Deep Learning Project

Deep Hedging

Niels Escarfail

nescarfail@student.ethz.ch
Student ID: 17-815-242

Dario Bolli

dbolli@student.ethz.ch
Student ID: 16-832-685

Guillaume Neven

gneven@student.ethz.ch
Student ID: 16-832-883

FS 2021

263-3210-00S: Deep Learning

Eidgenössische Technische Hochschule Zürich

Code : `https://gitlab.ethz.ch/gneven/Hand-in`

1 Problem Description

Hedging is a risk management strategy employed to balance losses in investments by taking an opposite position in a related asset. Mathematical finance provides a tractable solution to hedging in a frictionless and "complete market" models with risk-neutral pricing and hedging.

However, trading is subject to transaction costs, permanent market impact, and liquidity constraints in real markets, something not modelled by classical mathematical finance models. In our case, we are interested in hedging European call options [A.2] (European put options and American options would also be possible with slight modification of the code), by purchasing the appropriate amount of the underlying stock at each time.

Deep Hedging [1] helps to overcome these shortcomings. Deep hedging models the trading decisions in the hedging strategies as neural networks. It provides a model-free and efficient solution, enabling us to consider alternative factors such as market frictions. Deep Hedging models' feature sets consist of hedging instruments' prices and can contain additional information such as trading signals, news analytics, or past hedging decisions.

1.1 Financial setting

Some financial terms required to understand the underlying mechanisms behind deep hedging are reminded in the Appendix A.

1.2 Literature review

According to Ruf and Wang (2020) [2], while there are many papers on pricing using neural networks, only a few papers discuss hedging with neural networks.
Amilon (2003) [3] examined whether a neural network can price and hedge a call option better than the Black-Schole model with historical and implied volatility as a benchmark. They empirically revealed that neural network models outperform the benchmarks both in pricing and hedging performances.

Recently, Buehler et al. (2019) [1] developed Deep Hedging , a neural network framework for hedging under market frictions with convex risk measure as a loss function. Numerical experiments in Buehler et al. (2019) demonstrated that a deep hedging algorithm can feasibly be used to hedge a European option under exponential utility.

Deep Hedging [1] notably provides the theoretical framework for pricing and hedging using convex risk measures in discrete-time markets with frictions. Describing a market scenario generator, a loss function, market frictions and trading instruments. It includes the parametrization of appropriate hedging strategies by neural nets, and shows the surprising feasibility and accuracy of the method through several numerical experiments.

Several other contributions to deep hedging suggest different models, No Transaction Band Network [4], Deep Hedging under Rough Volatility [5], and Hedging Derivatives Under Generic Market Frictions Using Reinforcement Learning [6] to cite a few.

2 Contribution and implementation

Deep Hedging is a relatively recent method in finance; therefore, no clear benchmark on models' performances have yet been realized. With this in mind, several state-of-the-art deep learning models, with a focus on time series, were compared to a theoretical baseline.

We re-implement the metrics and benchmarks introduced by [6], such as the Profit-and-Loss, (Conditional) Value at Risk and delta plots to compare the models. Finally, we apply Deep-Hedging models on Brownian motions from generated stock and real-life data.

We base our implementation of Deep Learning algorithm to hedge derivatives on the deep hedging framework from YuMan-Tam's Github repository [7]. The input of our model is 100'000 option prices on a 30 days horizon, randomly generated using the Black Scholes model. The deep hedging model learns to take the actions which maximizes the wealth at the end of the time horizon. The model is composed of one strategy layer for each time step. At each time step, this layer takes the strategy of the previous time step and an information step derived by taking the log-normalized stock prices of the actual timestep, from which it infers a new strategy. After 30 days/timesteps, we compute and output the wealth.

The output layer is always a Dense layer. However, the input and hidden layers are modified to suit the desired deep learning model. The selected model are a MLP, a LSTM, a Transformer, all depicted during the lectures, and a Temporal Convolutional Network (TCN), a common time-series model, quickly depicted in appendix A.6 if unknown to the reader. In addition, we employ a finance-specific layer, called the No Transaction Band Network.

