Visual Time Series Forecasting: An Image-driven Approach

Naftali Cohen*
J. P. Morgan AI Research
New York, NY
naftali.cohen@jpmchase.com

Srijan Sood
J. P. Morgan AI Research
New York, NY
srijan.sood@jpmchase.com

Zhen Zeng
J. P. Morgan AI Research
New York, NY
zhen.zeng@jpmchase.com

Tucker Balch
J. P. Morgan AI Research
New York, NY
tucker.balch@jpmchase.com

Manuela Veloso
J. P. Morgan AI Research
New York, NY
manuela.veloso@jpmchase.com

ABSTRACT

In this work, we address time-series forecasting as a computer vision task. We capture input data as an image and train a model to produce the subsequent image. This approach results in predicting distributions as opposed to pointwise values. To assess the robustness and quality of our approach, we examine various datasets and multiple evaluation metrics. Our experiments show that our forecasting tool is effective for cyclic data but somewhat less for irregular data such as stock prices. Importantly, when using image-based evaluation metrics, we find our method to outperform various baselines, including ARIMA, and a numerical variation of our deep learning approach.

CCS CONCEPTS

• **Computing methodologies** → **Image representations**; • **Mathematics of computing** → **Time series analysis**.

KEYWORDS

time-series forecasting, Image representations, ARIMA, visualizations

ACM Reference Format:
Naftali Cohen, Srijan Sood, Zhen Zeng, Tucker Balch, and Manuela Veloso. 2021. Visual Time Series Forecasting: An Image-driven Approach. In *MileTS '21: 6th KDD Workshop on Mining and Learning from Time Series, August 14th, 2021, Singapore.* ACM, New York, NY, USA, 5 pages. https://doi.org/XX.XXXX/XXXXXXXX.XXXXXXXX

1 INTRODUCTION AND RELATED WORK

Time series forecasting is a standard statistical task that concerns predicting future values given historical information. Conventional forecasting tasks range from uncovering simple periodic patterns to forecasting intricate nonlinear patterns. The prevailing and most widely used forecasting techniques include linear regression, exponential smoothing, and ARIMA (e.g., [10, 17, 22]). In recent years, modern approaches emerge as tree-based algorithms, ensemble methods, neural network autoregression, and recurrent neural networks (e.g., [10]). These methods are useful for highly nonlinear and inseparable data but are often considered less stable than the more traditional approaches (e.g., [17, 18]).

In the last few years, deep learning approaches have been applied in the domain of time series analysis, for forecasting [4, 11, 27, 28], as well as unsupervised approaches for pre-training, clustering, and distance calculation [1, 24, 29, 31]. The common theme across these works is their use of stacked autoencoders (with different variations – vanilla, convolutional, recurrent, etc.) on numeric time series data. Autoencoders have also shown promise in the computer vision domain across tasks as image denoising [3, 14], image compression [2], and image completion and in-painting [20, 23].

This paper follows these studies and presents a new perspective on numerical time series forecasting by transforming the problem completely into the computer-vision domain. We capture input data as images and build a network that outputs corresponding subsequent images. To the best of our knowledge, this is the first study that aims at explicit visual forecasting of time series data as plots. Previous researches leveraged computer vision for time-series data but focused on classifying trade patterns [7, 8], numeric forecast [6], learning weights to combine multiple statistical forecasting methods [19], and video prediction for multivariate economic forecasting [32]. We follow up on these approaches but focus on an explicit regression-like image prediction task.

