decision-making and are traditionally difficult to model due to the inherent complexity and unpredictability of financial markets.

The objective of this study is to explore how deep learning models, including Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs), Convolutional Neural Networks (CNNs), and more recent architectures such as transformers and autoencoders, can be applied to real-world financial problems. By leveraging their ability to extract intricate patterns from data, these models provide enhanced predictive power and offer new opportunities for optimizing trading strategies, improving portfolio management, and mitigating financial risks.

This introduction outlines the structure of the report, beginning with an in-depth review of deep learning principles, followed by the presentation of specific financial applications, including detailed case studies. The final chapters address both the challenges and ethical considerations of using AI-driven models in finance, highlighting the importance of transparency, interpretability, and the management of biases in automated systems. Through this comprehensive analysis, the report aims to provide valuable insights into the transformative role of deep learning in modern finance.

Chapter 1

Deep Learning

We begin by introducing the general theoretical framework of deep learning along with several specific details.

1.0.1 Architecture

Deep learning is a form of machine learning. Machine learning involves using data to train a model, and then using the trained model to make predictions from new data. The fundamental problem of machine learning is to find a predictor for an output Y given an input X. A learning machine is defined as an input-output mapping $Y = F(X)$, where the input space is high-dimensional, and we write

$$Y = F(X) \quad \text{where} \quad X = (X_1, \dots, X_p),$$

and a predictor is denoted as $\hat{Y}(X) := F(X)$. The output T can be continuous, discrete as in classification, or mixed. For example, in a classification problem, we must learn a mapping $F : X \to Y$, where $Y \in \{1, \dots, K\}$ indexes the categories.

As a form of machine learning, deep learning trains a model on data to make predictions, but it is distinguished by the passage of learned features from the data through different levels of abstraction. Raw data is input at the lower level, and the desired output is produced at the higher level, the result of learning through many layers of transformed data. Deep learning is hierarchical in the sense that, at each layer, the algorithm extracts features into factors, and the factors of a deeper level become the features of the next level.

1.0.2 Neural Networks

Neural networks are a powerful tool for learning complex functions in high dimensions. A basic building block is the perceptron, or neuron, which takes a weighted sum of inputs, passes it through a nonlinear activation function, and produces an output. Specifically, consider a set of inputs $X_1, X_2, \ldots, X_p$. A perceptron computes the output

$$Y = \sigma\left(\sum_{j=1}^{p} w_j X_j + b\right),$$

where $\sigma(\cdot)$ is the activation function, w_j are the weights, and b is the bias term.

The nonlinear activation function $\sigma(\cdot)$ is essential for allowing the network to learn complex, nonlinear mappings. Common choices for the activation function include the sigmoid function, hyperbolic tangent, and rectified linear unit (ReLU):

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad \text{(sigmoid)},$$

$$\sigma(z) = \tanh(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} \quad \text{(hyperbolic tangent)},$$

$$\sigma(z) = \max(0, z) \quad \text{(ReLU)}.$$

A neural network is constructed by stacking multiple layers of perceptrons. The output of each layer becomes the input to the next layer. Let $Y^{(l)}$ be the output of the l-th layer. Then the output of the l-th layer is

$$Y^{(l)} = \sigma\left(W^{(l)} Y^{(l-1)} + b^{(l)}\right),$$

where $W^{(l)}$ and $b^{(l)}$ are the weights and biases for the l-th layer.

In a fully connected neural network, each neuron in a layer is connected to every neuron in the previous layer. The final output layer produces the prediction $\hat{Y}$.

1.0.3 Training Neural Networks

Training a neural network involves finding the weights and biases that minimize a loss function, typically the mean squared error for regression problems or cross-entropy for classification problems. The loss function is minimized using gradient descent and its variants. The key idea is to compute the gradient of the loss function with respect to the weights and biases and iteratively update them to reduce the loss.

Let $\mathcal{L}(Y, \hat{Y})$ be the loss function. The gradient descent update for the weights is given by

$$W^{(l)} \leftarrow W^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial W^{(l)}},$$

where η is the learning rate. Similarly, the biases are updated as follows:

$$b^{(l)} \leftarrow b^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial b^{(l)}}.$$

Backpropagation is the algorithm used to efficiently compute the gradients. It involves propagating the gradient of the loss function from the output layer back to the input layer, updating the weights and biases at each layer.

1.0.4 Deep Learning Models

Deep learning models differ by network architecture. Common deep learning models include:

- **Feedforward Neural Networks (FNNs)**: The simplest type of neural network, where information flows in a single direction from input to output layer without loops.
- **Convolutional Neural Networks (CNNs)**: Designed to process grid-like structured data, such as images. CNNs use convolutional layers that apply filters to the input to capture spatial hierarchies of features.
- **Recurrent Neural Networks (RNNs)**: Designed for sequential data, where each input depends on previous inputs. RNNs maintain a hidden state that captures information about the sequence.
- **Long Short-Term Memory Networks (LSTMs)**: A type of RNN that solves the vanishing gradient problem in long sequences by introducing memory cells that can retain information over long periods.

1.0.5 Regularization and Optimization

Deep learning models are prone to overfitting due to their high capacity. Regularization techniques are used to prevent overfitting and improve generalization. Common regularization methods include:

- **L2 Regularization**: Adds a penalty proportional to the squared value of the weights to the loss function, encouraging smaller weights.
- **Dropout**: Randomly drops units (and their connections) during training to prevent the network from relying too much on any individual neuron.
- **Early Stopping**: Monitors the model's performance on a validation set and stops training when performance begins to degrade, indicating overfitting.

Optimization methods beyond standard gradient descent include:

- **Momentum**: Accelerates gradient descent by adding a fraction of the previous update to the current update.
- **Adam**: Combines the advantages of Momentum and RMSProp, adapting the learning rate for each parameter.

Chapter 2

Deep Learning in Finance

Deep learning tools have become commodities. In addition to TensorFlow, there are other popular open-source implementations of deep learning, such as MXNet, supported on Amazon Web Services (AWS), and h2o, the latter being a versatile platform that supports many deep learning implementations, including Caffe, Torch, and Theano. Deep learning tools are supported in many programming languages like Python and R. Consequently, while the tools themselves have become commodities, the ideas and models of deep learning are not, and they require creative thinking to define the objectives of the neural network (NN) and the structure to use. This is where deep learning becomes an art.

There are many conditions that make deep learning an important architecture in the field of finance. First, the availability of large amounts of data, often streamed at rates that are unparalleled in other fields. Second, several financial applications depend on speed, and the advent of efficient hardware in deep learning makes it possible to achieve the response levels necessary to make trading algorithms viable. Third, much of finance involves pattern recognition using data, where various inputs are modeled to predict outputs. For example, predicting stock markets can be based on numerous variables (data streams on stock prices, interest rates, volatilities, etc.). Another example is found in retail banking, where customers are characterized by a multitude of variables to determine which products to offer them or to calculate their retention probabilities. We note that this pattern recognition on large amounts of data is analogous to the ImageNet problem. Therefore, deep learning architectures that can learn to recognize an image can be directly used to learn to recognize (for example) signatures in the stock market that predict the direction of an index. Or they can be used to train a model to learn how the market values options.

What does deep learning allow us to discover that is not possible with standard econometric models? The answer is "non-linearities." Most current econometric models are linear functions, or simple transformations of linear functions. However, the relationship between inputs and outputs can be extremely nonlinear. This is where the deep neural network excels. As the data passes from one layer to another, it is transformed into new data, and layers of non-linearity are gradually removed. This suggests that a deep neural network can learn almost any function with a high degree of precision. Thus, there is a wide range of applications in finance to which deep learning can be applied. [1]

2.1 Option Pricing

2.1.1 Definition

Option pricing is the process of determining the price of an option, which is a financial derivative instrument granting the buyer the right, but not the obligation, to buy (call option) or sell (put option) an underlying asset at a predetermined price, known as the strike price, at a future date or before a specified date.

