Comparison of deep learning algorithms for forecasting stock returns and portfolio optimization

Mathieu-Savvas Dimitriades and Sarah Witzman

Introduction

- Can deep learning techniques effectively forecast future returns or sign of returns from past information?
- How can we build, based on those forecast, a tradable portfolio?
- Can this portfolio outperform the market?
- How can we evaluate the quality of a given model?

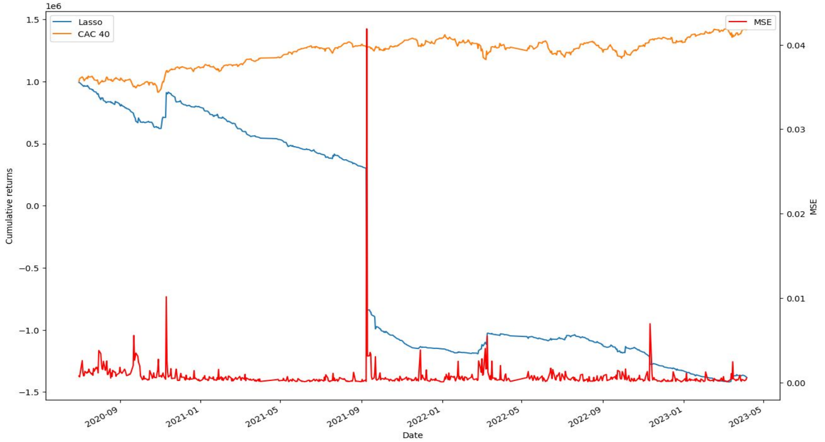
Data Preparation

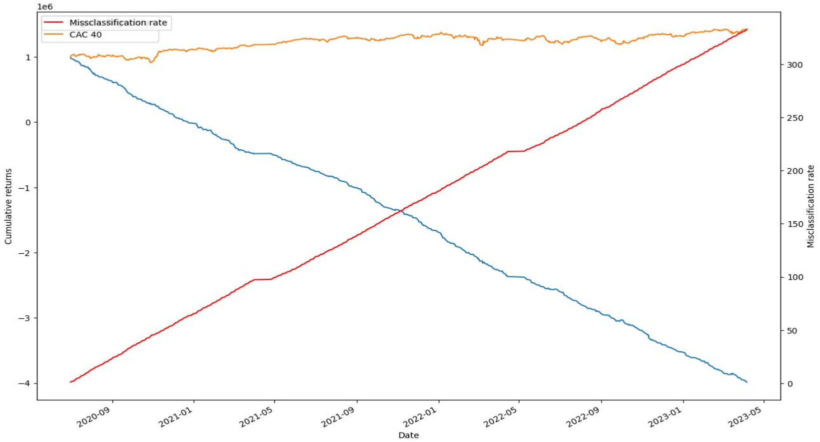
- CAC 40 stock data for June 1, 2009 to March 29, 2023
- Three types of signals: fundamental, macroeconomic and technical
- Drop stocks with >80% missing values
- Normalize with standard scaling

Baseline models

- Linear and logistic regression with LASSO penalty

Model	Yearly returns	Yearly volatility	Yearly Sharpe ratio	Total mean-squared error
LASSO	-92.2%	72.7%	-1.27	0.46
Model	Yearly returns	Yearly volatility	Yearly Sharpe ratio	Total misclassification error
Logistic Reg	-182.3%	13.4%	-13.6	50.4%





Models Used

- **Baseline models:** Linear and Logistic Regression
- **Deep learning models:**
  - Multi-layer Perceptron (MLP)
  - Convolutional Neural Network (CNN)
  - Recurrent Neural Network (RNN)
    - Long-Short Term Memory (LSTM)
    - Gated Recurrent Unit (GRU)
  - Time-Fusion Transformer (TFT)

MLP Architecture

- Varied dimensions of hidden layers and depth of network
- Mathematical model of hidden layers:

$$\mathbf{H} = \sigma_1(\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)})$$
$$\mathbf{O} = \sigma_2(\mathbf{H}\mathbf{W}^{(2)} + \mathbf{b}^{(2)})$$

