CS230

Option Pricing with Deep Learning

Alexander Ke
Department of Computer Science
Stanford University
alexke@stanford.edu

Andrew Yang
Department of Computer Science
Stanford University
ycm@stanford.edu

Abstract

We study the performance of deep learning models on pricing options using inputs to the popular Black-Scholes model. By viewing option prices as a function of contract terms and financial states, we can use a neural network to avoid assumptions about financial mechanics and learn from historical data. MLP1 and MLP2 models take 20-day historical volatility as an input, while the LSTM model takes in the latest 20 days of underlying price, allowing it to learn volatility. Because of our unprecedented data availability, we did not exclude any possibly illiquid options from our dataset, allowing our model to generalize to them. All models performed far superior to the Black-Scholes model, while we found multi-task learning for bid/ask instead of equilibrium price in MLP2 to be most successful. This hints that future efforts using historical data should consider predicting bid/ask prices.

1 Introduction

In 2018, the Chicago Board Options Exchange reported that over \$1 quadrillion worth of options were traded in the US [6]. Option contracts are a financial derivative that represents the right, but not the obligation, to buy (call) or sell (put) a particular security on (European type) or before (American type) an expiration date. A subset of these instruments were first priced with the Black-Scholes formula where the premium of a European call option without dividends is given by

$$C = S \cdot \Phi(d_1) - Xe^{-rT} \cdot \Phi(d_2)$$

with $d_1 = \frac{\ln(S/X)+(r+\sigma^2/2)T}{\sigma\sqrt{T}}$, $d_2 = d_1 - \sigma\sqrt{T}$, and $\Phi(\cdot)$ as the cumulative density function of the standard normal distribution [5]. The inputs are S the underlying price, X the exercise price, T the annualized fraction of time until expiration, r the risk free interest rate, and σ the standard deviation of stock price returns, which cannot be directly observed. This model was derived by assuming stock prices are continuous and follow geometric Brownian motion, r and σ are constant over the option life, and that the market is frictionless (no transaction costs, short squeeze, taxes, etc.).

However, even controlled for assumptions, Black-Scholes mismatches empirical findings and fails to explain the volatility surface. This precipitated more mathematically complex approaches such as the Heston model or jump-diffusion processes, as well as Monte Carlo methods to leverage advances in computing [12].

Alternatively, we can view an option as a function of the contract terms X and T, as well as information on the prevailing financial state S, r, and σ. This provides a foundation on which to build a computational model that can evade assumptions about financial mechanics and learn from historical options data. We hope to build a neural network model and compare its performance to the seminal Black-Scholes model. At the most basic level, our inputs are the terms of the option contract and the price of the underlying at transaction time and our output would be the price of the option. We will explore models that take advantage of previous state information such as recurrent neural networks (RNNs) or volatility estimates and risk free rate.

2 Related work

Malliaris and Salchenberger [11] first studied a neural network approach to estimate the close price of S&P 100 options using transaction data from the first six months of 1990. They supplemented the contract data with the option premium and underlying price for the day prior, as well as historical realized volatility. They further divided their training set to separately forecast the price of in the money and out of the money options, which may lead to overfitting and disagrees with the practically continuous property of stock prices. They constructed a neural network with a hidden layer of four nodes and one output node, and using mean squared error, this network outperformed the Black-Scholes model in about half the test periods [11]. Other early works continued to use a single hidden layer with up to 11 neurons [3, 1, 4].

Mezofi and Szabo [12] characterize the final outputs of these neural network approaches in three categories: directly predicting the option premium, while possibly adding in implied volatility or other engineered features; predicting implied volatility and using it as an input to Black-Scholes to return the option premium [2]; and finding the ratio between option premium and strike price. Garcia et al introduced the “homogeneity hint" to constrain the set of possible outputs such that the option pricing function is homogeneous in asset price and strike price with degree 1 [9].

Considering option pricing and volatility estimation as a supervised learning problem, the Multi-Layer Perceptron (MLP) has been the workhorse neural network [15]. Recent work has been focused around increasing the complexity of MLPs, up to 4 hidden layers with 400 each [10]. Ensemble and modular neural networks have also combined the outputs of independent MLPs to improve generalization [8, 11]. Although Recurrent Neural Networks (RNNs) have been extensively applied to the stock price prediction problem, little work exists for volatility estimation and no work exists for options pricing [14, 17].