The price of an option, often called the "premium," is influenced by several factors, including:

The price of the underlying asset: If the price of the underlying asset increases, the value of a call option generally increases, while that of a put option decreases, and vice versa.

The strike price: This is the price at which the option can be exercised. For a call option, the lower the strike price relative to the current price of the asset, the more valuable the option is.

Volatility The volatility of the underlying asset is a measure of how much its price fluctuates. Higher volatility generally increases the value of options, as there is a greater chance that the option will become profitable.

Time until expiration: The further away the expiration date, the more valuable the option, as there is more time for the price of the underlying asset to move in favor of the option.

Interest rates: Interest rates can influence the value of an option, especially in stock options. An increase in interest rates can raise the value of call options.

Dividends For stock options, expected dividends also affect option pricing.

The most commonly used option pricing model is the Black-Scholes model, which considers these factors to calculate the theoretical value of an option.

2.1.2 Deep learning for Option Pricing

Deep learning has emerged as a powerful tool for option pricing, offering significant improvements over traditional methods. This synthesis explores various deep learning frameworks and their applications in option pricing, highlighting key insights from recent research.

Improved Accuracy with Deep Learning Models Deep learning models, such as those using LSTM-GRU neural networks, have shown to improve the accuracy of option pricing compared to traditional models like Black-Scholes. Hybrid deep learning models can achieve better pricing accuracy, with some studies reporting up to 94.5 % improvement in mean squared error over the Black-Scholes model .

Handling High-Dimensional Options Deep recurrent neural networks (RNNs) and other deep learning frameworks can efficiently price and hedge high-dimensional American options, addressing the curse of dimensionality and improving computational efficiency. Deep learning methods based on backward stochastic differential equations (BSDEs) are effective for high-dimensional financial derivatives, providing accurate and efficient solutions.

Combining Deep Learning with Traditional Methods Frameworks like DeepOption combine deep learning with traditional parametric methods, using pre-training with simulated data to enhance performance even with imbalanced real option data . Deep reinforcement learning algorithms can produce high-quality hedging policies and accurate prices for complex options, outperforming traditional methods in dynamic environments.

Ensemble Learning and Feature Extraction Models that integrate ensemble learning methods with deep learning structures can improve option pricing by effectively extracting data features and adapting to small and medium datasets.

Deep learning offers substantial advancements in option pricing, providing more accurate and efficient solutions compared to traditional methods. By leveraging various neural network architectures and combining them with traditional techniques, researchers have developed models that handle high-dimensional data, improve computational efficiency, and deliver precise pricing and hedging strategies. These advancements underscore the potential of deep learning to revolutionize the field of financial derivatives.

2.1.3 An Exemple of the use of neural networks to learn the famous Black-Scholes option pricing formula

The Black-Scholes Formula The Black-Scholes formula (1973) and Merton (1973) (BSM) is arguably one of the most cited and used in finance, and has led to numerous variations and extensions. The formula is used to evaluate call and put options and is the solution to a particular partial differential equation (PDE), first derived by Black and Scholes (1973). The existence of a closed-form solution to this PDE is remarkable. The model starts from the assumption of a specific form of movement for the stock price, namely geometric Brownian motion (GBM), and based on the conditional payoff at the option's expiration, derives the PDE that must be satisfied to meet specific economic constraints so that arbitrage is not admissible.[1]

We start with the specific task of learning the Black-Scholes option pricing model from simulated data. If this is possible with high accuracy, it suggests that market data can be used to train an option pricing model that matches market prices better than BSM.[1]

Analysis

Hutchinson *et al.* (1994) explored the use of neural networks to learn Black-Scholes option pricing formula (1973), also developed and analyzed by Merton (1973). Using limited computing power and relatively small neural networks, they demonstrated remarkably good performance in imitating (learning) the following equation from simulated data. We present here the call option pricing equation (C).

$$C = Se^{-qT}N(d_1) - Ke^{-rT}N(d_2)$$

$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r - 0.5\sigma^2\right)T}{\sigma\sqrt{T}}$$

$$d_2 = d_1 - \sigma\sqrt{T}$$

where S is the current stock price, K is the option's strike price, T is the option's expiration, q and r are the annualized dividend and risk-free rates, respectively. Finally, σ is the annualized volatility, i.e., the standard deviation of the stock's return.

To create data to assess how a deep learning network would learn this equation, we simulated a range of call option prices using a range of parameters shown in Table 2.1.

Table 2.1: Range of parameters used to simulate 300,000 call option prices. Strike prices K were chosen to be close to the stock price S, in order to remain realistic.

Parameter	Range
Stock Price (S)	\$10 - \$500
Strike Price (K)	\$7 - \$650
Expiration (T)	1 day to 3 years
Dividend Rate (q)	0% - 3%
Risk-Free Rate (r)	1% - 3%
Volatility (σ)	5% - 90%
Call Option Price (C)	\$0 - \$328

The data was split into two random sets: a training set consisting of 240,000 option prices and a validation set with the remaining 60,000 prices.

Before feeding the prices into the deep learning network, an aspect of the Black-Scholes call option pricing function was leveraged. Indeed, the pricing function is linearly homogeneous in S and K, i.e.,

$$C(S, K) = K \cdot C\left(\frac{S}{K}, 1\right).$$

Thus,

$$\frac{C(S, K)}{K} = C\left(\frac{S}{K}, 1\right).$$

The data was therefore modified by dividing both the stock price S and the call option price C by the strike price K. These normalized data were then fed into the deep learning network to adjust the input variables S, K, T, q, r, σ (all features) to the output prices C.

The features of the deep learning network are as follows: the input size is 6 parameters, passed through 4 hidden layers of 100 neurons each. Neurons in each layer use different activation functions: LeakyReLU, ELU, ReLU, ELU, respectively. The final output layer consists of a single neuron with a standard exponential function

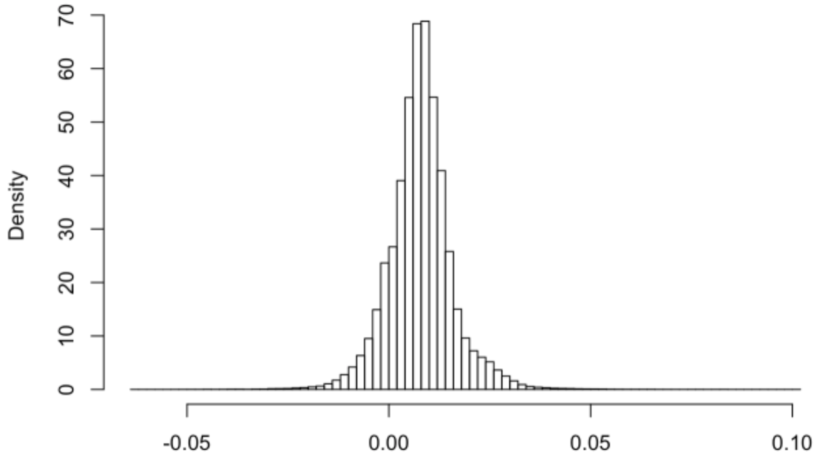
$$\exp(\cdot)$$

to ensure that the network's output is always non-negative, as option prices cannot be negative.