- Hidden layers: ReLU activations
- Final layer:
    - Sigmoid activation for classification
    - Identity activation for regression

MLP Results

Regression Models	Yearly returns	Yearly volatility	Yearly Sharpe ratio	Total mean-squared error
10 layers of dimension 32	**-13.14%**	**204.2%**	**-0.06**	**11.74**
10 layers of dimension 64	-20.5%	207.3%	-0.09	11.43
50 layers of dimension 64	-2.9%	3.3%	-0.87	0.28
Classification Models	**Yearly returns**	**Yearly volatility**	**Yearly Sharpe ratio**	**Misclassification Error**
10 layers of dimension 32	-198.3%	19.5%	-10.15	0.51
10 layers of dimension 64	-167.6%	17.0%	-9.85	0.51
50 layers of dimension 64	**1.6%**	**12.1%**	**0.13**	**0.49**

CNN Architecture

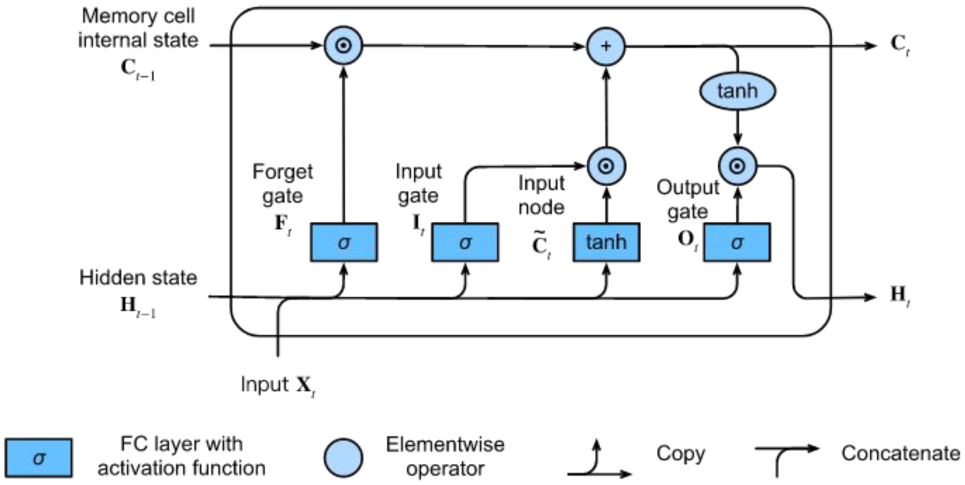
- Initial layer: linear layer mapping from feature space d to latent space of dimension p >> d with ReLu activation
- Subsequent hidden layers: Varying number of 1D-convolutions with identity activation
- Output linear layer:
    - Sigmoid activation for classification
    - Identity activation for regression

CNN Results

Regression Models	Yearly returns	Yearly volatility	Yearly Sharpe ratio	Total mean-squared error
1 layer of 64 channels	123.1%	265.3%	0.46	27.2
1 layer of 128 channels	-247.4%	564.3%	0.43	33.4
2 layers of 32, 64 channels	**404.1%**	**414.5%**	**0.97**	**25.9**
3 layers of 32, 64, 128 channels	243.1%	302.1%	0.80	29.0
Classification Models	**Yearly returns**	**Yearly volatility**	**Yearly Sharpe ratio**	**Misclassification Error**
1 layer of 64 channels	-7.1%	17.2%	-0.41	0.48
1 layer of 128 channels	**64.4%**	**66.1%**	**0.97**	**0.47**
2 layers of 32, 64 channels	-397.7%	33.4%	-11.9	0.54
3 layers of 32, 64, 128 channels	-147.9%	20.2%	-7.33	0.52