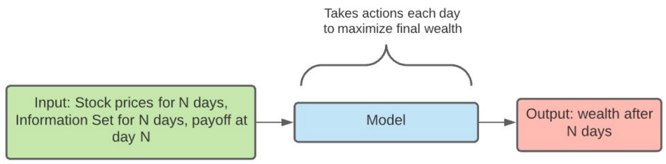
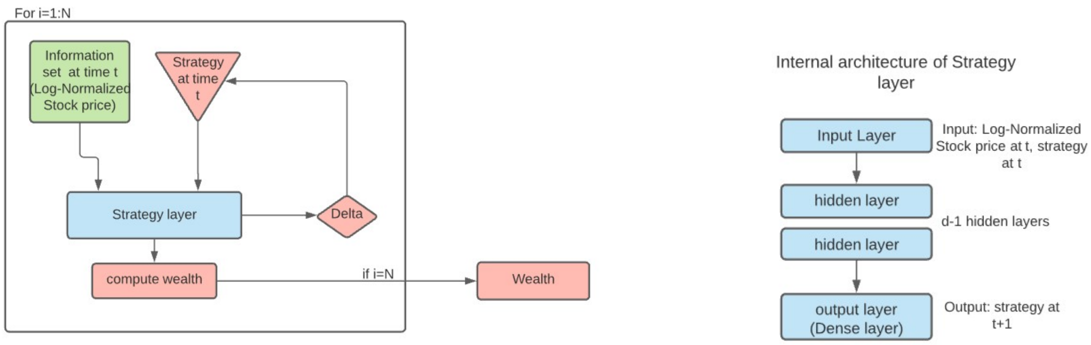


Figure 1: Deep Hedging Model



(a) Hedging Architecture

(b) Strategy layers Architecture

Figure 2: Model Architecture

The No Transaction Band Network In Deep Learning, a key component in the success of a model is to incorporate some proper inductive biases and task-specific structures in the model.

The No-Transaction Band Network introduced by Imaki et al. (2021) in [4] is directly inspired by the No-Transaction Band Strategy [8] [9]. This financial strategy consists in refraining from modifying the hedging decision unless the hedge ratio exceeds a range. It allows saving on transaction costs in a market with friction (costs). This strategy is proved to be optimal for European options and the exponential utility [9], and for a wider class of utilities and derivatives including exotics [4].
Another advantage of the no-transaction band network is that the output of the network is independent of the current hedging decision ratio. Whereas the classic deep hedging network's output is action dependent, that is the output depends on the previous hedging decision. Thus, the function space to explore in order to find the optimum is huge. The No-Transaction Band network does not use the current hedge ratio but rather only relies on the information set (the underlying price of the asset).
In practice, this strategy can be implemented by adding one layer to any neural network. The difference between a classic Deep Hedging architecture as we used previously and the No-Transaction Band architecture is illustrated in the following figures:

(a) Classic Deep hedging with a Neural Network [4]

(b) Deep hedging using No Transaction Band Network [4]

The No Transaction Band neural network outputs two values, which define a range $[b_l; b_u]$ around the Black Scholes delta value:

$$[b_l; b_u] = NN(I_{ti})$$

The hedging decision for the next time-step is obtained as follows: $\delta_{t_{i+1}} = clamp(\delta_{t_i}; b_l; b_u)$;

with the clamp function: clamp($\delta_{t_i}; b_l; b_u$) = $\begin{cases} \text{b}_l & \text{if } \delta_{t_i} < b_l, \\ \delta_{t_i} & \text{if } b_u >= \delta_{t_i} >= b_l, \\ \text{b}_l & \text{if } \delta_{t_i} > b_u, \end{cases}$

This strategy ensures that the hedge is always in the interval $[b_l; b_u]$. As long as the current hedge ratio is in the newly predicted interval, we do not modify the hedge ratio. When it is outside of the predicted interval, we change it to one of the boundary (depending on which one we crossed), and in this case, a transaction happens.