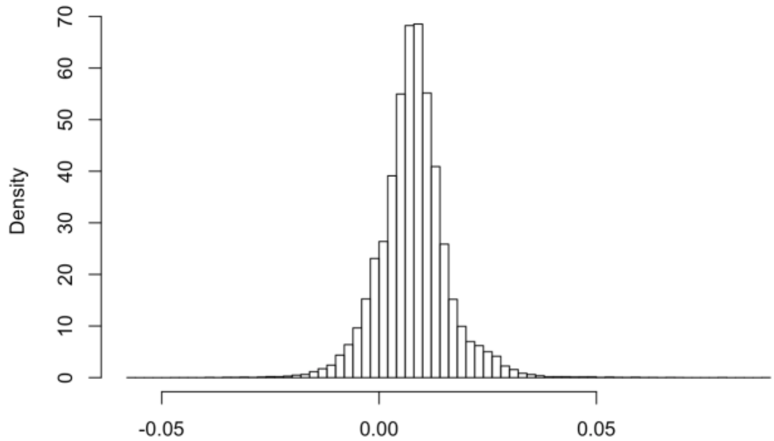
For the deep learning network, simple hyperparameters were chosen: a dropout rate of 25% in each hidden layer to mitigate overfitting, a loss function based on mean squared error (MSE), a batch size of 64, and 10 epochs. In total, 31,101 coefficients (weights) were adjusted for the deep learning model. The model was trained using Google's TensorFlow package. The obtained results are as follows.

In-sample: The root mean square error (RMSE) is 0.0112, which can be compared to the fact that strike prices are all normalized to \$1. Therefore, the average error is $\pm 1\%$ of the strike price. The average percentage pricing error (error divided by option price) is 0.0420, i.e., 4%. The histogram of pricing errors is shown in Panel 1 of Figure 2.1. It is clear that errors are very low. Additionally, a regression of the model's Black-Scholes equation values on the actual values was estimated, and an $\mathrm{R}^2 = 0.9982$, which is very high, was obtained.

Out-of-sample: The root mean square error (RMSE) is also 0.0112, with an average error of $\pm 1\%$ of the strike price. The average percentage pricing error (error divided by option price) is 0.0421, i.e., 4%. The histogram of pricing errors is shown in Panel 2 of Figure 2.1. Again, errors are very low. Additionally, a regression of the model's Black-Scholes equation values on the actual values was estimated, and an $\mathrm{R}^2 = 0.9982$, which is very high, was obtained. Given that the in-sample and out-of-sample test results are identical, no overfitting was observed.



(a) Panel 1: In-Sample Pricing Error



(b) Panel 2: Out-of-Sample Pricing Error

Figure 2.1: Call option pricing error from the adjusted deep neural network.

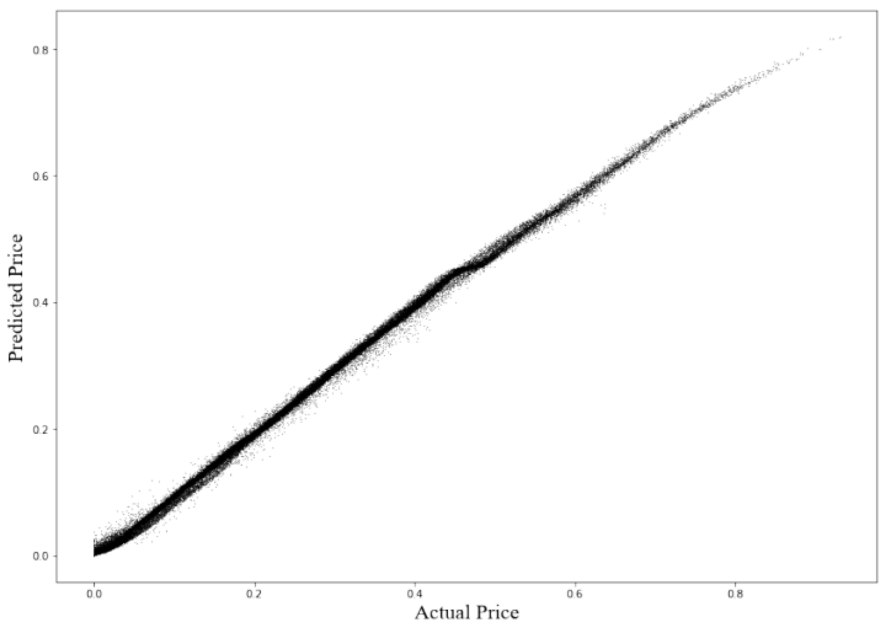


Figure 2.2: Actual Price vs. Predicted Price

Figure 2.2 plots the actual price against the predicted price of each option, producing a narrow line with very few significant deviations, indicating a very limited number of significant pricing errors.[1]

2.2 Price forecasting of financial time series

2.2.1 Forcasting backrground information

Before delving into the details of deep learning models for price forecasting of financial time series, we first provide background information on the price-forecasting task. The following subsections explain the concept of financial time series, the definition of price forecasting, and the workflow of prediction .

2.2.2 Concept of financial time series

A set of data points that show a financial variable's value over time is called a financial time series. These data points, such daily stock prices, monthly interest rates, or quarterly GDP figures, are usually recorded on a regular basis. Financial time series analysis is an important branch of economics and finance that studies these sequences in order to predict future movements, comprehend past trends, and make well-informed judgments.

2.2.3 Definition of price forecasting

Price forecasting of financial time series complies with the rule of point forecasting of general time series. Given a time series y, whose time steps are given as $k \in [1, K]$, a time series forecasting model $f(\cdot)$ estimates the future value $\hat{y}$ at a forecast horizon $H \geq 1$, utilizing the historical values (from time steps k to $k - H_1$, with H_1 being the time lags) of the desired time series and several exogenous time series, which are denoted as $y[k], \ldots, y[k-H_1]$ and $u^T[k], \ldots, u^T[k-H_1]$, respectively . This functional relationship is defined as follows:

$$\hat{y}[k+H] = f\left([y[k], \ldots, y[k-H_1], u^T[k], \ldots, u^T[k-H_1]; \theta\right); k > H_1, \tag{2.1}$$

where the vector θ denotes the model parameters.

In the context of price forecasting, the forecasting model $f(\cdot)$ maps an input sequence to a future value of the asset price. The input sequence comprises two subsequences. The first is $y[k], \ldots, y[k - H_1]$, which refers to a univariate sequence extracted from the financial time series that contains observations of the target variable. The second is $[u^T[k], \ldots, u^T[k - H_1]$, which refers to a multivariate sequence extracted from exogenous time series that contain observations of other related variables, such as technical indicators, sentiment indicators, or macroeconomic variables. Generally, the construction of input sequences for price forecasting is highly empirical.

2.2.4 Workflow of price forecasting

The workflow of price forecasting involves several stages, as illustrated in Figure 2.3. First, relevant historical raw data are collected, and their quality and consistency are checked. Second, the model inputs are prepared through a series of data processing steps. These steps may involve data denoising, which aims to reduce noise from the raw data, or feature engineering, which aims to extract informative feature sets from data. After these steps, the entire time series, univariate or multivariate, is transformed into short sequences using a sliding window approach in a supervised learning scheme. Fixed-length segments of data are extracted as the window (time lag) moved over the entire series. These segments are then labelled with the corresponding observations of the target variable to train the forecasting models. Third, a forecasting model is selected or developed based on specific requirements. This step is largely determined by the data availability, computational resources, and domain knowledge.