3 Dataset and Features

Because of the huge diversity of historical options data and their relatively few applications, no open source dataset exists. We obtained our option prices [18] and security prices [19] from Wharton Research Data Services, which maintains access to a database with the daily trading outcomes of all listed option contracts and their corresponding security prices. We supplemented this data with treasury yields obtained from the US Treasury Resource Center [16], which will inform the risk-free rate r for our model.

This dataset provides us the information on the contract terms X and T, as well as observable financial state S. To find the appropriate risk-free rate r, we matched the yield on the US Treasury instrument having maturity closest to the time until expiration of each option, a widely accepted options trading practice. However, to find the volatility σ for Black-Scholes and our models, we assume that the historical volatility from the previous 20 trading days (approximately one trading month) is representative of the volatility over the life of the option. We can then feed this new feature into the Black-Scholes model, and as an additional input into our MLP models. Also, the trading data only provides the inner bid and ask prices of the contract, so we use the equilibrium price (average of bid and ask) as our label for the fair value of the option.

Earlier inquiries by Anders et al. [3] and Stark [13] suggest using exclusion criteria to remove “non-representative" examples that represent illiquid or extraordinary options circumstances. Such criteria filter out options that are too deep in the money or out of the money, options with over 2 years to expiration, or options that are traded at such low prices that the discrete nature of security prices becomes a consideration. However, these studies had up to 100,000 observations, yet we will train on over ten times that amount of data. With this additional data, we hope that a neural network is able to generalize to these rarer circumstances and explain behavior outside of Black-Scholes such as the volatility smile.

Our total dataset size is 12,268,772 examples of roughly half calls and half puts. 98% of our data is used as a training set with 1% used as a validation set during training and 1% as our test set.

4 Methods

We explore three network architectures for this option pricing problem that differ as follows: MLP1 using the 20-day historical volatility as an input to find the equilibrium price of an option, MLP2 which also uses 20-day historical historical volatility but uses multi-task learning to calculate the bid and ask prices as outputs, and a long short-term memory (LSTM) network that attempts to approximate volatility from successive states instead of as a precomputed feature. Each model is trained and evaluated on call and put options separately. Since option pricing is a regression task, our objective is to minimize the mean squared error of our predictions. Our source is available at `https://github.com/ycm/cs230-proj` and uses the Keras library [7].

4.1 MLP1

We extended the hyperparameter search in Liu et al. [10], who trained a network on the similar task of computing implied volatilities from Black-Scholes generated data. Our network uses contains four hidden layers: three layers at 400 neurons each and output layer with one neuron. Liu et al. [10] did not try the Leaky ReLU activation, yet we found that using this allows the network to learn slightly faster, so our 400-neuron layers use Leaky ReLU. Our output node uses a ReLU activation, appropriate since option prices are non-negative. As a result of these activation choices, weights were initialized with Glorot initialization. Contrary to Liu et al. [10], we found that batch normalization significantly improves the training speed and loss at convergence, so we apply batch normalization after the 400-neuron layers. We did not apply any regularization techniques because the option price is sensitive to all the inputs provided, so dropout does not improve accuracy for price prediction [10]. This is also supported by our train and test MSEs in Table 1, which suggest low variance.

4.2 MLP2

MLP2 uses the same architecture as MLP1, but approaches the problem of predicting the bid and ask prices instead of the equilibrium price of an option. Consequently, the output layer of this network has two neurons with ReLU activation: one neuron to output the bid price and another neuron to output the ask price. By segregating the estimation of the fair price of an option from historical bid and ask prices, we hope this model reveals whether the bottleneck on accuracy is caused by the information limits of our inputs or by extrapolating the fair price from historical bid and ask prices.

4.3 LSTM

Our LSTM network is the pioneering inquiry into approaching the option pricing problem using RNN architectures. Because RNNs capture state information, we hope that this architecture can learn to estimate volatility from recent observations to improve option pricing performance. An 8-unit LSTM takes in the closing price at each timestep over 20 timesteps, chosen because we calculate historical volatility over 20 days. The output sequence is fed forward for three layers of 8-unit LSTMs. The final timestep's prediction is then concatenated with the S, X, T, and r, and fed through the same MLP1 architecture to output the equilibrium price. The architecture is shown in Appendix Figure 3.