This layer was implemented on top of all the previously mentioned Deep Learning models (MLP, LSTM and TCN) except the Transformer, as the vanilla implementation already under performed. All models were tuned and selected parameters can be found in appendix A.7.

3 Results

3.1 Evaluation methods

We use the following metrics to evaluate and compare the models. We will be using the notations introduced in section A.4.

The predicted price is the additional minimal cash injection needed to implement the optimal hedge so that the overall position becomes acceptable for the agent. Theoretically speaking, as seller of options, the predicted price of the option should be close to the cost of the hedge. The optimality criterion we're predicting and aiming to optimize is this option price.

Profit-and-Loss (PnL) histograms At time T, given:

1. The hedging profit and loss: $(\delta, S)_T := \sum_{k=0}^{n-1} \delta_k.(S_k - S_{k-1})$
2. The value of the options owned: Z
3. The total cost of trading a strategy δ up to maturity $T : C_T(\delta) := \sum_{k=0}^{n} c_k(\delta_k - \delta_{k-1})$ [1]

We can finally define the agent's terminal profit and loss at time T as the hedging profits:

$$PL_T(Z, \delta) := (\delta.S)_T - Z - C_T(\delta)$$

PnL histograms display the agent's terminal portfolio value for each option path in the simulated test set. It is the distribution of the final wealth over multiple generated paths shifted by the risk free price (i.e. the best price once the final option price is known). In a market without transaction costs, the "perfect hedging" PnL is given by the Black-Schole model, the lower the required cost and variance in the resulting PnL histogram, the better the hedging. For later analysis, it need to be kept in mind that a good prediction is centered in 0 (i.e. neither too high or too low price), a low variance and a small left tail (i.e. extreme losses).

The Value at Risk (VaR) is a metric used to quantify the risk (extent and probability) of potential losses in a portfolio. The p-VaR is defined such that the probability of a loss greater than VaR is (at most) $(1 - p)$ while the probability of a loss less than VaR is (at least) p.
For a given portfolio, time horizon, and probability p, the p-VaR can be defined informally as the maximum possible loss during that time after excluding all worse outcomes whose joint probability is at most p.

The conditional Value at Risk (CVaR) is a risk assessment measure that quantifies the amount of tail risk of a portfolio. The CVaR is derived by taking a weighted average of the "extreme" losses in the tail of the distribution of the resulting profit and losses, beyond the Value at Risk (VaR) cutoff point.

The delta The delta, or decision, of our models can be compared to the theoretical one from the Black-Schole[A.4]. That is, the ratio between the change in the price of an option and the corresponding movement of the underlying asset's value. In addition, the effect of transaction costs on the delta are also analysed.

3.2 Hedging without transaction costs

Baseline The Black-Schole model gives a mathematical equation for pricing European call options. One of its major drawback is its assumption of the market being friction-less and transaction free. First, we simulate a friction-less stock

[1]To add trading costs, if the agent decides to buy a position $n \in \mathbf{R}^d$ in S at time t_k, we add a proportional cost $c_k(n)$.

dataset using Brownian motions; the Black-Schole model, therefore, gives the "perfect hedge". Further information about the Black-Schole model can be found in Appendix A.1. It will be used as a benchmark.

Figure 4 presents the various models' performance compared with the Black-Schole model. All models performed similarly, and no clear improvement was brought by the implementation of time-series. A more thorough comment will be made in section 4. It is also interesting to note the lack of performance of the Transformer. Indeed, the Transformer predicts less conservative prices (slightly shifted to the right) leading to potentially more competitive, but more risky options prices. This risk is seen as a long tail to the right, expressing very bad hedging, i.e. the price needed to cover the risk is far from sufficient.