Figure 2.4 illustrates the effect of changing the model complexity and data availability on model performance. Generally, a complex model requires a larger training dataset than a simpler model to achieve optimal performance. Utilising a sizable dataset to train a simple model may lead to high bias or underfitting, whereas using a smaller dataset to train a complex model could result in high variance or overfitting. It is crucial to mention that deep learning models are data-driven, and a complex model does not consistently surpass a simpler model in all forecasting undertakings. Individual domain expertise and data handling inclinations also play roles in the model selection. When the model is trained and its performance is deemed satisfactory after the model evaluation, it can be deployed to make future price predictions.[2]

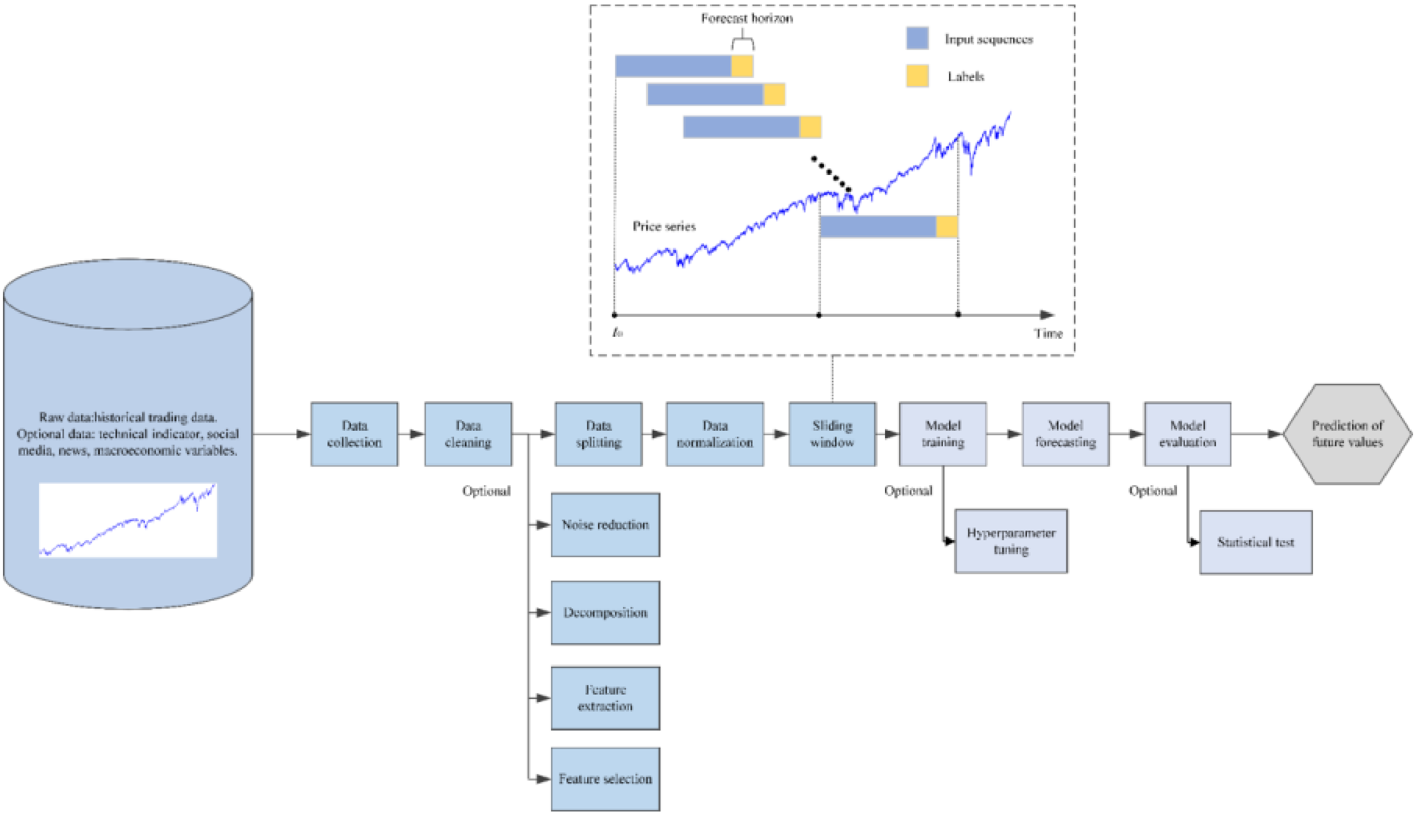


Figure 2.3: Workflow of price forecasting of financial time series.

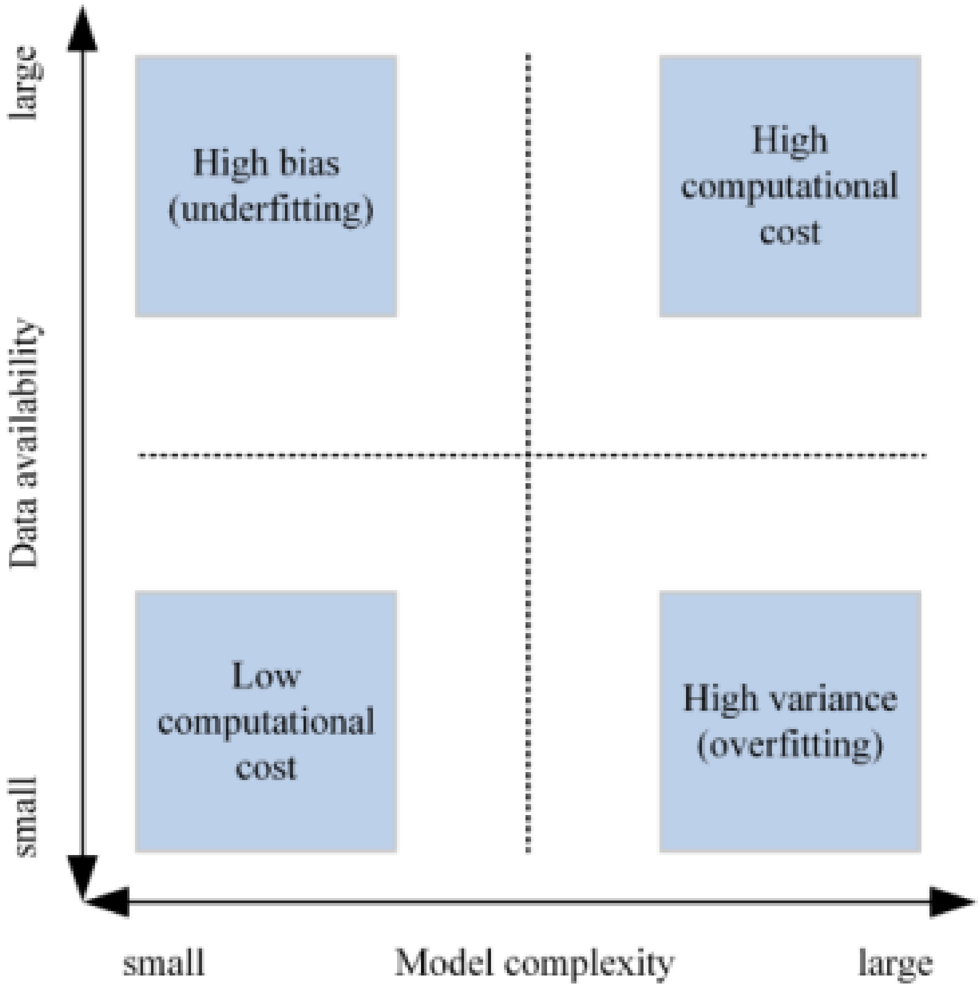


Figure 2.4: Effect of changing model complexity and data availability on model performance.

2.2.5 Deep Learning Models for Price Forecasting

In this section, we present various deep learning models proposed for price forecasting. These models are classified into two categories: individual models and ensemble models, based on their structure and nature. Individual models provide independent and complete forecasts, while ensemble models combine parallel models to generate collective predictions.

Recurrent Neural Networks (RNN) RNNs are particularly suited for modeling time series, such as price forecasting, because they can account for sequential dependencies in the data. However, RNNs may struggle to capture long-term relationships due to the gradient problem.

