Gabelli School of Business

Fordham University

Financial Machine Learning Lab

Karandeep Sonewane
MSQF 25, Fordham University

Professor N. K. Chidambaran
Professor and Associate Dean of Research and Faculty Development
Finance and Business Economics, Fordham University

The complete code and additional resources for this project can be accessed at the following GitHub repository:
https://github.com/karandeeps18/BlackScholesFunctionApproximationUsingDNN

Financial Machine Learning Lab, Fordham University

Comparative Study of Neural Network Architectures for Black-Scholes Function Approximation

In this study, we approach the problem of option pricing as a **function approximation task** using deep neural networks. The traditional method for pricing European call options relies on the well-known **Black-Scholes formula**, which provides an analytical solution under specific market assumptions, such as **constant volatility and log-normal asset price distribution**. While the Black-Scholes model has been widely used, it may face limitations in real-world scenarios where market conditions deviate from these idealized assumptions.

We propose leveraging deep learning models — specifically, **Feedforward Neural Networks (FFN)**, **Residual Neural Networks (ResNet)**, and **Radial Basis Function Networks (RBFN)** — to approximate the complex non-linear relationship between the input features and the option price. The neural networks are trained to learn this mapping:

$$f(S, X, T, r, \sigma) \approx Option\ Price$$

where:

- S is the Stock Price,
- X is the Strike Price,
- T is the Time to Maturity,
- r is the Risk-Free Rate,
- σ is the Volatility.

Function Approximation Approach:

The deep learning models act as **universal function approximators**, capturing the underlying relationships in the data without relying on the restrictive assumptions of the Black-Scholes formula. This allows the models to generalize well even in non-ideal market conditions, such as when volatility is non-constant or when the asset price distribution deviates from the log-normal assumption.

1. Short Overview

The objective of this project was to develop and compare various deep neural network architectures for approximating the Black-Scholes pricing formula for European call options. The models explored include:

1. **Feedforward Neural Network (FFN)**
2. **Residual Neural Network (ResNet)**
3. **Radial Basis Function Network (RBFN)**

Dataset:

We have created synthetic data generated using the Black-Scholes formula with controlled parameters. The **strike price X and stock price S** is sampled from a **uniform distribution** between $50 and $300. **Time-to-maturity** includes 1 week, 1, 3, 6, 9 months, and 1, 2, 3 years. **risk-free rate r is 4%, 4.5%, or 5%. Volatility σ** can be **25%, 50%, or 75**%, representing low, medium, and high volatility scenarios.

- Features: Stock Price, Strike Price, Time to Maturity, Risk-Free Rate, and Volatility.
- Target: Option Price.
- Number of samples: 14,400 (after increasing num_samples during data generation).

2. Model Specifications and Hyperparameters

A. Feedforward Neural Network (FFN):

- Architecture: Fully connected layers with ReLU activation
- Layers: 3 hidden layers with 64, 128, and 64 units respectively
- Regularization: L2 regularization (l2=0.001)
- Optimizer: Adam with a learning rate of 0.001
- Loss Function: Mean Squared Error

B. Residual Neural Network (ResNet):

- Architecture: Residual blocks with skip connections
- Layers: 3 residual blocks, each with 2 layers (64 units per layer)
- Activation: ReLU activation
- Regularization: L2 regularization (l2=0.001) and Dropout (rate=0.2)
- Optimizer: Adam with a learning rate of 0.001
- Loss Function: Mean Squared Error

C. Radial Basis Function Network (RBFN):

- Architecture: RBF layer followed by a fully connected dense layer
- RBF Units: 40 radial basis function units
- Kernel: Gaussian RBF kernel
- Regularization: L2 regularization (l2=0.001)
- Optimizer: Adam with a learning rate of 0.0005
- Loss Function: Mean Squared Error

Hyperparameter Tuning:

- Method: Keras Tuner (RandomSearch).
- Parameters Tuned:
  - Number of units per layer (32, 64, 128)
  - Learning rate (0.001, 0.0005)
  - Regularization (l2=0.001)
  - Dropout rate (0.0, 0.2)
  - Number of RBF units (10, 20, 30, 40, 50)