5 Results and Discussion

Our error analysis for all models is reported in Table 1, where train-MSE reports the mean squared error on the training set, while all other metrics are calculated over the test set. Bias is the median percent error, AAPE is the average absolute percent error, MAPE is the median absolute percent error, and PEX% is the percentage of observations within $\pm X\%$ of the actual price.

5.1 MLP1

Since we train on approximately 5x more data than Liu et al. [10], we use a batch size 4x bigger at 4096 examples. Because we use batch normalization, we chose our optimizer to be Adam. We

	Model	train-MSE	MSE	Bias	AAPE	MAPE	PE5	PE10	PE20
Call	BS	322.95	321.37	-0.05	78.79	4.81	50.52	59.33	67.43
	MLP1	23.71	24.00	0.01	24.49	2.12	61.04	68.39	74.33
	MLP2	7.70	15.21	0.09	23.45	1.73	63.03	70.10	75.54
	LSTM	30.61	30.97	0.13	26.58	2.33	58.94	66.35	72.42
Put	BS	543.48	533.25	97.37	68.00	97.46	12.87	18.22	23.58
	MLP1	15.65	15.66	5.03	43.73	18.48	30.46	40.51	51.13
	MLP2	2.03	8.84	3.85	39.59	14.32	33.74	44.25	55.01
	LSTM	22.81	23.15	6.01	48.32	26.05	27.45	36.24	46.17

Table 1: Error metrics comparing MLP1 price and MLP2 equilibrium price with Black-Scholes prices. Note all metrics beside MSE are percentages.

evaluate our model against Black-Scholes using relative metrics which are more appropriate for the wide range of option values.

To help the model find the best optima, we manually adjusted the learning rate with the following schedule: 10 epochs at $\eta = 10^{-3}$, 10 epochs at $\eta = 10^{-4}$, and 10 epochs at $\eta = 10^{-5}$. Each step in learning rate further reduced the loss and both the call option and put option models converged by the 30th epoch. The losses over training are shown in Appendix Figure 4.

Because our testing MSEs for MLP1 are approximately equal to the training MSEs, there is no evidence of overfitting. Our large training set therefore allowed the model to generalize such that there is no need for dropout, in line with Liu et al. [10]. Our MLP1 model significantly improves over the Black-Scholes model in all error metrics, and the PEX% metrics additionally reveal that MLP1 prices illiquid options much more accurately than Black-Scholes, which struggled as seen in PE20. Black-Scholes particularly struggled in pricing put options, where MLP1 offers improvements in all metrics.

5.2 MLP2

We trained MLP2 models using the same 4096 batch size and Adam optimizer. Our learning rate schedule included 30 epochs at $\eta = 10^{-3}$, 10 epochs at $\eta = 10^{-4}$, and 10 epochs at $\eta = 10^{-5}$, and 10 epochs at $\eta = 10^{-6}$ when both call and put option models have converged. The losses over training are shown in Appendix Figure 5.

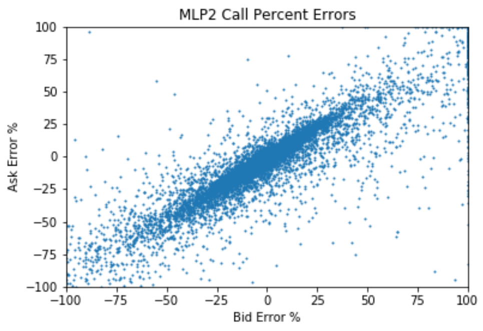
In our Table 1 error analysis, we averaged MLP2's bid and ask prices on the test set to obtain an equilibrium price that we compared with the corresponding historical equilibrium prices. All error metrics are slightly superior to their corresponding metrics yielded from MLP1, which shows that the conversion of bid/ask to equilibrium prices somewhat limits the accuracy of models that seek to regress equilibrium price. This multitask learning approach was successful because regressing bid and ask prices are similar tasks that share features.

When we consider the errors in bid and ask separately, we find that the distribution of relative errors are heavily clustered around the center $(0, 0)$ and very positively correlated. This behavior shows our model quotes prices that are tighter or wider on both bid and ask than the historical bid and ask. For call options, our model is extremely accurate, where 32.1% test examples are within $\pm 1\%$ of the bid/ask, while 60.5% are within $\pm 5\%$. Our accuracy is worse for put options with 8.8% within $\pm 1\%$ and 29.8% within $\pm 5\%$. This pattern is shown by the wider kernel-density estimate in Figure 2.