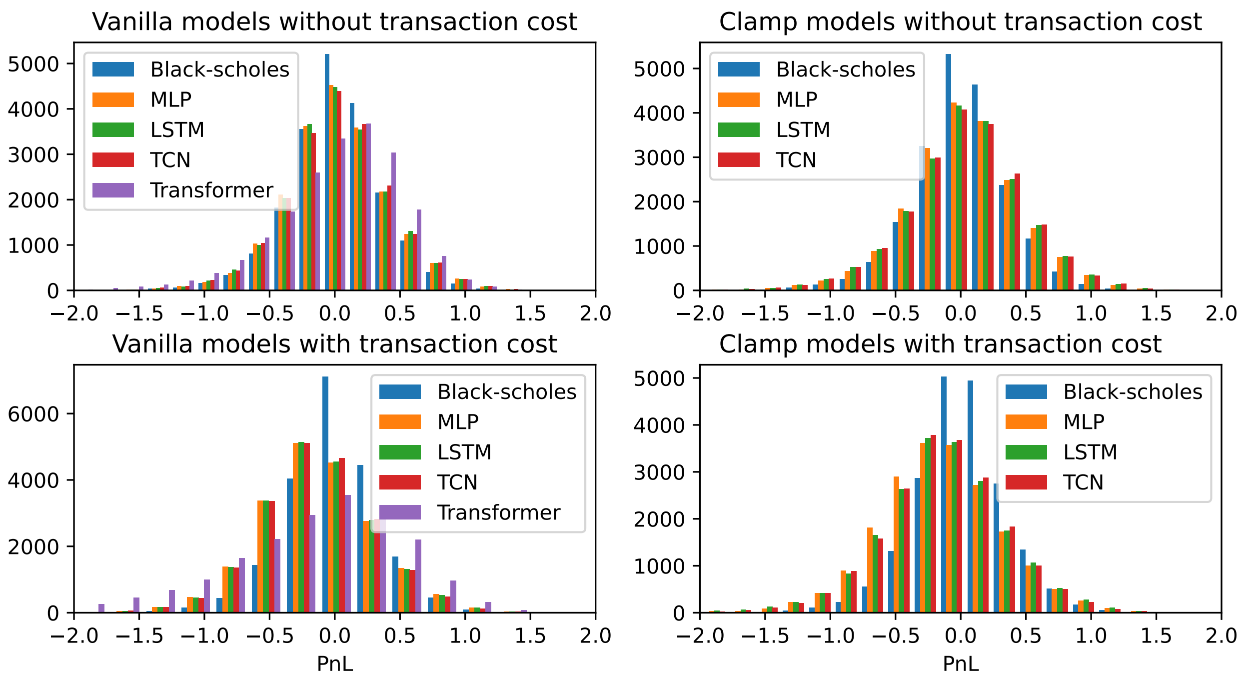


Figure 4: Profit and Loss

On the other hand, the results with transactions cost tell a different story. First, as expected, the Black-Schole is far too optimistic by the absence of transaction cost, as seen by its shift to the right. On the other hand, the other models all have meaningful predictions, except for the Transformer, once again. However, while the "Clamp" layer had no influence on pricing for no transactions costs scenarios, its mitigation of excessive buy/sell allows for better pricing. Indeed, the data from the right-hand side of the figure 4 is more centered around 0, whereas clamp-free PnLs are shifted to the left, expressing more conservative prices due to higher losses in transaction cost.

3.2.1 Impact of look-back window on the delta

In time-dependent models such as LSTM and TCN implementations of Deep-Hedging, we can study the impact of the look-back window parameter $maxT$ on the hedging results.