Long Short-Term Memory (LSTM) LSTMs are a variant of RNNs designed to solve the gradient problem by storing long-term information. They are very popular for time series forecasting, including price forecasting, because they can model long-term dependencies in the data.

Gated Recurrent Unit (GRU) The GRU is another variant of RNNs, similar to LSTM, but with a simpler structure, making it faster to train. It is also effective for time series modeling, while being less complex than LSTMs.

Convolutional Neural Networks (CNN) CNNs, typically used for image processing, can also be applied to price forecasting by extracting local features from the data. For example, they can capture local patterns in time series.

Transformer Transformer models, like the Attention model, are increasingly being used for time series. They can model relationships between different time steps in the series using an attention mechanism, which can be very effective for price forecasting.

Autoencoders Autoencoders are unsupervised neural networks used to learn compressed representations of data. They can be used for price forecasting by learning latent representations of time series.

Temporal Fusion Transformer (TFT) The TFT is an advanced model that combines the advantages of Transformers with multi-scale learning techniques for time series forecasting. It is particularly effective for price forecasting, as it can capture complex and multi-scale relationships in the data.

Figure 2.5 illustrates the distribution of the deep learning models proposed in the reviewed studies. Notably, LSTM networks continue to hold a significant position in time series forecasting, both as key components of individual and ensemble models. This result suggests that capturing the temporal dependencies between data points is a priority for effective time series forecasting. In addition, the application of novel architectures, such as Transformers, GNNs, GANs, and DQNNs, for price forecasting is still in its early stages, mainly owing to the model complexity and high computational resources required for training. Moreover, neural architectures such as DNNs and 1D CNNs exhibit limited generalisation across different price forecasting tasks, despite their lower computational demands for training. Finally, we summarise the advantages and disadvantages of various deep learning models in Table 2.2 and 2.3.[2]

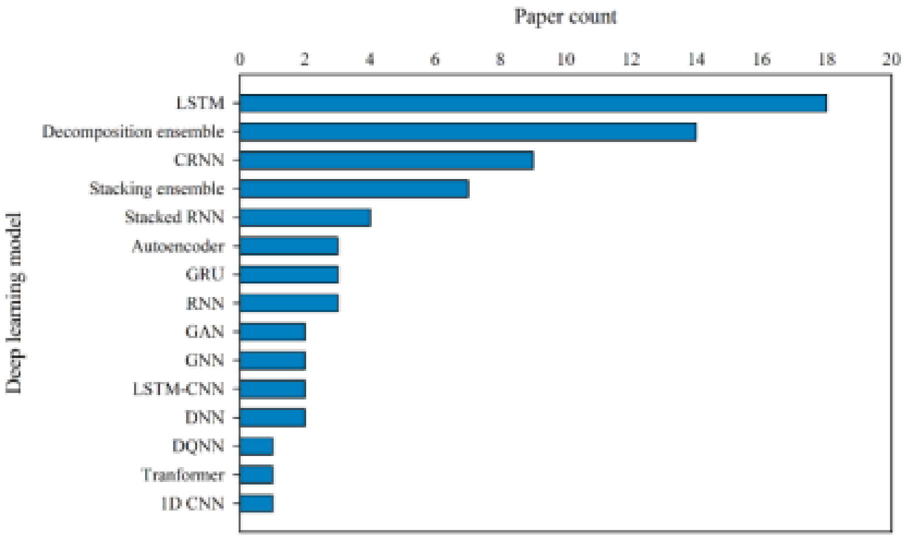


Figure 2.5: Distribution of deep learning models for price forecasting.

DL Model	Advantages	Disadvantages
DNNs	Can learn hierarchical representations, capturing complex abstractions.	Not explicitly designed for sequential data. Limited ability to capture temporal dependencies.
1D CNNs	Effective at capturing local dependencies and patterns in sequential data. Can learn hierarchical representations.	Difficulty in modeling long-term dependencies. Limited applications for price forecasting.
RNNs	Widely applied and effective in capturing temporal dependencies in time series data.	Fixed memory capacity. May struggle with complex patterns or long-term dependencies. Sensitive to noise.
CRNNs	Captures both spatial features and temporal dependencies. Suitable for price forecasting tasks with spatio-temporal dynamics.	Computationally expensive. Challenges related to hyperparameter tuning. Interpretability can be difficult.
Autoencoders	Feature learning, captures relevant patterns and relationships.	Lack of interpretability. Requires sufficient training data. Risk of overfitting.
Transformers	Parallel processing, computational efficiency. Captures complex patterns and dependencies. Incorporation of diverse information.	Computational complexity for extremely long sequences.

Table 2.2: Advantages and disadvantages of different deep learning models for price forecasting (Part 1).

DL Model	Advantages	Disadvantages
GNNs	Captures complex dependencies and relationships between assets. Improves forecasting performance.	Constructing the input graph can be challenging. Difficulty with limited data or sudden market changes. High computational requirements.
GANs	Captures complex patterns and dependencies. Generates realistic price sequences.	Requires large amounts of data. Training instability.
DQNNs	Exploits principles of quantum mechanics to model financial data dynamics.	Requires specialized software and hardware. Difficult training and optimization.
Ensemble Stacking	Combines diverse models. Improves prediction accuracy. Adaptable to market conditions.	Additional computational resources. Increased model complexity. Challenges in interpretability.
Ensemble by Decomposition	Captures different components of the time series. Better understanding of patterns. Interpretable.	Exposure to future features. Risk of data leakage.

Table 2.3: Advantages and disadvantages of different deep learning models for price forecasting (Part 2).

2.3 Smart Indexing

Smart indexing in the context of deep learning in finance refers to the creation and management of financial indices (such as stock indices) using advanced techniques that go beyond traditional methods based on market capitalization or price. Instead of relying on fixed rules, smart indexing uses data-driven approaches, often leveraging deep learning models to optimize the selection and weighting of assets in an index.

Here's how it works:

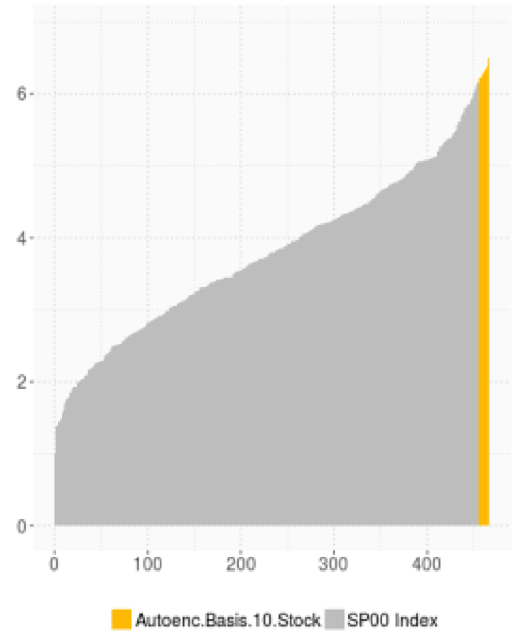
2.3.1 Traditional Indexing vs. Smart Indexing:

Traditional Indexing: Indices such as the S&P 500 or the Dow Jones Industrial Average are created using simple rules, such as market capitalization or price. These indices do not account for complex models or future predictions.

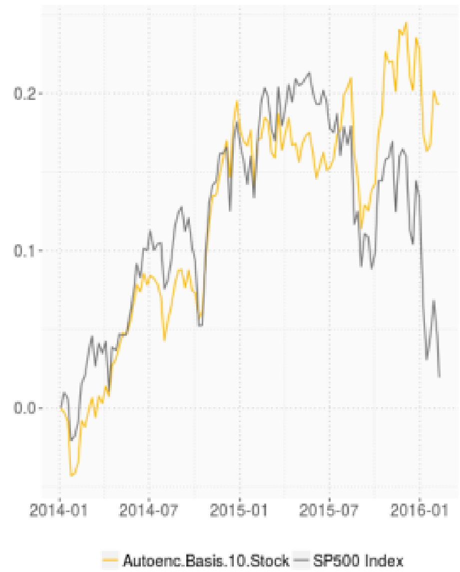
Smart Indexing: This approach incorporates more sophisticated strategies, including factor-based models, machine learning, and deep learning. The goal is to enhance risk-adjusted returns or capture specific market inefficiencies.