This work presents a few advantages. Visual time series forecasting is a data-driven non-parametric method, not constrained to a predetermined set of parameters. Thus, the approach is flexible and adaptable to many data forms, as shown by application across various datasets. This bears a stark contrast with classical time series forecasting approaches that are often tailored to the particularity of the data in hand. The main advantage of this method is that its prediction is independent of other techniques. This is important as it was repeatedly shown that an aggregate of independent techniques outperforms the best-in-class method (e.g., [10, 12, 15]). Secondly, visual predictions result in inherent uncertainty estimates as opposed to pointwise estimates, as they represent distributions over pixels as opposed to explicit value prediction. In addition, financial time series data are often presented and act upon without having access to the underlying numeric information (e.g., financial trading using the smartphone applications). Thus, it seems viable to examine the value in inferring using visualizations alone. Lastly, as

MileTS '21, August 14th, 2021, Singapore

ACM ISBN XXX-X-XXXX-XXXX-X/XX/XX...$15.00
https://doi.org/XX.XXXX/XXXXXXXX.XXXXXXXX

will be discussed later on, we show that transforming the continuous numeric data to a discrete bounded space using visualization results in robust and stable predictions. We evaluate predictions using multiple metrics. When considering object-detection metrics such as Intersection-over-Union (IoU), visual forecasting outperforms the corresponding numeric baseline. However, when utilizing more traditional time-series evaluation metrics as the symmetric mean absolute percentage error (SMAPE), we find the visual view to perform similarly to its numerical baselines.

2 DATASETS

This paper uses four datasets - two synthetic and two real - with varying degrees of periodicity and complexity to examine the utility of forecasting using images. Each dataset consisted of approximately 40-50k training samples, 4-5k validation samples, and 15k testing samples. Figure 1 shows examples of the data, and the supplementary material contains a detailed description of each of the datasets and how they were curated.

Harmonic: multi-periodic data sampled from harmonic functions. It is derived synthetically with a linearly additive two-timescale harmonic generating function consisting of sine waves on two timescales: short oscillations that are composed on a much longer wave trains.

OU: synthesized using mean-reverting time series based on Ornstein–Uhlenbeck (OU) processes [5]. These often resemble characteristics of financial interest rates or volatility: noisy on finer scales but predictable on the larger scale.

ECG: signals measured from 17 different people adopted from MIT-BIH Normal Sinus Rhythm Database [13]. The data has prominent spikes about every second, which makes the data predictable. However, there is noticeable noise between spikes that is much harder to predict.

Financial: stock data from Yahoo! Finance consisting of daily Adjusted Close values of stocks that contributed to the S&P-500 index from 2000-2019. Each time series segment consists of 80 days and is generally considered notoriously hard to predict [26].

3 PROBLEM STATEMENT

Given a time series signal, our goal is to produce a visual forecast of its future. We approach this problem by first converting the numeric time series into an image (detailed procedure described in supplementary material), and then producing a corresponding forecast image using deep-learning techniques. By doing so, we obtain an image in which the pixel values in each column sum to 1; each column can be perceived as a discrete probability distribution (see Figure 2). Columns represent the independent variable time, while rows capture the dependent variable: pixel intensity. The value of the time series S at time t is now simply the pixel index r (row) at that time (column) with the highest intensity.

Let X be the set of images of input time series signals, and Y be the set of corresponding forecast output images. The overlap constant c defines the overlap fraction between the input image $x \in X$ and the forecast $y \in Y$, where $c = 1$ implies $x = y, \forall x \in X$, and $c = 0$ implies that $x \cap y = \emptyset, \forall x \in X$, i.e., x and y are distinct. In our experiments, we use $c = 0.75$ which means the first 75% of the forecast image y is simply a reconstruction of the later 75% of the input image x, and the rest 25% of y corresponds to visual forecasting of the future. We chose $c = 0.75$ such that the reconstructed overlap region (first 75% in y) serves as a sanity check on the effectiveness of a forecasting method, and the prediction region (later 25% in y) provides forecasting into the near future. Please refer to the supplementary material for an illustration.