LSTM Architecture

- Input is passed through a gated memory cell with its own internal state
- Counteracts vanishing gradient problem
- Input gate, forget gate, and output gate are computed as:



$$
\begin{aligned}
\mathbf{I}_t &= \sigma(\mathbf{X}_t \mathbf{W}_{xi} + \mathbf{H}_{t-1} \mathbf{W}_{hi} + \mathbf{b}_i) \\
\mathbf{F}_t &= \sigma(\mathbf{X}_t \mathbf{W}_{xf} + \mathbf{H}_{t-1} \mathbf{W}_{hf} + \mathbf{b}_f) \\
\mathbf{O}_t &= \sigma(\mathbf{X}_t \mathbf{W}_{xo} + \mathbf{H}_{t-1} \mathbf{W}_{ho} + \mathbf{b}_o)
\end{aligned}
$$

$$
\begin{aligned}
\tilde{\mathbf{C}}_t &= \mathbf{tanh}(\mathbf{X}_t \mathbf{W}_{xc} + \mathbf{H}_{t-1} \mathbf{W}_{hc} + \mathbf{b}_c) \\
\mathbf{C}_t &= \mathbf{F}_t \odot \mathbf{C}_{t-1} + \mathbf{I}_t \odot \mathbf{C}_t \\
\mathbf{H}_t &= \mathbf{O}_t \odot \tanh(\mathbf{C}_t)
\end{aligned}
$$

Source: https://d2l.ai/chapter_recurrent-modern/lstm.html

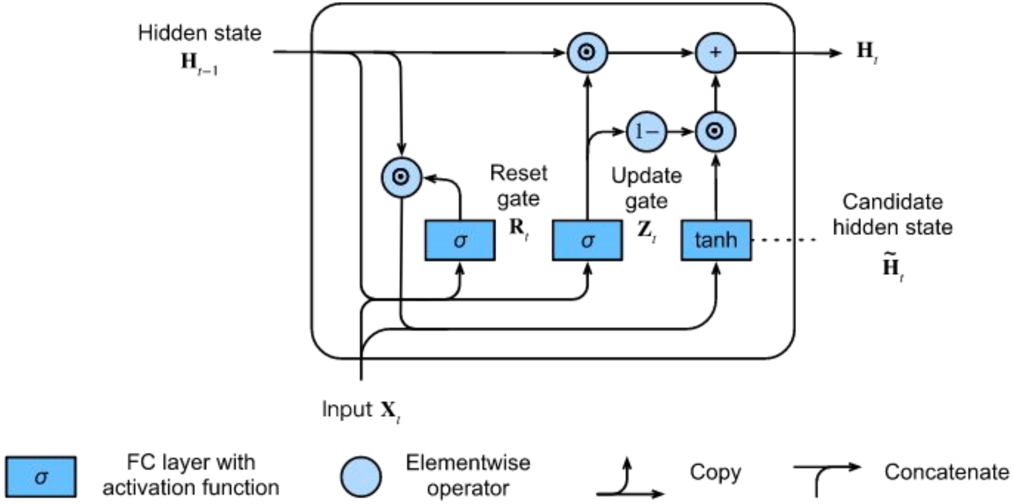
LSTM Results

Regression Models	Yearly returns	Yearly volatility	Yearly Sharpe ratio	Total mean-squared error
1 layer	**60.6%**	**188.4%**	**0.32**	**12.78**
2 layers	32.8%	140.8%	0.23	9.95
Classification Models	**Yearly returns**	**Yearly volatility**	**Yearly Sharpe ratio**	**Misclassification Error**
1 layer	**-376.3%**	**14.7%**	**-25.5**	**0.53**
2 layers	-412.9%	14.4%	-28.7	0.52