2.3.2 Comparing Classic Linear Models and Deep Learning Approaches in Index Replication

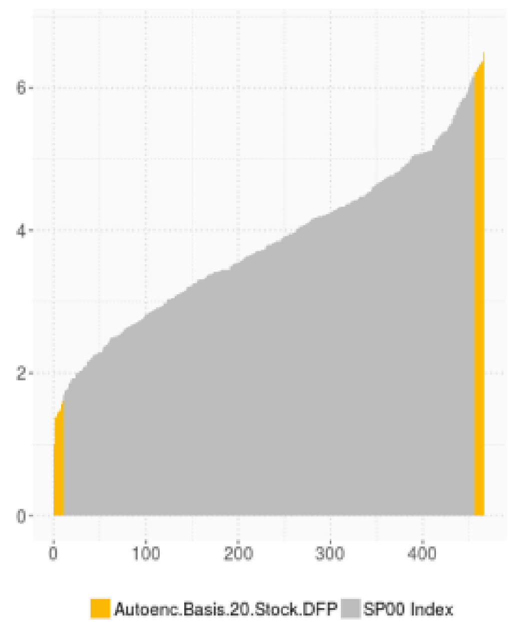
When aiming to replicate (or approximate) a stock index using a subset of stocks, there are two conceptual approaches to choose from:
(i) Identify a small group of stocks that have historically performed very similarly to the observed index.
(ii) Identify a small group of stocks which historically have represented an over-proportionally large part of the total aggregate information of all the stocks the index comprises of.



(a) Information content of individual stocks (by size)



(b) Auto-Encoder smart indexing



(c) Information content of individual stocks (by size)

(d) Auto-Encoder DFP indexing

Figure 2.6: A deep autoencoder compresses the stocks of the SP500 index. Then, we rank all the stocks based on their proximity to the autoencoded information and create an equally weighted portfolio from the top 10 autoencoded base stocks. Subsequently, we use the top ten (most common) and the bottom ten (most individualistic) stocks to create an autoencoder base on which we train a Deep Feature Policy (DFP) for approximating the SP500 index.[3]

we refer to the resulting strategy for approximation (or prediction) as a deep feature policy (DFP) [3]

Top Figure 2.6(Simple Auto-encoder): The stocks of the SP500 are compressed using an auto-encoder. Stocks that are closest to their compressed form are selected as having the highest "communal information." An equally weighted portfolio of the 10 most "communal" stocks approximates the SP500, but this approximation is slightly off in some periods.

Bottom Figure 2.6 (DFP): A deep feature policy (DFP) model is trained on the 10 most communal and 10 least communal stocks. DFP uses hierarchical non-linear transformations of these stocks to create an optimized strategy for replicating the SP500. This method improves accuracy, particularly in specific periods where the simple auto-encoder struggled.

2.3.3 Advantages of Smart Indexing:

Enhanced Performance: By leveraging deep learning, smart indices can potentially offer higher returns or lower risk compared to traditional indices.

Customization: Smart indices can be tailored to specific investment goals, such as focusing on sustainability, low volatility, or growth.

Adaptability: These indices can evolve with market conditions, improving their relevance and effectiveness over time.

2.3.4 Application in Finance

Creation of ETFs: Smart indexing is used in the creation of Exchange-Traded Funds (ETFs) that track the performance of a smart index, offering investors access to advanced strategies in a passive investment vehicle. Portfolio Management: Asset managers can use smart indexing techniques to build portfolios aligned with specific investment strategies, such as factor investing or thematic investing.

2.4 Financial Sentiment Analysis and Behavioral Finance

The integration of deep learning into financial sentiment analysis and behavioral finance has profoundly transformed the way financial market movements are understood and predicted. This approach merges advanced data analysis techniques with psychological principles to offer a more nuanced perspective on market dynamics.

2.4.1 Financial Sentiment Analysis

Financial sentiment analysis leverages natural language processing (NLP) and deep learning techniques to evaluate the tone and opinions expressed in various financial texts. These texts can include corporate reports, news articles, social media posts, and earnings call transcripts. The goal is to extract sentiment indicators that can be used to predict market trends and investor behavior.

2.4.2 Deep Learning Techniques for Sentiment Analysis

1. **Recurrent Neural Networks (RNNs)**: RNNs, particularly Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks, are highly effective for analyzing text sequences. They excel in capturing long-term dependencies within the text, which is crucial for understanding the context and subtleties of expressed sentiment [4].

2. **Convolutional Neural Networks (CNNs)**: Although traditionally used for image processing, CNNs have proven effective for text analysis. They can identify local patterns in text data that indicate sentiment, making them useful for parsing large volumes of financial documents [5].

3. **Transformers**: Transformer-based models, such as BERT (Bidirectional Encoder Representations from Transformers), have demonstrated impressive performance in various NLP tasks, including sentiment analysis. Their attention mechanisms enable a deep understanding of context, allowing for more accurate sentiment extraction [6].

2.4.3 Applications in Finance

- **Market Prediction**: By analyzing the sentiment of financial news, social media, and reports, models can predict market movements more accurately. Positive sentiment might indicate a bullish market, while negative sentiment could signal a bearish trend.

- **Risk Management**: Sentiment analysis can help in identifying potential risks by highlighting negative sentiments that may precede market downturns or company-specific challenges.

- **Investment Strategies**: Investors and fund managers use sentiment analysis to inform their trading strategies. For instance, sentiment indicators might trigger buy or sell decisions based on the prevailing market mood.

2.4.4 Behavioral Finance

Behavioral finance examines the psychological factors that influence investor decisions, often leading to market anomalies that cannot be explained by traditional financial theories. Deep learning models contribute to this field by identifying patterns in behavioral data that correlate with market trends.

2.4.5 Deep Learning Techniques in Behavioral Finance

1. **Reinforcement Learning**: This technique can be used to model the decision-making process of investors, taking into account cognitive biases [8].

2. **Deep Neural Networks (DNNs)**: DNNs can be employed to identify complex patterns in market data that may indicate irrational behavior or market anomalies [9].

3. **Generative Adversarial Networks (GANs)**: GANs can simulate different market scenarios and study how behavioral biases might influence outcomes [10].

2.4.6 Applications in Finance

- **Identifying Behavioral Biases**: Deep learning can help in identifying and quantifying biases such as overconfidence, loss aversion, and herd behavior in financial markets.

- **Predictive Modeling**: By understanding how emotions and psychological biases affect market behavior, deep learning models can predict market movements that deviate from rational expectations.

- **Strategy Development**: Traders can develop strategies that exploit market inefficiencies caused by behavioral biases, improving risk management and decision-making processes.

The application of deep learning in financial sentiment analysis and behavioral finance offers significant advantages in understanding and predicting market dynamics. These technologies provide a deeper insight into the psychological underpinnings of market behavior, leading to more informed and effective financial decision-making.

However, the complex and dynamic nature of financial markets, along with the inherent limitations of data, requires ongoing validation and careful consideration of ethical implications. Despite these challenges, the future of financial sentiment analysis and behavioral finance looks promising, with the potential to fundamentally transform our understanding and interaction with financial markets.

2.5 Financial Text Mining

Financial text mining involves extracting valuable information from large volumes of textual data in the financial domain. This process is critical for analyzing trends, making predictions, and supporting decision-making in finance.