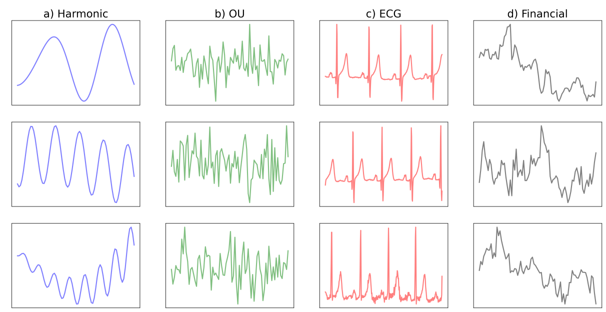

Figure 1: Sampled examples of the four datasets: Harmonic, OU, ECG, and Financial.

4 METHOD

4.1 Image-to-Image Regression

As mentioned in Section 1, recent work has seen the extensive use of autoencoders in both the time series and computer vision domains. Following these, we extend the use of autoencoders to our image-to-image time series forecasting setting. We use a simplistic convolutional autoencoder to produce a visual forecast image with the continuation of an input time series image, by learning an undercomplete mapping $g \circ f$,

$$\hat{y} = g(f(x)),\ \forall x \in X,$$

where the encoder network $f(\cdot)$ learns meaningful patterns and projects the input image x into an embedding vector, and the decoder network $g(\cdot)$ reconstructs the forecast image from the embedding vector. We purposely do not use sequential information or LSTM cells as we wish to examine the benefits of framing the regression problem in an image setting. This can later be extended to more complex architectures.

We call this method **VisualAE**. We used 2D convolutional layers with a kernel size of 5×5, stride 2, and padding 2. All layers are followed by ReLU activation and batch normalization. The encoder network consists of 3 convolutional layers which transform a $80 \times 80 \times 1$ input image to $10 \times 10 \times 512$, after which we obtain an embedding vector of length 512 using a fully connected layer. This process is then mirrored for the decoder network, resulting in a forecast image of dimension 80×80. We include a diagram illustrating this architecture in the supplementary material.

4.2 Loss Functions

We care about the likelihood of pixel intensity in a particular location (row) in each column of the forecast image. This can be achieved by leveraging metrics that compare two probability distributions. We do so in a column-wise manner: the loss L to compare

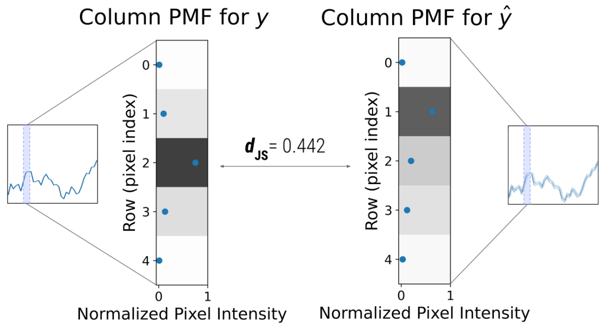

Figure 2: A depiction of comparison of two sample column probability distributions $y = [0.01, 0.1, 0.75, 0.13, 0.01]$ **and** $\hat{y} = [0.02, 0.63, 0.2, 0.12, 0.03]$**.**

target ground-truth (GT) image y with prediction image $\hat{y}$ is the sum of column-wise distances between the two,

$$L(y, \hat{y}) = \sum_{i=1}^{w} d(y_i, \hat{y}_i),$$

where $y_i, \hat{y}_i$ are the ith column in the ground truth and forecast images, d is any distance measure between two distributions (y_i and $\hat{y}_i$ in this case), and w is the width of images. This process is depicted in Figure 2.

Measures such as the Kullback-Leibler Divergence have been extensively used as loss functions ([15]), as they provide a way of computing the distance from an *approximate distribution* Q to a *true distribution* P. In this study, following [16], we choose d to be the **Jensen-Shannon Divergence** (JSD), which is a symmetric, more stable version of the Kullback-Leibler Divergence having the property that $D_{JS}(P\|Q) = D_{JS}(Q\|P)$. Here, JSD is computed as

$$D_{JS}(P\|Q) = \frac{1}{2}D_{KL}(P\|M) + \frac{1}{2}D_{KL}(Q\|M)$$

where $M = \frac{1}{2}(P + Q)$.