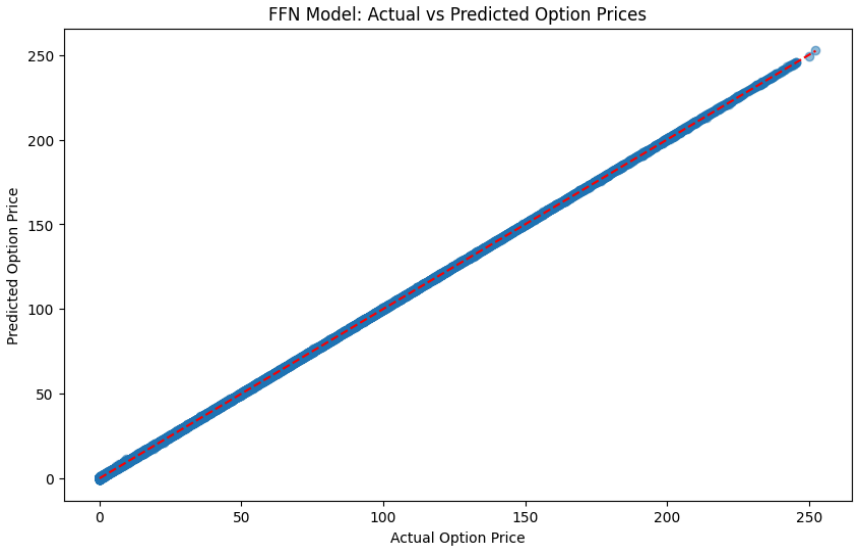
3. Results and Performance Metrics

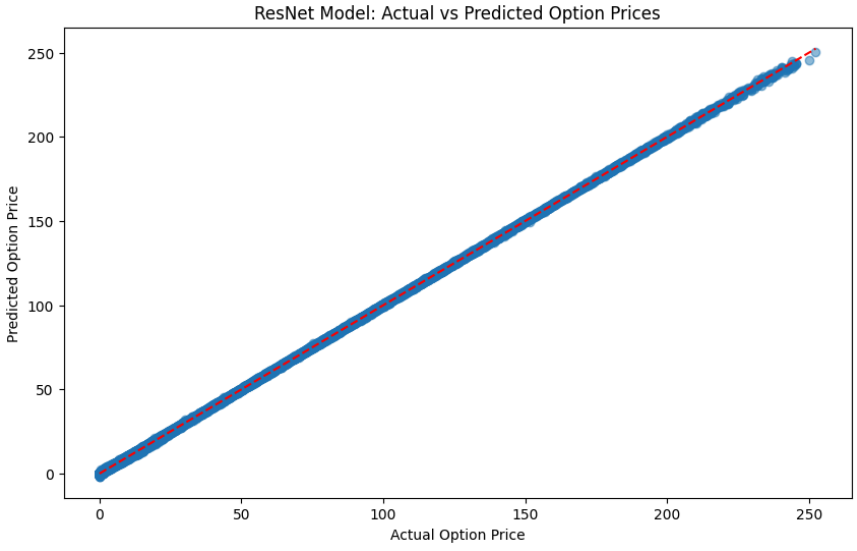
Model	Test MSE	Test MAE	Observations
FFN	**1.2460**	**0.8361**	Best performance; low residuals and high accuracy.
ResNet	4.8986	1.6019	Decent performance, but slightly worse than FFN; higher residuals.
RBFN	22.3970	3.6053	Worst performance; struggled with non-linearities in the data.

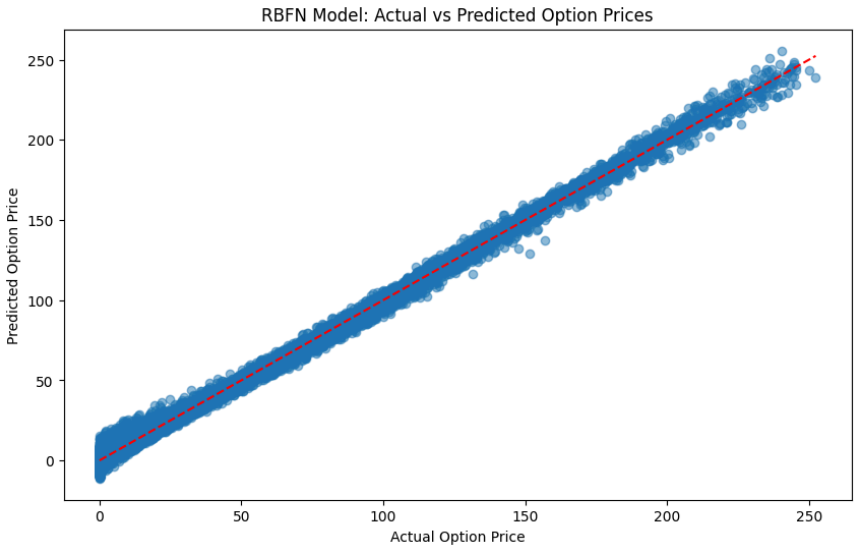
The **FFN model outperformed both ResNet and RBFN, achieving the lowest MSE and MAE**. It effectively captured the non-linear patterns in the data. The ResNet model showed reasonable performance but had slightly higher errors. It may benefit from additional tuning or deeper residual blocks. Whereas RBFN model performed poorly, likely due to the limited flexibility of radial basis functions in approximating complex non-linearities.