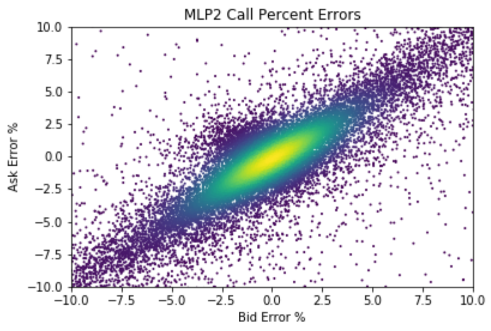
5.3 LSTM

We trained the LSTM using the same batch size of 4096 and Adam optimizer. This model consisted of three LSTM layers that fed one output neuron to the MLP1 architecture after being merged with the contract terms S, X, T, and r. Our learning rate schedule was 10 epochs at $\eta = 10^{-2}$, 5 epochs at $\eta = 10^{-3}$, and 5 epochs at $\eta = 10^{-4}$, after which the model converges to an optima with higher loss than our MLP models (Appendix Figure 6). This is the result of a hyperparameter search that covered learning rates from 10^{-1} to 10^{-7}, and LSTM layers from 1 to 8.

Our LSTM approach is motivated by the fact that historical volatility was explicitly computed for Black-Scholes and our MLP models—we wanted to investigate whether added LSTM layers can

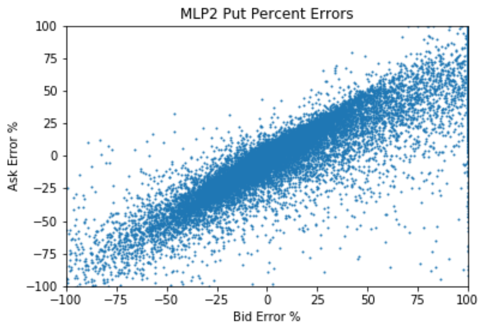


(a) Scatterplot of relative errors. Notice the frame $[-100, 100]$ displays 99.1% of all errors.

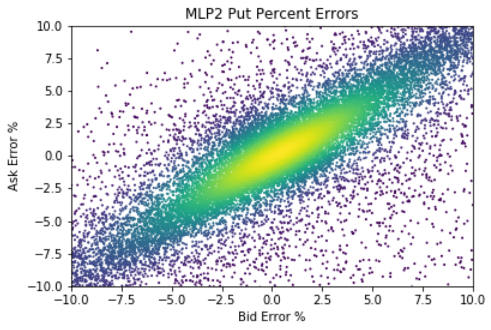


(b) Scatterplot colored with kernel-density estimate using Gaussian kernels. Notice the frame $[-10, 10]$.

Figure 1: Error plots of MLP2 model trained on call data.



(a) Scatterplot of relative errors. Notice the frame $[-100, 100]$ displays 99.1% of all errors



(b) Scatterplot colored with kernel-density estimate using Gaussian kernels. Notice the frame $[-10, 10]$.

Figure 2: Error plots of MLP2 model trained on put data.

learn a feature either as informative as or more informative than historical volatility. However, the fact that we underperformed the MLP models suggests that there is more work to be done. Because of computational concerns, we only fed twenty timesteps, which may not have been enough for the LSTM layers to learn this feature. Additionally, we predict equilibrium price instead of bid/ask directly, while the MLP2 performance hints we may get better performance if we output bid/ask.

6 Conclusion

Even with a naïve estimation of volatility, we were able to achieve performance much superior to Black-Scholes using the same set of features. By learning from historical options data, we were able to evade financial assumptions to view options pricing as a function that is well approximated by a neural network. We've found even better performance by forecasting bid and ask prices separately instead of the equilibrium, so future studies using historical data should consider this alternative problem formulation.

With more time, we would like to continue searching for optimal hyperparameters for our LSTM approach. We could also train models on the reverse problem to find the volatility implied by a given option price. This would allow us to plot the volatility surface and compare it with the volatility surfaces from the Heston model or GARCH family models. Additionally, we could do a deeper error analysis and examine pricing bias to see if our models perform better or worse given particular features (near the money, time until expiry, etc.), as well as estimating the partial derivatives of this model to draw curves of the risk metrics or Greeks and compare with Black-Scholes Greeks curves.