5 EXPERIMENTS

We experimented with four datasets: Harmonic, OU, ECG, and Financial, as they cover a wide range of complexity and predictability in time series data (illustrated in the dataset analysis in supplementary). In this study we used PyTorch Lightning [9, 25] for implementation and Nvidia Tesla T4 GPUs in our experiments. As described in Section 3, there is a 75% overlap between input and output. In our experiments, each sample contains 80 datapoints; we aim to forecast the last 20 datapoints (last 25%) of the output image. We benchmark the proposed method against three baseline methods.

5.1 Methods

We summarize the benchmarked methods as following. Please refer to a thorough description (including training details, and data preprocessing) of each method in the supplementary material.

- **VisualAE**: This is the proposed method as discussed in Section 4.1. We train on images with size 80×80.
- **NumAE** (Numeric AE): We also train an autoencoder network to produce numeric forecasts of the original numerical time series signal.
- **ARIMA**: Autoregressive Integrated Moving Average (ARIMA) models are a class of methods that are designed to capture autocorrelations in the data.
- **RandomWalk**: We used the random walk without drift model as a naive numeric forecasting baseline for comparison ([30]).

5.2 Forecast Accuracy Metrics

We use a variety of measures to assess the accuracy of forecast predictions from each method. Some of these metrics are extensively used in the time series forecasting domain, whereas the others we extend from the overarching machine learning field to this task.

The baseline methods **ARIMA**, **NumAE** and **RandomWalk** produce continuous numeric forecasts, whereas our method **VisualAE** produces an image. Accordingly, we convert this image back to a numeric forecast which we can use to assess predictions using the metrics described in Section 5.2.1. Similarly, to leverage the image based metrics described in Section 5.2.2, we transform the numeric predictions of the baseline methods into images. We discuss the interplay between these metrics across Section 5.3, along with in-depth discussions and insights regarding comparisons between numeric and image based metrics in the supplementary material.

5.2.1 ***Numeric Measures.*** We use the Symmetric Mean Absolute Percentage Error (**SMAPE**), and the Mean Absolute Scaled Error (**MASE**) to evaluate the numeric forecasts. These two metrics are widely used in the literature for forecast accuracy evaluation [21]. Please refer to supplementary materials for equations of these metrics.

5.2.2 ***Image based Measures.*** In addition to utilizing tradition forecasting error metrics, we can measure the similarity between the predicted image and the ground-truth image in our setting to evaluate forecast accuracy. We use Jensen-Shannon Divergence (**JSD**), which is the same as the loss described in Section 4.2. In addition, we use an extended version of Intersection-over-Union (**IoU**) to measure image similarity columnwise. We first obtain the 1D bounding box of non-zero pixels for each column, then compute the IoU between bounding boxes of each corresponding column in the ground-truth and predicted images. This ranges from 0.0 to 1.0, with higher values indicating better forecasts.

5.3 Results

All reported metrics mentioned in Section 5.2 are over the unseen future prediction region. For both **VisualAE** and **NumAE**, we averaged these metrics over five independently trained models with different random weight initializations. We demonstrate that the proposed method **VisualAE** outperforms baseline methods **NumAE**, **RandomWalk**, and **ARIMA** across all four datasets when evaluated using image-based metrics (such as IoU). However, as we will discuss in this section, traditional numeric metrics are inconsistent with this finding. We demonstrate the value of using a visual