2.5.1 Importance of Financial Text Mining

In the vast landscape of financial information, text mining is essential for processing unstructured data like news articles, social media posts, and financial reports. By transforming this textual data into structured information, financial text mining helps analysts and decision-makers identify trends, sentiments, and key events that impact markets.

2.5.2 Deep Learning Techniques in Financial Text Mining

Word Embeddings Word embeddings, such as Word2Vec, GloVe, and FastText, represent words as dense vectors in a high-dimensional space. These representations capture semantic relationships between words, which is crucial for understanding financial terminology and concepts [12].

Recurrent Neural Networks (RNNs) RNNs, especially Long Short-Term Memory (LSTM) networks, are effective for processing sequential data like text. They can capture long-term dependencies in financial texts, making them suitable for tasks like predicting stock prices based on news articles [13].

Transformer Models Transformer-based models like BERT (Bidirectional Encoder Representations from Transformers) and its finance-specific variants (e.g., FinBERT) have shown state-of-the-art performance in various financial text mining tasks. These models can understand context and nuances in financial texts better than previous approaches [14].

Convolutional Neural Networks (CNNs) While primarily used in image processing, CNNs have also been applied successfully to text classification tasks in finance. They can identify important n-grams and local patterns in text that are indicative of certain financial events or sentiments [15].

2.5.3 Applications of Financial Text Mining

Sentiment Analysis : As discussed in the previous chapter, sentiment analysis is a key application of text mining in finance. Deep learning models can analyze the tone and sentiment of financial news, social media posts, and corporate disclosures to gauge market sentiment and predict price movements [16].

Named Entity Recognition (NER) : NER involves identifying and classifying named entities (e.g., company names, person names, locations) in financial texts. This is crucial for tasks like relationship extraction and event detection in financial contexts [17].

Document Classification : Deep learning models can automatically categorize financial documents (e.g., research reports, regulatory filings) into predefined categories. This aids in organizing and retrieving relevant information efficiently [18].

Event Extraction : Identifying and extracting specific events (e.g., mergers and acquisitions, product launches) from financial news and reports is another important application. This information can be used to predict market reactions and inform investment strategies [19].

Fraud Detection : Text mining techniques can be applied to detect potential fraudulent activities by analyzing patterns and anomalies in financial reports and disclosures [20].

Financial text mining, powered by deep learning, is transforming how financial institutions and investors process and leverage textual information. As techniques continue to evolve, we can expect even more sophisticated applications that provide deeper insights into financial markets and trends. However, it's crucial to approach these technologies with an understanding of their limitations and potential biases, ensuring responsible and effective use in financial decision-making processes.

2.6 Portfolio Management

Portfolio management involves the art and science of selecting and overseeing a group of investments that meet the long-term financial objectives and risk tolerance of an investor or institution. The integration of deep learning techniques into portfolio management has opened new avenues for optimizing investment strategies, risk management, and asset allocation.

2.6.1 Introduction to Portfolio Management

Portfolio management traditionally involves balancing the trade-off between risk and return. The main objectives are to maximize returns for a given level of risk, or equivalently, to minimize risk for a given level of expected return. This involves decisions on asset allocation, security selection, and performance evaluation. Deep learning models, with their ability to process large amounts of data and uncover hidden patterns, are increasingly being used to enhance these processes.

2.6.2 Deep Learning Techniques in Portfolio Management

Reinforcement Learning : Reinforcement learning (RL) is a powerful tool in portfolio management, where an agent learns to make decisions by interacting with an environment to maximize cumulative rewards. In the context of portfolio management, the agent represents an investor, the actions are the selection of assets, and the rewards are the returns on the portfolio. RL algorithms such as Q-learning and Deep Q Networks (DQN) are used to optimize portfolio strategies dynamically [21].

Recurrent Neural Networks (RNNs) : RNNs, particularly Long Short-Term Memory (LSTM) networks, are well-suited for modeling financial time series data, which is crucial in portfolio management. By capturing temporal dependencies in asset prices and economic indicators, RNNs can be used to forecast future returns and inform rebalancing decisions within a portfolio [22].

Autoencoders : Autoencoders are a class of neural networks used for unsupervised learning. In portfolio management, they can be employed to reduce the dimensionality of asset return data, identifying latent factors that drive market movements. This compressed representation can be used to construct portfolios that capture the essential risk factors, thereby improving diversification and risk management [20].

2.6.3 Applications in Portfolio Management

Asset Allocation : Deep learning models are increasingly used to enhance asset allocation strategies by predicting the expected returns and volatilities of different asset classes. For example, a study by Heaton et al. (2017) demonstrated that neural networks could outperform traditional models in predicting asset returns, leading to more efficient asset allocation decisions [24].

Risk Management : Risk management is a critical component of portfolio management. Deep learning models can be used to identify and quantify risks that are not apparent through traditional statistical methods. For instance, Autoencoders can detect anomalies in asset price movements, serving as early warning signals for potential market downturns [25].

Portfolio Optimization : Deep learning techniques, such as Reinforcement Learning, can be used to optimize portfolio strategies by dynamically adjusting the asset weights in response to changing market conditions. This approach can help in achieving higher returns while keeping the risk within acceptable limits. Research by Jiang et al. (2017) showed that a reinforcement learning-based approach to portfolio management could significantly outperform static strategies [26].

Deep learning has the potential to revolutionize portfolio management by offering new ways to predict returns, manage risks, and optimize asset allocations. While the field is still in its early stages, the results so far suggest that deep learning techniques can provide significant advantages over traditional methods. However, it is essential to approach these techniques with a clear understanding of their limitations and to continuously validate and refine models as new data becomes available.

2.7 Risk Assessment

Risk assessment is a critical aspect of financial management, involving the identification, analysis, and prioritization of potential risks that could impact an investment portfolio or a financial institution. The use of deep learning in risk assessment has introduced advanced methodologies that enhance the accuracy and efficiency of identifying and mitigating financial risks.

2.7.1 Introduction to Risk Assessment

Risk assessment in finance traditionally involves evaluating the likelihood of adverse events and their potential impact on investments or financial stability. This process typically includes market risk, credit risk, liquidity risk, and operational risk. The integration of deep learning techniques into risk assessment provides a more data-driven approach, allowing for the analysis of vast datasets to uncover complex patterns and correlations that traditional models might miss [27].

2.7.2 Deep Learning Techniques in Risk Assessment

Autoencoders for Anomaly Detection : Autoencoders are unsupervised neural networks that learn to compress and then reconstruct input data. In the context of risk assessment, autoencoders are used for anomaly detection, where deviations from the normal pattern (reconstruction error) can signal potential risks. This is particularly useful for detecting fraudulent activities or market anomalies that could indicate financial instability [28].

Recurrent Neural Networks (RNNs) : Recurrent Neural Networks (RNNs), and especially Long Short-Term Memory (LSTM) networks, are effective in modeling time series data and can be used to predict future risks based on historical data. RNNs can identify temporal dependencies in data, making them suitable for forecasting financial risks, such as market downturns or credit defaults [29].

Convolutional Neural Networks (CNNs) : While traditionally used in image processing, Convolutional Neural Networks (CNNs) have been adapted for use in risk assessment by applying them to structured financial data. CNNs can capture local patterns in data, such as correlations between different financial indicators, which can help in identifying emerging risks [30].

Generative Adversarial Networks (GANs) : Generative Adversarial Networks (GANs) can be used to simulate different market scenarios, helping in stress testing and scenario analysis. By generating realistic but synthetic financial data, GANs can model extreme events (tail risks) that may not be well represented in historical data, providing a more robust risk assessment framework [31].