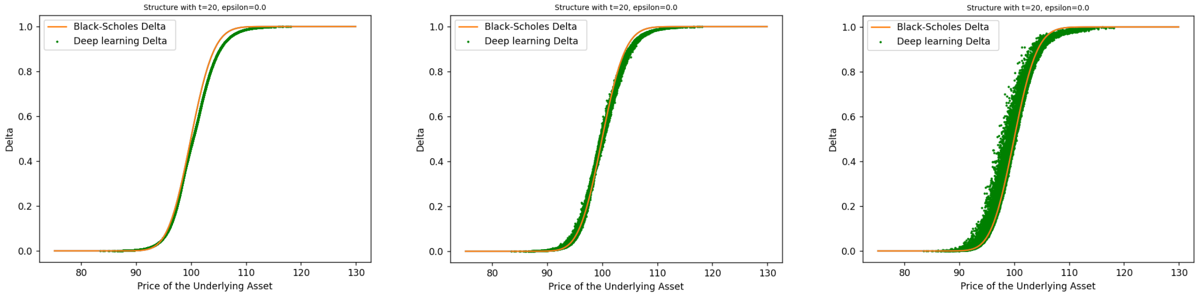
This parameter allows the network to take into account the Log-normalized stock prices and delta-strategy of the last $maxT$ timesteps when hedging at time step t.

We notice on Fig.5 that with a larger $maxT$, the network will not always take the same decisions for the same stock price (as it takes into account its last decisions), whereas its hedging decisions are deterministic on the stock price for $maxT = 1$.

Judging by the satisfying results of both non time-dependent (MLP) and time-dependent models (LSTM, TCN), we can say that taking into account past decisions is helpful, but not necessary to an investor for efficient hedging. Similarly, in the financial world, it is well known that past performance is not indicative of future results; however, market participants base their decisions on time-dependent factors such as trends when acting in financial markets.

3.2.2 Hedging with real market data

There is a lack of historical option prices data, making testing Deep Hedging models on real market data very difficult. Therefore existing researches avoid trying the models on real market data. However, it is relevant to try out these models on actual market data instead of simulated Brownian motions.

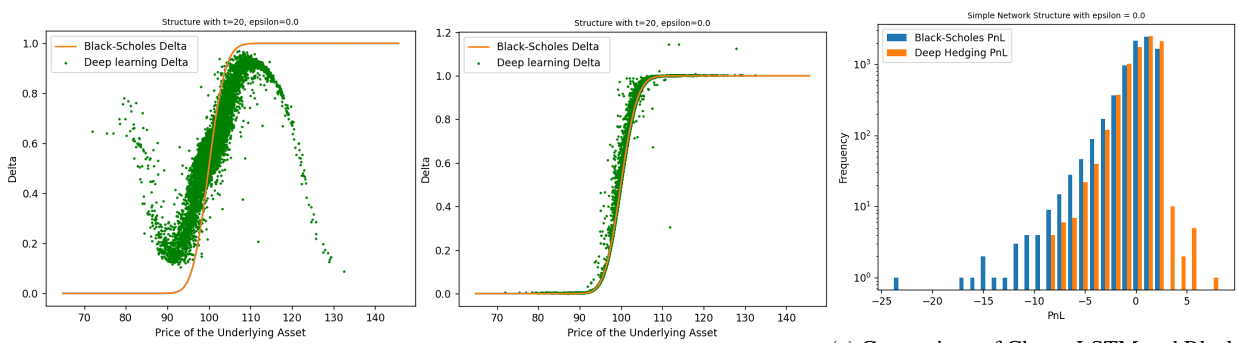


(a) $maxT = 1$ (b) $maxT = 6$ (c) $maxT = 11$

Figure 5: Impact of the look-back window $maxT$ on the resulting delta for the Clamp-LSTM network

To train and test the models on real market motions, we query the daily close prices of the S&P500 for the last ten years from the `Yahoo-Finance` [10] api. Doing so, we obtain 40000 price paths, which we normalize and rescale to fit the strike price of 100, as in the simulated datasets.

We compare the resulting delta-strategy (after 20 timesteps) of the LSTM network and of the LSTM-Clamp network. Note that the results shown below are without transaction costs, to be able to compare deep learning methods with the Black-Schole model.



(a) Deltas vanilla LSTM (b) Deltas LSTM Clamp (c) Comparison of Clamp-LSTM and Black-Schole PnL (log scale)

Figure 6: Deep Hedging results on real market data

Using real-market data, which is noisier than the generated data, we clearly see the Clamp-LSTM network is able to better approximate the delta than the "vanilla" implementation of the LSTM network (Fig.6b).