	Method	SMAPE $\mu \pm \sigma$	MASE $\mu \pm \sigma$	IoU $\mu \pm \sigma$	JSD $\mu \pm \sigma$
Harmonic	RandomWalk	1.239 ± 0.440	5.106 ± 3.405	0.179 ± 0.060	0.501 ± 0.043
	NumAE	**0.480 ± 0.297**	**1.258 ± 1.081**	0.423 ± 0.107	0.334 ± 0.103
	ARIMA	0.580 ± 0.398	2.694 ± 3.350	0.447 ± 0.238	0.343 ± 0.186
	VisualAE	0.527 ± 0.303	1.518 ± 1.482	**0.460 ± 0.088**	**0.271 ± 0.115**
OU	RandomWalk	**0.018 ± 0.069**	1.007 ± 0.385	0.257 ± 0.021	0.381 ± 0.019
	NumAE	**0.014 ± 0.056**	471.411 ± 8486.706	0.165 ± 0.076	0.543 ± 0.052
	ARIMA	**0.014 ± 0.056**	**0.736 ± 0.133**	0.141 ± 0.014	0.556 ± 0.011
	VisualAE	**0.014 ± 0.060**	0.748 ± 0.119	**0.469 ± 0.017**	**0.257 ± 0.010**
ECG	RandomWalk	1.173 ± 0.463	1.551 ± 1.384	0.164 ± 0.014	0.501 ± 0.021
	NumAE	1.097 ± 0.200	0.979 ± 0.280	0.278 ± 0.047	0.463 ± 0.051
	ARIMA	1.409 ± 0.305	1.535 ± 1.688	0.160 ± 0.011	0.576 ± 0.009
	VisualAE	**0.596 ± 0.254**	**1.658 ± 0.321**	**0.485 ± 0.022**	**0.230 ± 0.041**
Financial	RandomWalk	**0.036 ± 0.028**	**3.364 ± 2.217**	0.186 ± 0.054	**0.475 ± 0.050**
	NumAE	**0.036 ± 0.028**	**3.364 ± 2.205**	0.132 ± 0.069	0.598 ± 0.059
	ARIMA	**0.042 ± 0.035**	4.034 ± 14.697	0.119 ± 0.072	0.606 ± 0.063
	VisualAE	**0.043 ± 0.028**	4.007 ± 2.084	**0.212 ± 0.080**	0.511 ± 0.070

Table 1: Summary of various metrics on out-of-sample data with mean ± standard deviation for the forecast region.Lower SMAPE/MASE/JSD error (or higher IoU score) implies better prediction accuracy.

approach to time-series forecasting, and how image-based evaluation metrics can help address some of the caveats of traditional numeric metrics.

We report the mean and standard deviation of various prediction accuracy metrics in Table 1. **VisualAE** achieves higher IoU scores than all baselines across the four datasets. The same holds true for JSD (with the exception of **RandomWalk** scoring better in the Financial dataset). The numeric metrics are often inconsistent – within themselves (SMAPE and MASE) – as well as across the four datasets. According to the numeric metrics, **VisualAE** is a close second (if not similar) to **NumAE**, with the exception of the ECG dataset, where **VisualAE** performs the best, and the OU dataset, where **ARIMA** and **VisualAE** perform similarly to **NumAE** . Please refer to our supplementary material for a more detailed discussion on the characteristics of benchmarked methods, along with qualitative examples.

5.4 Numeric vs. Image based Metrics

In Table 1, numeric metrics are often inconsistent with the image-based ones, and sometimes do not agree amongst each other (e.g., SMAPE & MASE values for OU dataset). They are sensitive and often fail to recognize good quality forecasts (e.g., **RandomWalk** reportedly performing the best for the Financial dataset). Picking a percentage error such as SMAPE also carries the inability to compare forecast method quality across series (e.g., low errors in the Financial dataset do not capture that it is the most challenging to predict).

The IoU metric is able to capture this information across the datasets, along with preserving rank-ordering of forecast quality amongst the four methods. As shown in Figure 3, the IoU metric is better at discerning which forecast better captures ground-truth trends. This is evident with higher IoU values when the visual shape of predictions matches the ground truth well. We believe using a two-pronged approach of utilizing both numeric and visual approaches holds immense value for the field of time series forecasting.