2.7.3 Applications in Risk Assessment

Market Risk Management : Deep learning models are increasingly being used to enhance market risk management by predicting extreme market movements and tail risks. Autoencoders, for example, can detect anomalies in market data that may precede significant market corrections, allowing risk managers to take proactive measures [32].

Credit Risk Analysis : In credit risk analysis, deep learning models such as LSTMs are used to forecast the likelihood of default by analyzing a borrower's historical payment patterns. This allows financial institutions to better assess the creditworthiness of borrowers and manage their loan portfolios more effectively [33].

Fraud Detection : Fraud detection is a critical application of deep learning in risk assessment. Techniques such as autoencoders and RNNs can analyze transaction data to identify unusual patterns that may indicate fraudulent activities. These models can be trained on historical fraud data to improve their detection accuracy over time [34].

2.7.4 Challenges and Future Directions

Despite the advancements that deep learning brings to risk assessment, there are challenges that need to be addressed. These include the interpretability of deep learning models, the need for large and high-quality datasets, and the risk of overfitting. Future research is likely to focus on improving model transparency and developing hybrid models that combine the strengths of deep learning with traditional risk management techniques.

The integration of deep learning into risk assessment offers significant improvements in the ability to predict and mitigate financial risks. By leveraging complex models capable of analyzing vast amounts of data, financial institutions can gain a more comprehensive understanding of potential risks and develop more effective risk management strategies. However, it is essential to continuously refine these models and validate their predictions to ensure they remain robust and reliable in dynamic financial environments.

General conclusion

The application of deep learning in finance has shown significant potential in enhancing financial analysis, forecasting, and decision-making processes. This report thoroughly examined the use of advanced machine learning models across various financial domains, including price forecasting, option pricing, financial sentiment analysis, and portfolio management. The results demonstrate that deep learning techniques, particularly those leveraging neural networks and natural language processing, offer a substantial improvement in capturing complex patterns, modeling non-linear relationships, and predicting market trends with higher accuracy than traditional methods.

Deep learning models, such as LSTM networks, transformers, and autoencoders, have proven effective in analyzing large datasets and identifying subtle correlations that are often overlooked by conventional approaches. These capabilities have been particularly beneficial in areas like risk assessment and anomaly detection, where understanding intricate data relationships is critical.

To further enhance the robustness of these models, future efforts should focus on optimizing model architectures and integrating a broader range of data sources, including real-time market data and alternative financial indicators. Additionally, ongoing attention to ethical considerations, such as model transparency and the mitigation of biases, will be essential in ensuring responsible deployment in real-world financial applications.

In conclusion, the findings from this study underscore the transformative impact of deep learning in the financial sector, offering powerful tools for improving prediction accuracy, managing risks, and ultimately driving better financial outcomes.

Bibliography

[1] Culkin, R., Das, S. R. (2017). Machine Learning in Finance: The Case of Deep Learning for Option Pricing. arXiv preprint arXiv:1908.10063.

[2] Zhang, C., Sjarif, N. N. A., Ibrahim, R. (2022). Deep learning models for price forecasting of financial time series: A review of recent advancements: 2020-2022.

[3] Heaton, J. B., Polson, N. G., Witte, J. H. (2016). Deep Learning in Finance. arXiv preprint arXiv:1602.06561.

[4] Jiang, W. (2021). Applications of deep learning in stock market prediction: recent progress. *Expert Systems with Applications*, 184, 115537.

[5] Araci, D. (2019). FinBERT: Financial Sentiment Analysis with Pre-trained Language Models. *arXiv preprint arXiv:1908.10063*.

[6] Yang, L., Xu, Z., & Shi, J. (2021). Transformer-based models for financial sentiment analysis: A comparative study. *Applied Soft Computing*, 106, 107356.

[7] Xing, F. Z., Cambria, E., & Welsch, R. E. (2018). Natural language based financial forecasting: a survey. *Artificial Intelligence Review*, 50(1), 49-73.

[8] Fischer, T. G. (2018). Reinforcement learning in financial markets-a survey. *FAU Discussion Papers in Economics*, (12/2018).

[9] Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied Soft Computing*, 90, 106181.

[10] Koshiyama, A., Firoozye, N., & Treleaven, P. (2021). Generative adversarial networks for financial trading strategies fine-tuning and combination. *Quantitative Finance*, 21(5), 797-813.

[11] Johnson, J., Tellis, G. J., & MacInnis, D. J. (2005). Losers, winners, and biased trades. *Journal of Consumer Research*, 32(2), 324-329.

[12] Ding, X., Zhang, Y., Liu, T., & Duan, J. (2015). Deep learning for event-driven stock prediction. In *Proceedings of the 24th International Conference on Artificial Intelligence* (pp. 2327-2333).

[13] Kraus, M., & Feuerriegel, S. (2017). Decision support from financial disclosures with deep neural networks and transfer learning. *Decision Support Systems*, 104, 38-48.

[14] Yang, L., Xu, Z., & Shi, J. (2021). Transformer-based models for financial sentiment analysis: A comparative study. *Applied Soft Computing*, 106, 107356.

[15] Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[16] Araci, D. (2019). FinBERT: Financial Sentiment Analysis with Pre-trained Language Models. arXiv preprint arXiv:1908.10063.

[17] Luo, X., & Liao, C. (2021). Named entity recognition for Chinese financial texts based on a finance-specific domain ontology. *IEEE Access*, 9, 133368-133379.

[18] Mai, F., Tian, S., Lee, C., & Ma, L. (2019). Deep learning models for bankruptcy prediction using textual disclosures. *European Journal of Operational Research*, 274(2), 743-758.

[19] Jacobs, G., Lefever, E., & Hoste, V. (2018). Economic event detection in company-specific news text. In *Proceedings of the First Workshop on Economics and Natural Language Processing* (pp. 1-10).

[20] Hajek, P., & Henriques, R. (2017). Mining corporate annual reports for intelligent detection of financial statement fraud–A comparative study of machine learning methods. *Knowledge-Based Systems*, 128, 139-152.

[21] Jiang, Z., Xu, D., & Liang, J. (2017). A deep reinforcement learning framework for the financial portfolio management problem. *arXiv preprint arXiv:1706.10059.*

[22] Nelson, D. M., Pereira, A. C., & de Oliveira, R. A. (2017). Stock market's price movement prediction with LSTM neural networks. In *2017 International Joint Conference on Neural Networks (IJCNN)* (pp. 1419-1426). IEEE.

[23] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

[24] Heaton, J. B., Polson, N. G., & Witte, J. H. (2017). Deep learning in finance. *Annual Review of Financial Economics*, 11(1), 9-21.

[25] Hinton, G. E., Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 504-507.

[26] Moody, J., Saffell, M. (2001). Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks*, 12(4), 875-889.

[27] Alexander, C. (2008). *Market Risk Analysis, Quantitative Methods in Finance*. John Wiley & Sons.

[28] Chalapathy, R., & Chawla, S. (2019). Deep Learning for Anomaly Detection: A Survey. *arXiv preprint arXiv:1901.03407*.

[29] Malhotra, P., Vig, L., Shroff, G., & Agarwal, P. (2015). Long Short Term Memory Networks for Anomaly Detection in Time Series. In *23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*.

[30] Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.

[31] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative Adversarial Nets. In *Advances in Neural Information Processing Systems* (pp. 2672-2680).

[32] Kuo, R., & Lee, L. (2016). Predicting stock market trends using data mining and deep learning techniques. *Expert Systems with Applications*, 40(9), 6135-6146.

[33] Ding, X., Liu, Y., Liao, L., & Wu, H. (2016). Using Structured Events to Predict Stock Price Movement: An Empirical Investigation. *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM)*.

[34] Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications*, 39(3), 3446-3453.