Result Summary:

1. Actual vs Predicted Prices:

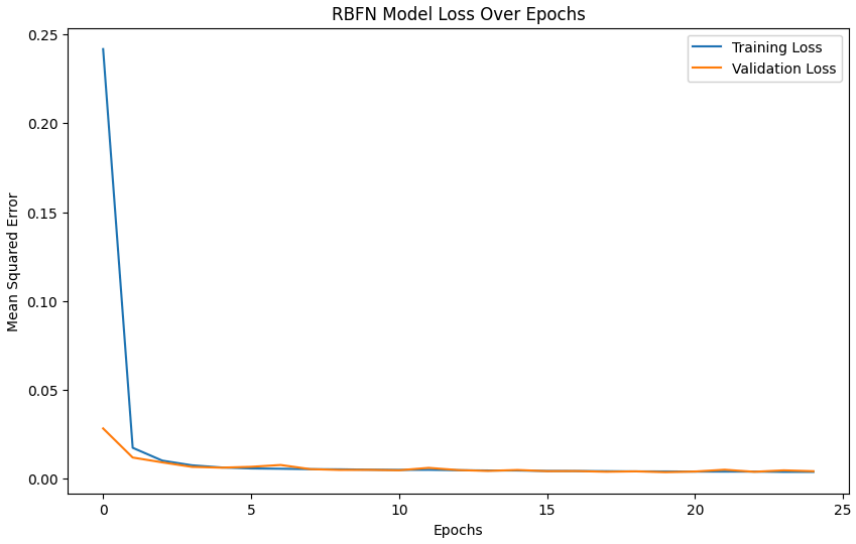


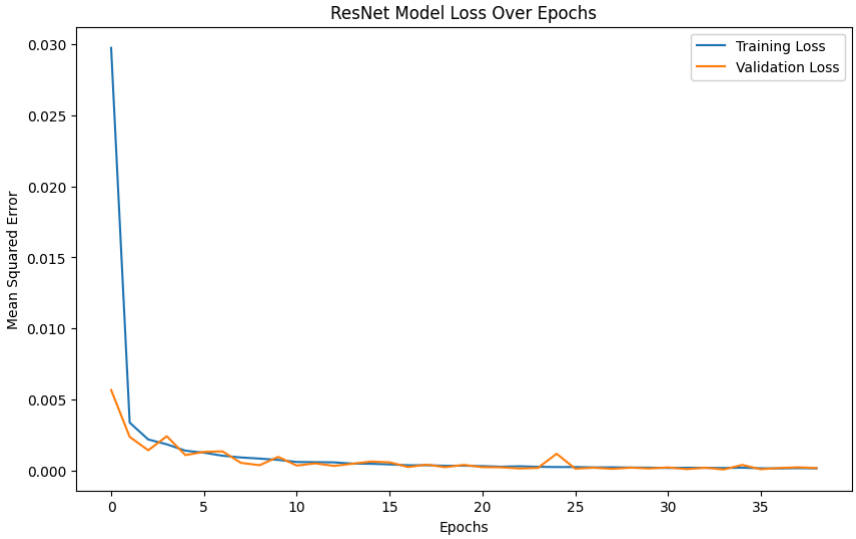


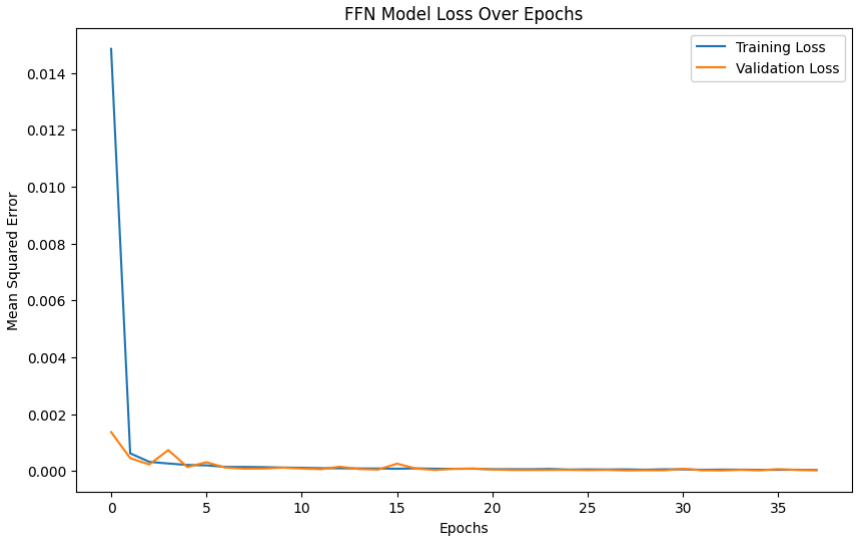


- FFN predictions were closest to the perfect fit line, indicating the best fit.
- ResNet predictions showed slightly more deviation but were still reasonable.
- RBFN predictions had the most scatter, indicating poor approximation.

Residual/Errors :