6 SUMMARY AND CONCLUSION

To the best of our knowledge, this study is the first to explicitly forecast time series using visual representations of numeric data. We show that image-based measures can capture prediction quality more consistently than traditional numeric metrics. The proposed visual forecasting approach, albeit simplistic, performs well across datasets. Our findings show promising results for both periodic time series (including abrupt spikes in ECG) and irregular financial data. We believe that leveraging visual approaches holds immense promise for the field of time series forecasting in the future, especially when used in conjunction with traditional methods.

REFERENCES

[1] Abubakar Abid and James Y Zou. 2018. Learning a warping distance from unlabeled time series using sequence autoencoders. In *Advances in Neural Information Processing Systems*. 10547–10555.

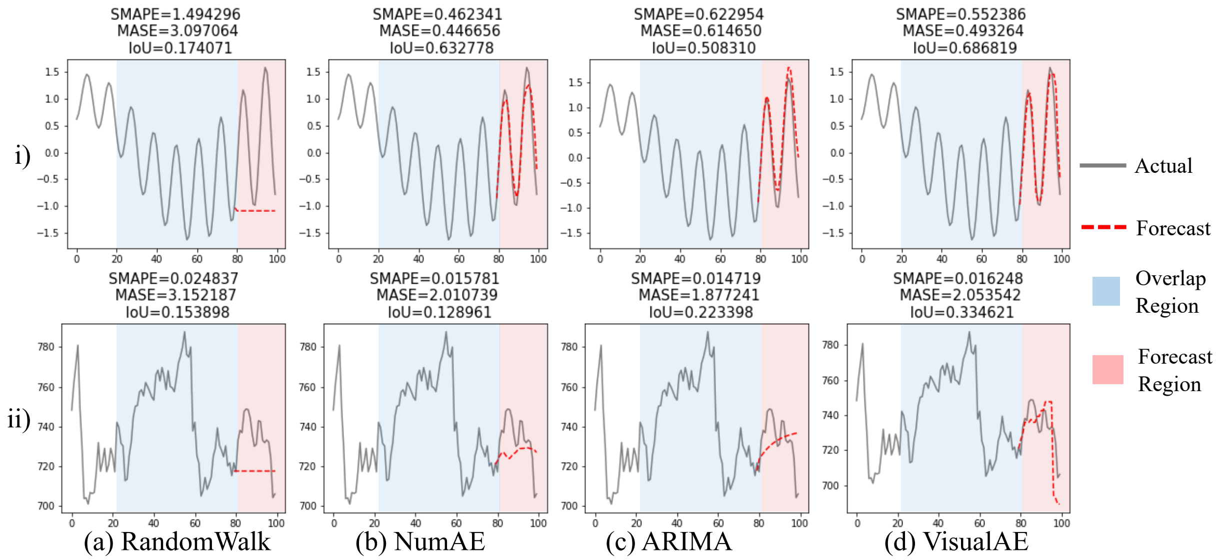

Figure 3: IoU metric better captures visual forecast accuracy compared to traditional numeric metrics SMAPE and MASE.

[2] Pinar Akyazi and Touradj Ebrahimi. 2019. Learning-Based Image Compression using Convolutional Autoencoder and Wavelet Decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops.*

[3] Guillaume Alain and Yoshua Bengio. 2014. What regularized auto-encoders learn from the data-generating distribution. *The Journal of Machine Learning Research* 15, 1 (2014), 3563–3593.

[4] Wei Bao, Jun Yue, and Yulei Rao. 2017. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PloS one* 12, 7 (2017), e0180944.

[5] David Byrd. 2019. Explaining agent-based financial market simulation. *arXiv preprint arXiv:1909.11650* (2019).

[6] Naftali Cohen, Tucker Balch, and Manuela Veloso. 2019. The effect of visual design in image classification. *arXiv preprint arXiv:1907.09567* (2019).

[7] Naftali Cohen, Tucker Balch, and Manuela Veloso. 2019. Trading via image classification. *arXiv preprint arXiv:1907.10046* (2019).