7 Contributions

Alexander: retrieved options data, created MLP models, performed literature review, produced tables and figures.

Andrew: preprocessing options data, producing the Black-Scholes baseline, producing LSTM model, designing poster.

Both members contributed to the progress report and final report.

References

[1] H. Amilon. A neural network versus black–scholes: a comparison of pricing and hedging performances. *Journal of Forecasting*, 22(4):317–335, 2003.

[2] S. Amornwattana, D. Enke, and C. H. Dagli. A hybrid option pricing model using a neural network for estimating volatility. *International Journal of General Systems*, 36(5):558–573, 2007.

[3] U. Anders, O. Korn, and C. Schmitt. Improving the pricing of options: a neural network approach. *Journal of Forecasting*, 17(5-6):369–388, 1998.

[4] J. Bennell and C. Sutcliffe. Black–scholes versus artificial neural networks in pricing ftse 100 options. *Intelligent Systems in Accounting, Finance and Management*, 12(4):243–260, 2004.

[5] F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973.

[6] CBOE. Monthly statistics report for index/other options — december 2018. `https://www.theocc.com/webapps/monthly-volume-reports`. Accessed: 2019-10-27.

[7] F. Chollet et al. Keras. `https://keras.io`, 2015.

[8] Z. A. Dindar and T. Marwala. Option pricing using a committee of neural networks and optimized networks. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, volume 1, pages 434–438 vol.1, Oct 2004.

[9] R. Garcia and R. Gençay. Pricing and hedging derivative securities with neural networks and a homogeneity hint. *Journal of Econometrics*, 94(1):93 – 115, 2000.

[10] S. Liu, C. Oosterlee, and S. Bohte. Pricing options and computing implied volatilities using neural networks. *Risks*, 7:16, 02 2019.

[11] M. Malliaris and L. Salchenberger. Beating the best: A neural network challenges the black-scholes formula. In *Proceedings of 9th IEEE Conference on Artificial Intelligence for Applications*, pages 445–449, March 1993.

[12] B. Mezofi and K. Szabo. Beyond black-scholes: A new option for options pricing, Feb 2019.

[13] L. Stark. Machine learning and options pricing: a comparison of black-scholes and a deep neural network in pricing and hedging dax 30 index options. 2017.

[14] P. Tino, C. Schittenkopf, and G. Dorffner. Financial volatility trading using recurrent neural networks. *IEEE Transactions on Neural Networks*, 12(4):865–874, July 2001.

[15] H. A. Trønnes. Pricing options with an artificial neural network: A reinforcement learning approach. 2018.

[16] U.S. Department of the Treasury. Daily treasury yield curve rates. `https://www.treasury.gov/resource-center/data-chart-center/interest-rates/pages/textview.aspx?data=yield`, 2019. Accessed: 2019-10-27.

[17] A. Vejendla and D. Enke. Evaluation of garch, rnn, and fnn models for forecasting volatility in the financial markets. 10, 05 2013.

[18] Wharton Research Data Services. Optionmetrics - option prices. `wrds.wharton.upenn.edu`, 2018. Accessed: 2019-10-27.

[19] Wharton Research Data Services. Optionmetrics - security prices. `wrds.wharton.upenn.edu`, 2018. Accessed: 2019-10-27.