GRU Architecture

- Less computationally complex than LSTM
- Reset and update gates computed according to:

$$\mathbf{R}_t = \sigma(\mathbf{X}_t\mathbf{W}_{xr} + \mathbf{H}_{t-1}\mathbf{W}_{hr} + \mathbf{b}_r)$$
$$\mathbf{Z}_t = \sigma(\mathbf{X}_t\mathbf{W}_{xz} + \mathbf{H}_{t-1}\mathbf{W}_{hz} + \mathbf{b}_z)$$
$$\tilde{\mathbf{H}}_t = \tanh(\mathbf{X}_t\mathbf{W}_{xh} + (\mathbf{R}_t \odot \mathbf{H}_{t-1})\,\mathbf{W}_{hh} + \mathbf{b}_h)$$
$$\mathbf{H}_t = \mathbf{Z}_t \odot \mathbf{H}_{t-1} + (1 - \mathbf{Z}_t) \odot \tilde{\mathbf{H}}_t$$



GRU Results

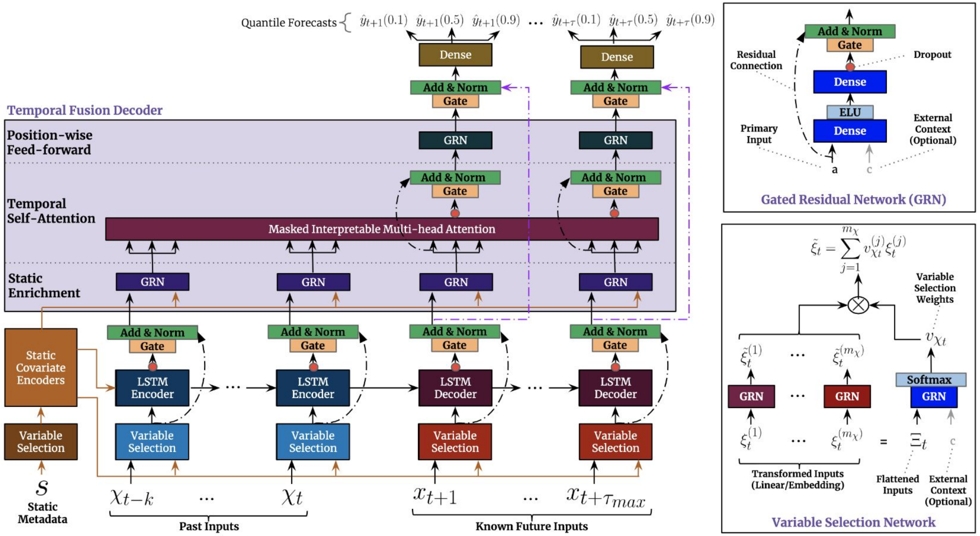
Regression Models	Yearly returns	Yearly volatility	Yearly Sharpe ratio	Total mean-squared error
1 layer	**-37.5%**	**125.7%**	**-0.29**	**13.33**
2 layers	-95.6%	76.5%	-1.25	10.42
Classification Models	**Yearly returns**	**Yearly volatility**	**Yearly Sharpe ratio**	**Misclassification Error**
1 layer	**-423.7%**	**17.2%**	**-24.6**	**0.55**
2 layers	-399.9%	14.4%	-27.8	0.49

TFT Architecture

$$\mathbf{D} = \{(\mathbf{k}_1, \mathbf{v}_1), \ldots, (\mathbf{k}_m, \mathbf{v}_m)\} \qquad \mathbf{f}(\mathbf{q}, \mathbf{D}) = \sum_{i=1}^{m} \alpha(\mathbf{q}, \mathbf{k}_i)\mathbf{v}_i$$

$$\forall i \in (1, \ldots, h), \mathbf{h}_i = f(\mathbf{W}_i^{(q)}\mathbf{q}, \mathbf{W}_i^{(k)}\mathbf{k}, \mathbf{W}_i^{(v)}\mathbf{v}) \in R^{p_v}$$

- Gating mechanisms
- Variable selection networks
- Static covariate encoders
- Interpretable multi-head attention
- Temporal processing
- Prediction intervals



Source: Lim, Arik, Loeff and Pfister, Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting, 2020

TFT results

	Yearly returns	Yearly volatility	Yearly Sharpe ratio	Total mean-squared error
TFT (regression)	-32.3%	10%	-3.21	0.39
	Yearly returns	**Yearly volatility**	**Yearly Sharpe ratio**	**Misclassification Error**
TFT (classification)	0.5%	9.4%	0.0055	49.1%