[8] Bairui Du and Paolo Barucca. 2020. Image Processing Tools for Financial Time Series Classification. *arXiv preprint arXiv:2008.06042* (2020).

[9] WA Falcon. 2019. PyTorch Lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning* 3 (2019).

[10] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2001. *The elements of statistical learning.* Springer series in statistics New York.

[11] Andre Gensler, Janosch Henze, Bernhard Sick, and Nils Raabe. 2016. Deep Learning for solar power forecasting—An approach using AutoEncoder and LSTM Neural Networks. In *2016 IEEE international conference on systems, man, and cybernetics (SMC).* IEEE, 002858–002865.

[12] Aurélien Géron. 2019. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* O'Reilly Media.

[13] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. 2000. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *circulation* 101, 23 (2000), e215–e220.

[14] Lovedeep Gondara. 2016. Medical image denoising using convolutional denoising autoencoders. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW).* IEEE, 241–246.

[15] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning.* MIT press Cambridge.

[16] Ferenc Huszár. 2015. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101* (2015).

[17] Rob J Hyndman and George Athanasopoulos. 2018. *Forecasting: principles and practice.* OTexts.

[18] R Krispin. 2019. Hands-On Time Series Analysis with R: Perform time series analysis and forecasting using R.

[19] Xixi Li, Yanfei Kang, and Feng Li. 2020. Forecasting with time series imaging. *Expert Systems with Applications* 160 (2020), 113680.

[20] Yijun Li, Sifei Liu, Jimei Yang, and Ming-Hsuan Yang. 2017. Generative face completion. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 3911–3919.

[21] Spyros Makridakis and Michele Hibon. 2000. The M3-Competition: results, conclusions and implications. *International journal of forecasting* 16, 4 (2000), 451–476.

[22] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. 2020. The M4 Competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting* 36, 1 (2020), 54–74.

[23] Xiao-Jiao Mao, Chunhua Shen, and Yu-Bin Yang. 2016. Image restoration using convolutional auto-encoders with symmetric skip connections. *arXiv preprint arXiv:1606.08921* (2016).

[24] S Mostafa Mousavi, Weiqiang Zhu, William Ellsworth, and Gregory Beroza. 2019. Unsupervised clustering of seismic signals using deep convolutional autoencoders. *IEEE Geoscience and Remote Sensing Letters* 16, 11 (2019), 1693–1697.

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems.* 8026–8037.

[26] Lasse Heje Pedersen. 2019. *Efficiently inefficient: how smart money invests and market prices are determined.* Princeton University Press.

[27] Pablo Romeu, Francisco Zamora-Martínez, Paloma Botella-Rocamora, and Juan Pardo. 2015. Stacked denoising auto-encoders for short-term time series forecasting. In *Artificial Neural Networks.* Springer, 463–486.

[28] Alaa Sagheer and Mostafa Kotb. 2019. Time series forecasting of petroleum production using deep LSTM recurrent networks. *Neurocomputing* 323 (2019), 203–213.

[29] Alaa Sagheer and Mostafa Kotb. 2019. Unsupervised pre-training of a Deep LStM-based Stacked Autoencoder for Multivariate time Series forecasting problems. *Scientific Reports* 9, 1 (2019), 1–16.

[30] Steven E Shreve. 2004. *Stochastic calculus for finance II: Continuous-time models.* Vol. 11. Springer Science & Business Media.

[31] Neda Tavakoli, Sima Siami-Namini, Mahdi Adl Khanghah, Fahimeh Mirza Soltani, and Akbar Siami Namin. 2020. Clustering Time Series Data through Autoencoder-based Deep Learning Models. *arXiv preprint arXiv:2004.07296* (2020).

[32] Zeng Zhen, Tucker Balch, and Manuela Veloso. 2021. Deep Video Prediction for Economic Forecasting. *arXiv preprint arXiv:2102.12061* (2021).