A Appendix Figures

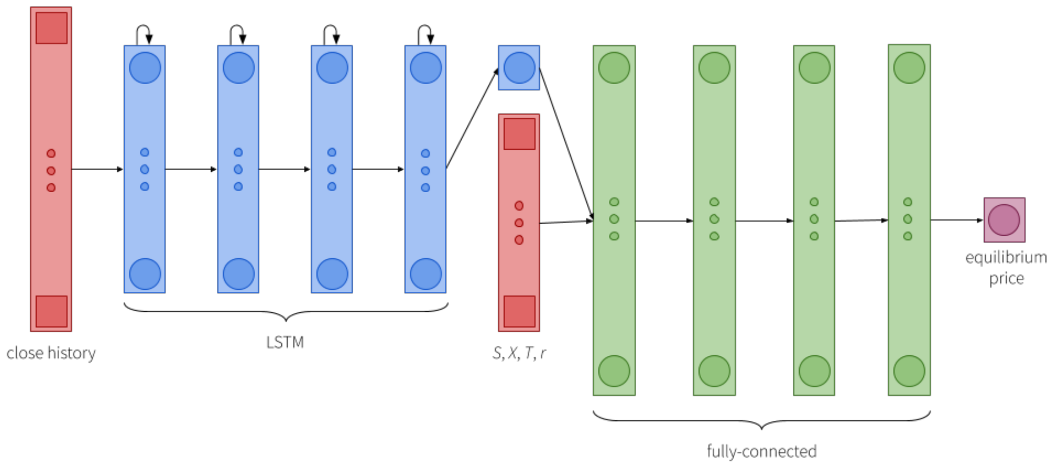
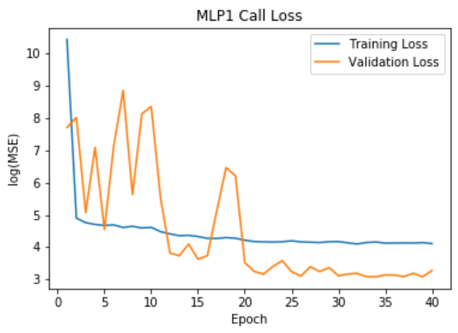
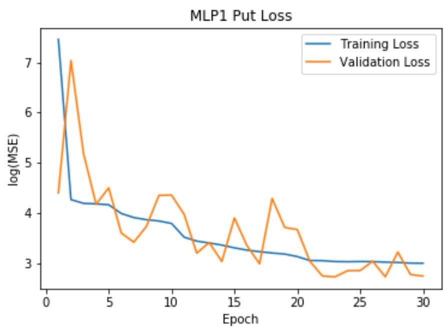


Figure 3: LSTM Architecture

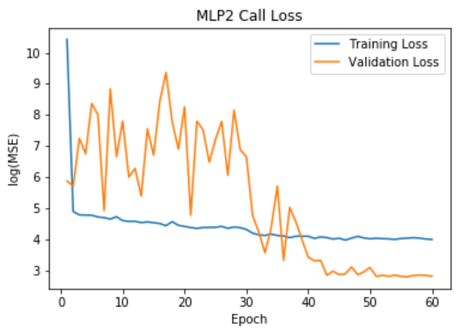


(a) MLP1 trained on call option data.

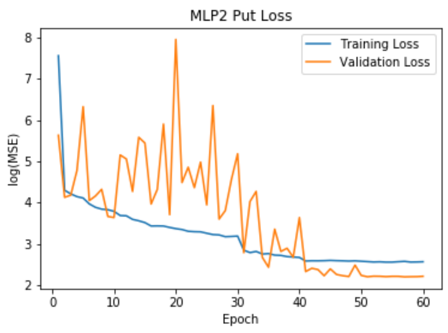


(b) MLP1 trained on put option data.

Figure 4: MSE over training of MLP1 models. Note the logarithmic y-axis.

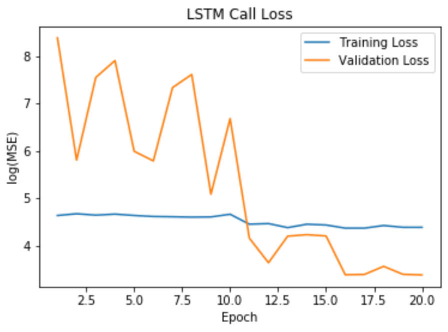


(a) MLP2 trained on call option data.

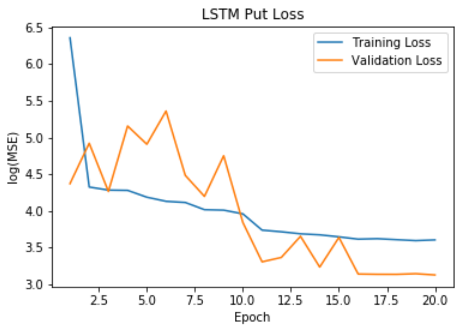


(b) MLP2 trained on put option data.

Figure 5: MSE over training of MLP2 models. Note the logarithmic y-axis.



(a) LSTM trained on call option data.



(b) LSTM trained on put option data.

Figure 6: MSE over training of LSTM models. Note the logarithmic y-axis.