Portfolio Construction

For regression:

- Solve the portfolio optimization problem

$$\max_{\omega_t \in R^n} \alpha_t^T \omega_t - \frac{\lambda}{2} \omega_t{}^T \Sigma_t \omega_t \Rightarrow \omega_t^* = \frac{1}{\lambda} \Sigma_t^{-1} \alpha_t$$

- Use **exponential smoothing** to obtain the the historical covariance matrix of the returns
- Obtain precision matrix by using graphical LASSO:

$$\hat{\Theta} = \mathrm{argmin}_{\Theta \geq 0} \left(\mathrm{tr}(S\Theta) - \log \det(\Theta) + \lambda \sum_{j \neq k} |\Theta_{jk}| \right)$$

For classification: long stock with expected positive returns else short

Overall Results

- In regression
  - Highest Sharpe ratio and highest MSE for CNN
  - Beats market index
  - But **unreliable model**
- In classification
  - Highest Sharpe and lowest misclassification error for CNN
  - Beats market index
  - **Best overall model**

Table 7: Summary statistics - benchmark

	Yearly returns	Yearly volatility	Yearly Sharpe ratio
CAC 40	13.2%	18.9%	0.7

Table 8: Summary statistics - regression

Model	Yearly returns	Yearly volatility	Yearly Sharpe ratio	Total mean-squared error
LASSO	-92.2%	72.7%	-1.27	0.46
MLP	-2.9%	3.3%	-0.87	0.28
CNN	**404.1%**	**414.5%**	**0.97**	**25.94**
LSTM	32.8%	140.8%	0.23	9.95
GRU	-37.5%	125.7%	-0.29	13.33
TFT	-32.3%	10.0%	-3.2	0.39

Table 9: Summary statistics - classification

Model	Yearly returns	Yearly volatility	Yearly Sharpe ratio	Misclassification rate
Logistic Reg	-182.3%	13.4%	-13.6	50.4%
MLP	1.6%	12.1%	-0.13	48.5%
CNN	**64.4%**	**66.1%**	**0.97**	**47.2%**
LSTM	-376.3%	14.7%	-25.5	52.8%
GRU	-399.9%	14.4%	-27.8	49.0%
TFT	0.5%	9.3%	0.0055	49.1%

Future Work

- Further hyperparameter optimization
- Improve covariance matrix estimation methodology
- Perform 3-class classification to avoid predicting noise
- Application to different datasets or markets
- Comparison of combined models

Conclusion

- ***Can deep learning techniques effectively forecast future returns from past information?***
  - Deep learning models for regression can be used to predict market returns but we **could not find a regression model which has an acceptable mean-squared error while outperforming the market index**
  - They **significantly outperform standard linear models when it comes to predicting the sign of those returns** and we find that the CNN has the best overall prediction accuracy and Sharpe ratio
- ***How can we build, based on those forecast, a tradable portfolio? Can this portfolio outperform the market?***
  - Using portfolio optimization to build our portfolio leads to **good performance overall**
  - We were able to **outperform the CAC 40 benchmark** based on our portfolio construction methodologies
  - Our **best performing model yields a Sharpe ratio of 0.97 while the benchmark has a Sharpe ratio of 0.7 over the same period**
- ***How can we evaluate the quality of a given model?***
  - Statistical and risk-return performance metrics are **not always aligned**
  - They should therefore **both be considered** when assessing the portfolio performance

Questions?