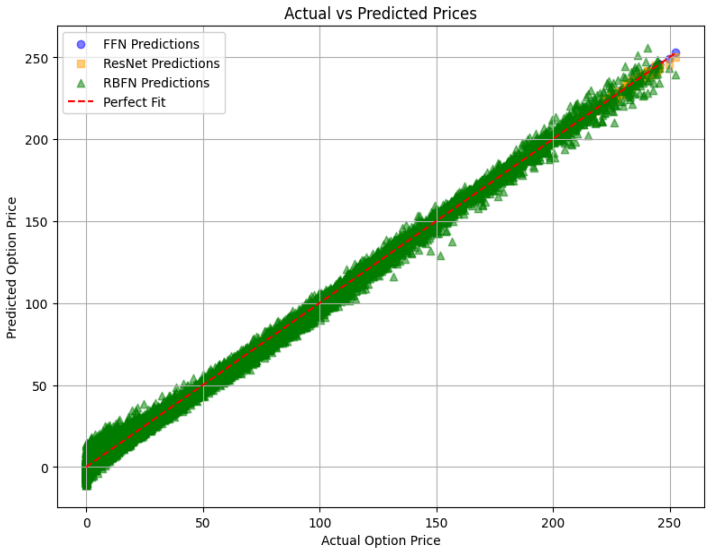


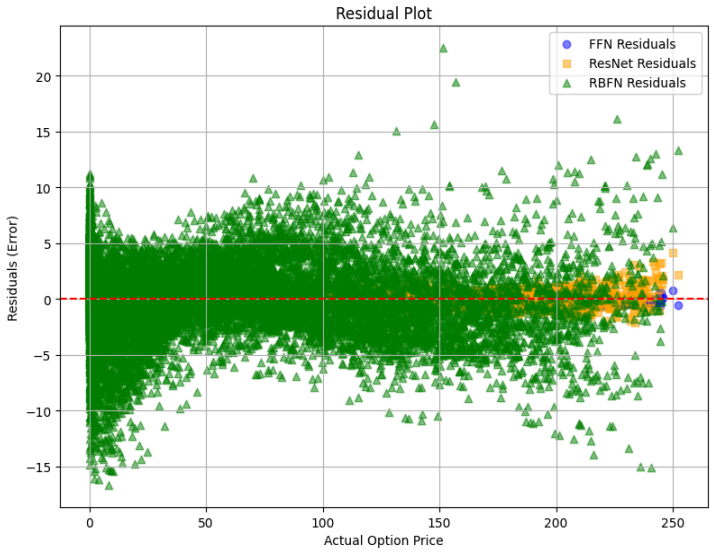


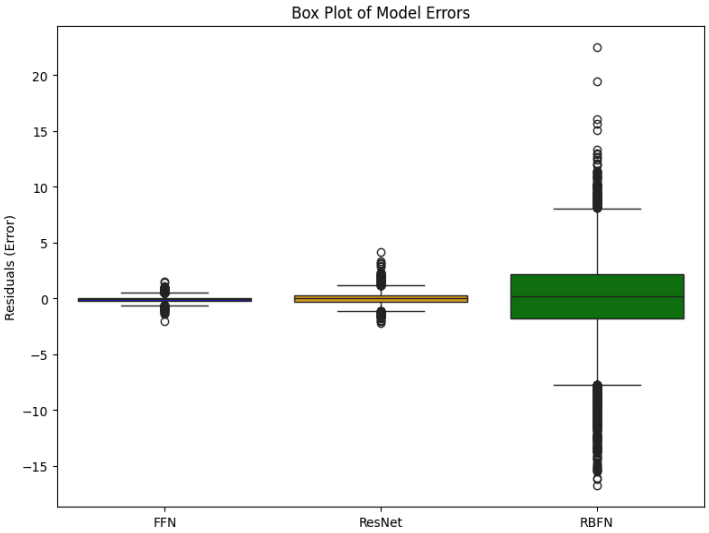


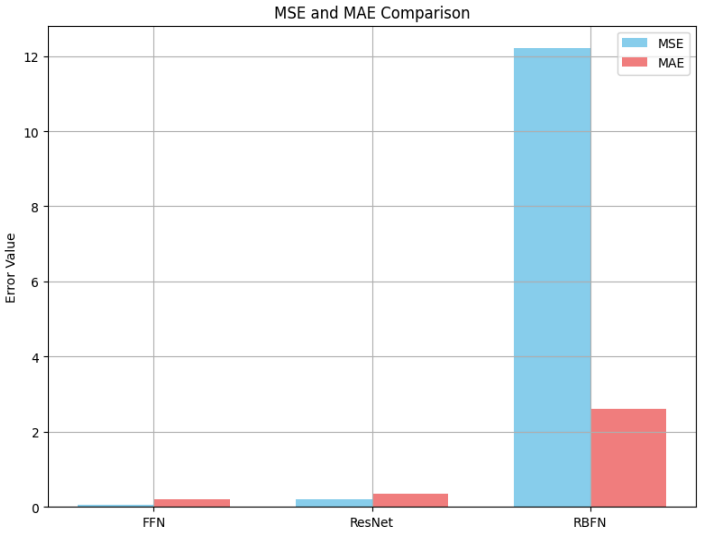
FFN had the smallest residuals, centered around zero. ResNet residuals were more spread out but still showed reasonable performance. However, RBFN had large residuals, especially at extreme values, indicating underfitting.

Model Comparison:









- FFN had the smallest spread of residuals, indicating consistent performance.
- ResNet showed slightly larger error variance.
- RBFN had the widest spread and many outliers, suggesting poor model fit.

MSE and MAE Comparison Bar Plot:

- FFN had the lowest MSE and MAE values.
- ResNet had slightly higher error values.
- RBFN had significantly higher error values.

4. Conclusion

In conclusion, the **Feedforward Neural Network (FFN)** demonstrated the most robust performance in approximating the Black-Scholes formula, achieving the lowest Mean Squared Error (MSE) and Mean Absolute Error (MAE). Its strong predictive accuracy and generalization capabilities suggest that a simpler architecture can effectively capture the underlying relationships in the synthetic data.

The **Residual Neural Network (ResNet)**, while exhibiting reasonable performance, did not surpass the FFN. Its slightly higher error metrics indicate potential room for improvement, such as deeper residual blocks or enhanced regularization techniques to better capture intricate data patterns.