As postulated, the deep hedging models outperform the Black-Schole model on real market data (Fig.6). The Black-Schole model's assumptions (e.g. assuming constant volatility,...) make it hardly usable straight away in real-life situations. This shows the potential of deep hedging in actual financial markets.

4 Discussion and Conclusion

This work replicates and assembles the results of Deep Hedging [1] and No Transaction Band Network [4]. This allows a further exploration of new model architectures for deep hedging, using Temporal Convolutional Networks, Transformers, and new variants of No Transaction Band Networks.

The improvements brought by the No Transaction Band network implementations over their vanilla counterpart in an environment with transaction costs, were shown through their more accurate delta approximation and better PnL distribution.

Notably, the impact of various model parameters on the resulting hedge was investigated, giving further insight into the functioning of time-dependent models in this context. Yet, no clear improvement was noticed, even-though a clear change in the decision process happens, as shown in figure 5. This would imply that past information influences the deep hedging model, but leads to a similar conclusion. The nature of the financial dataset, where stocks are highly volatile and depend on surrounding environment (e.g. politics, shortage, pandemic, etc...) also increases the complexity of an already challenging problem.

References

[1] Hans Bühler, Lukas Gonon, Josef Teichmann, and Ben Wood. Deep hedging, 2018.

[2] Johannes Ruf and Weiguan Wang. Neural networks for option pricing and hedging: a literature review, 2020.

[3] Henrik Amilon. A neural network versus black-scholes: A comparison of pricing and hedging performances, 2003.

[4] Shota Imaki, Kentaro Imajo, Katsuya Ito, Kentaro Minami, and Kei Nakagawa. No-transaction band network: A neural network architecture for efficient deep hedging, 2021.

[5] Blanka Horvath, Josef Teichmann, and Zan Zuric. Deep hedging under rough volatility, 2021.

[6] Josef Teichmann Ben Wood Baranidharan Mohan Jonathan Kochems Hans Buehler, Lukas Gonon. Deep hedging: Hedging derivatives under generic market frictions using reinforcement learning, 2021.

[7] Yuman-Tam. Deep-hedging demo. `https://github.com/YuMan-Tam/deep-hedging`, 2020.

[8] M.H. Davis and A.R. Norman. Portfolio selection with transaction costs. *Mathematics of operations re- search*, 15(5):676–713, 1990.

[9] Panas V.G. Davis, M.H. and T. Zariphopoulou. European option pricing with transaction costs. *SIAM Journal on Control and Optimization*, 31:470–493, 1993.

[10] Yahoo finance. `https://finance.yahoo.com/`.

[11] Baichenjia. Tensorflow temporal convolutional network. `https://github.com/Baichenjia/Tensorflow-TCN`, 2019.

[12] Philippe Rémy. Keras temporal convolutional network. `https://github.com/philipperemy/keras-tcn.git`, 2021.

A Appendix: Hedging with derivatives

The most common way of hedging in the investment world is through derivatives. Derivatives are securities that move in correspondence to one or more underlying assets. There are many kinds of derivatives, the ones we are dealing with here are European Call options. The underlying assets can be stocks, bonds, commodities, currencies, indices or interest rates.

The Deep Hedging model aims to define the pricing of derivatives from the point of view of a bank. A well-known mathematical model used for derivatives pricing is the *Black-Schole* model.

A.1 The Black-Schole model

The Black-Schole model is a differential equation widely used to price options contracts. The Black-Schole model requires five input variables:

1. The strike price of an option K
2. The current stock price S_t
3. The time to maturity (or expiration) t
4. The risk-free interest rate r
5. The volatility σ

The Black-Schole model gives a mathematical equation for pricing European call options. The call option price C is given by:

$$C = N(d_1)S_t - N(d_2)Ke^{-rt}$$

with $d_1 = \frac{ln(\frac{S_t}{K})+(r+\frac{\sigma}{2})t}{\sigma\sqrt{t}}$ and $d_2 = d_1 - \sigma\sqrt{t}$.
With C the call option price, N the CDF of the normal distribution, S_t the spot price of an asset, K the strike price, r the risk-free interest rate, t the time to maturity, and σ the volatility of the asset.
Though usually accurate, the Black-Schole model makes certain assumptions (i.e no transaction costs) that can lead to prices that deviate from the real-world results. The standard Black-Schole model is only used to price European options, as it does not take into account that American options could be exercised before the expiration date.