References

Huang, J., Chai, J., Cho, S. Deep learning in finance and banking: A literature review and classification, 2020

Jean Dessain, Machine learning models predicting returns: Why most popular performance metrics are misleading and proposal for an efficient metric, Expert Systems with Applications, Volume 199, 2022

Hum Nath Bhandari, Binod Rimal, Nawa Raj Pokhrel, Ramchandra Rimal, Keshab R. Dahal, Rajendra K.C. Khatri, Predicting stock market index using LSTM, Machine Learning with Applications, Volume 9, 2022

Sarker, I.H. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. SN COMPUT. SCI. 2, 420, 2021

Cybenko, G., Approximation by superpositions of a sigmoidal function. Mathematics of control, signals and systems, 2(4), 303–314, 1989

Ian Goodfellow and Yoshua Bengio and Aaron Courville. Deep Learning. MIT Press. p. 326, 2016

Serkan Kiranyaz and Onur Avci and Osama Abdeljaber and Turker Ince and Moncef Gabbouj and Daniel J. Inman, 1D Convolutional Neural Networks and Applications: A Survey, 2019

Ashish Vaswani and Noam Shazeer and Niki Parmar and Jakob Uszkoreit and Llion Jones and Aidan N. Gomez and Lukasz Kaiser and Illia Polosukhin, Attention Is All You Need, 2017

George Bird and Maxim E. Polivoda, Backpropagation Through Time For Networks With Long-Term Dependencies, 2021

References

Bengio, Simard and Frasconi, Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5(2), 157-166, 1994

Hochreiter and Schmidhuber, Long short-term memory, Neural computation, 9(8), 1735–1780, 1997

Cho, Van Merri ̈enboer, Bahdanau and Bengio. On the properties of neural machine translation: encoder-decoder approaches, 2014

Jimmy Lei Ba, Jamie Ryan Kiros and Geoffrey E. Hinton, Layer Normalization, 2016

Bryan Lim, Sercan O. Arik, Nicolas Loeff and Tomas Pfister, Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting, 2020

Harry Markowitz, Portfolio Selection, The Journal of Finance, Vol. 7, No. 1., pp. 77-91, 1952

Lasse Heje Pedersen, Abhilash Babu, and Ari Levine, Enhanced Portfolio Optimization, Financial Analysts Journal, 77(2): 124-151, 2021

Jerome Friedman, Trevor Hastie and Robert Tibshirani, Sparse inverse covariance estimation with the graphical lasso, 2007

Hendrik Bessembinder, Trade Execution Costs and Market Quality after Decimalization, The Journal of Financial and Quantitative Analysis, Vol. 38, No. 4, pp. 747-777 (31 pages), 2003

Bell, F., Smyl, S., 2018. Forecasting at Uber: An introduction. Accessed on 2023-04-23. https://eng.uber.com/forecasting-introduction/

https://optuna.org/

References

Wei Bao, Jun Yue and Yulei Rao, A deep learning framework for financial time series using stacked autoencoders and long-short term memory, 2017

Jingyi Shen and M. Omair Shafiq, Short-term stock market price trend prediction using a comprehensive deep learning system, 2020

Ryo Akita, Akira Yoshihara , Takashi Matsubara, Kuniaki Uehara, Deep learning for stock prediction using numerical and textual information, 2016

M. Nabipour, P. Nayyeri, H. Jabani, A. Mosavi, E. Salwana, and S. S., “Deep Learning for Stock Market Prediction,” Entropy (Basel), vol. 22, no. 8, p. 840, Jul. 2020, doi: 10.3390/e22080840.

Mukherjee, S., et al.: Stock market prediction using deep learning algorithms. CAAI Trans. Intell. Technol. 8(1), 82– 94 (2023). https://doi.org/10.1049/cit2.12059

B L, S. and B R, S. (2023), ”Combined deep learning classifiers for stock market prediction: integrating stock price and news sentiments”, Kybernetes, Vol. 52 No. 3, pp. 748-773. https://doi-org.ezproxy.princeton.edu/10.1108/K-06-2021-0457

K. Rekha and M. Sabu, “A cooperative deep learning model for stock market prediction using deep autoencoder and sentiment analysis,” PeerJ Comput Sci, vol. 8, p. e1158, Nov. 2022, doi: 10.7717/peerj-cs.1158.

Jimmy Lei Ba and Jamie Ryan Kiros and Geoffrey E. Hinton, Layer Normalization, 2016