On the other hand, the **Radial Basis Function Network (RBFN)** faced challenges in modeling the complex non-linearities of the option pricing data, leading to higher error rates. This suggests that the RBFN's current configuration may not be optimal for this problem, and future work could explore alternative kernel functions or an increased number of RBF units to enhance its approximation power. Overall, the results underscore the effectiveness of neural networks as function approximators and highlight the importance of model selection and tuning in financial applications.

Recommendations:

1. We can further o**ptimize the ResNet model by experimenting with deeper architectures or different activation functions.**
2. We can explore ensemble methods, combining predictions from both FFN and ResNet, to potentially improve performance.
3. Additionally we can consider alternative non-linear approximation methods or architectures for future work, such as Long Short-Term Memory (LSTM) networks or Transformer-based models for sequential data.

References

1. **Hornik, K., Stinchcombe, M., & White, H. (1989)**. *Multilayer feedforward networks are universal approximators*.
2. **Hutchinson, J. M., Lo, A. W., & Poggio, T. (1994)**. *A nonparametric approach to pricing and hedging derivative securities via learning networks*.
3. **Broomhead, D. S., & Lowe, D. (1988)**. *Multivariable functional interpolation and adaptive networks*.
4. **Park, J., & Sandberg, I. W. (1991)**. *Universal approximation using radial-basis-function networks*.
5. **He, K., Zhang, X., Ren, S., & Sun, J. (2016)**. *Deep residual learning for image recognition*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
6. **E, W., & Yu, B. (2018)**. *The deep Ritz method: A deep learning-based numerical algorithm for solving variational problems*. Communications in Mathematics and Statistics