In a situation without transaction cost, we use the Black-Schole model as a benchmark for testing our implementations.

A.2 European Call options

A *European Call option* is a version of an option contract, that gives the owner the right to acquire the underlying asset at a specific price at expiry.
The *strike price* (or exercise price) of a call option is the price at which the underlying asset can be bought at maturity. For an investor to profit from a call option, the stock's price, at expiry, has to be trading high enough above the strike price to cover the cost of the option premium.

A.3 Hedging with European Call options

Derivatives can be effective hedges against their underlying assets since the relationship between the two is more or less clearly defined. It's possible to use derivatives to set up a trading strategy in which a loss for one investment is mitigated or offset by a gain in a comparable derivative. Derivatives pricing is related to the downside risk in the underlying security. The downside risk is an estimate of the likeliness that the value of a stock will drop if market conditions change.

We are conducting our hedging from the point of view of a bank. Our goal is therefore to determine the pricing of European Call options which we are selling to investors.

When an outside investor has a short position in an underlying, and wishes to hedge against a potential upside, he purchases a call option and thus transfers the upside risk to the seller (us). To be able to do this, he has to pay a premium. In order to hedge this given upside risk, we purchase some shares (and *delta hedging* consists of estimating the proper ratio of these shares at each time period) of the stock on which he has taken the option.

If the stock price rises, the price of the option will go up and the investor will have gotten the best deal, as we would have priced the option higher for the higher underlying stock price. However, we counterbalance this loss with the shares of the stock that we bought and that have increased in value.
If the stock goes down, we might lose on the shares we bought, but the option's price will go down and we will profit from the premium paid by the investor.

An option's price depends on the strike price, the stock price, and the time left until maturity. At maturity, if the stock price has fallen, the client will not exercise his call option, and if the hedging is well done, we will not have bought too much of the stock, and we hope that the option price the investor paid is enough to balance the losses with the stock we have.
If the stock price rises, the client will buy the stock at the strike price. The investor will make a profit from us, the option *payoff*: `payoff = number_of_options_bought * (final_stock_price-strike_price)`. To balance this, if we have hedged well, we should have bought a lot of stock, which should offset the fact that the insurance became more expensive just before maturity.

A.4 Delta hedging and the delta

The delta is used to estimate the change in the option price as a function of the stock price. Since the call option price and the change in the stock price are opposite for us, the change in one is balanced by delta times the change in the other.

Delta hedging is an option's strategy that seeks to be directionally neutral [2] by establishing balancing in long and short positions in the same underlying asset. The investor tries to reach a *delta neutral* state and not have a directional bias on the hedge.

The ***delta*** is a ratio between the change in the price of an options contract and the corresponding movement of the underlying asset's value. It measures the number of shares of stocks needed to hedge an option position.

$$\delta = \frac{\Delta \text{cost of an option}}{\Delta \text{price of the underlying asset}}$$

Traders want to know an option's delta since it can tell them how much the value of the option or the premium will rise or fall following a change in the associated stock price. The theoretical change in premium for each unit or \$1 change in the price of the underlying is the delta.

The delta of a call option ranges between zero and one. The price of a call option with a delta of 0.40 is expected to rise by 40% if the underlying asset rises by \$1. A call option with a 0.50 delta has a strike that's equal to the stock's price. Therefore, for a call option to be profitable, it needs to have a delta between 0.5 and 1.

We can thus directly deduce the number of assets to buy to hedge again a call option from the delta. Our network estimates the delta at each time step, such that at the end of the 30 days, the deduced option price is as close as possible to the risk-neutral price. [3]

A.5 Mathematical setting

We consider a discrete-time financial market with:
- A finite time horizon T.
- Trading times(in our case in days) $0 = t_0 < t_1 < ... < t_n = T$.

The market contains d options with generated asset prices given by the stochastic process $S = (S_k)_{k=0,...,n}$.

Our portfolio of derivatives is a random variable Z meant to represent a mix of liquid and European call options. The maturity T is the maturity of all options, at which point all payments are known. The payoff of the derivative Z at maturity, with strike price K is: $Z(S) = max(S_T - K, 0)$.

In order to hedge a liability Z at T, we may trade in S using a stochastic process: $\delta = (\delta_k)_{k=0,...,n-1}$, with $\delta_k = (\delta_{k1}, ..., \delta_{kd})$.

Here, δ_{ki} denotes the agent's holdings of the ith asset at time t_k .

Each δ_k is subject to additional trading constraints, in our implementation, $\delta_k \in [0, 1]$.

From the side of a dealer of a derivative Z, the final wealth at maturity when performing hedging is:

$$P(-Z, S, \delta) = -Z(S) + (\delta * S)_T - C_T(S, \delta)$$

Where $(\delta * S)_T = \sum_{k=0}^{n=T} \delta_k * \Delta S_k$, and $C_k(S, \delta)$ is the total transaction costs until time t_k.

[2] the direction of the underlying stock price does not affect the results

[3] The risk-neutral price is defined as the price that solves the "Risk-Neutral Pricing Equation" (RNPE):
`price = discounted ‘‘expected’’ future payoff` , obtained by Itô's formula for risk-neutral pricing. Here it is negative due to the market drift.

Our objective as a dealer is to bring this final wealth as close as possible to 0 (if we want to bring it higher, then we have to ask too much to the client). The theoretical a posteriori best value we can achieve is the risk-neutral price. The objective for our network is then to be as close as possible to this theoretical price. Our network's predicted price, is what we will then ask to our clients (investors) as an option price, to be sure that we do not lose any money (and eventually win some if we ask for a higher price).

A.6 Temporal Convolutional Network

Temporal Convolutional Networks models (TCN) are an adaptation of Convolutional Neural Networks (CNN). Those models are often used in computer vision with translation independent features, i.e. that do not rely on their position in the image. To perform a local analysis on the images, only the neighbouring pixels are taken into account. A similar idea is applied temporally, where only the few past time steps are taken at the same time.

Figure 7 depicts this mechanism, where k successive inputs are taken at a time ($k = 2$ in fig 7). To allow longer dependencies, hidden layers can be added with their respective dilation factor d. To build such a network, 3 hyper-parameters are needed:

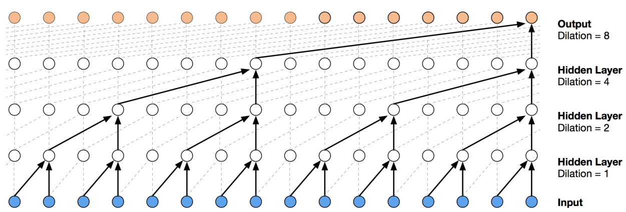


Figure 7: Latent Adversarial Network [11]

- The number of channel (or features) of each Hidden layers
- The kernel size k
- The dilation rate d (normally a kernel size multiple)

In practice, we replace the layers in the strategy by TCN layers. For the TCN layers, we used the package `keras-tcn` from [12] which has already taken care of designing a simple TCN layer.

A.7 Selected parameters

	Neurons	#layers	maxT	Activation function
MLP_CLAMP	30	2	-	Leaky Relu*
Transformer	15	2	6	Sigmoid
MLP	30	2	-	Sigmoid
LSTM	30	2	3	Sigmoid
LSTM_CLAMP	30	2	30	Leaky Relu*
TCN	30	2	6	Sigmoid
TCN_CLAMP	30	2	30	Leaky Relu*

*Similarly to [7], a slope of 0.001 was used.