



HAL
open science

A dynamical model for stock forecasting via deep recurrent dictionary learning

Shalini Sharma, Émilie Chouzenoux, Víctor Elvira, Angshul Majumdar

► **To cite this version:**

Shalini Sharma, Émilie Chouzenoux, Víctor Elvira, Angshul Majumdar. A dynamical model for stock forecasting via deep recurrent dictionary learning. 2023. hal-03654152v2

HAL Id: hal-03654152

<https://hal.science/hal-03654152v2>

Preprint submitted on 9 Jan 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

13 A dynamical model for stock forecasting via deep
14 recurrent dictionary learning

15 Shalini Sharma^a, Émilie Chouzenoux^b, Víctor Elvira^c, Angshul Majumdar^d

16 ^a*Indraprastha Institute of Information Technology-Delhi, India*

17 ^b*Inria Saclay, University Paris Saclay, 91190 Gif-sur-Yvette, France*

18 ^c*School of Mathematics, University of Edinburgh, Edinburgh, UK*

19 ^d*Institute of Advanced Intelligence, TCG CREST, Kolkata, India*

20 **Abstract**

State-space models (SSM) and recurrent neural networks (RNN) are widely used approaches for dynamical system modeling. In the case of SSMs, they include explicit modeling of all components, including the noise characterization, and thus allow for interpretability and uncertainty quantification. However, the underlying dynamical model parameters must be specified, and closed-form inference is possible only in a few simple cases. RNNs, on the other hand, can learn, through supervised training, rather complex nonlinearities from the data but lack the aforementioned advantages of SSMs. In this work, we combine the benefits of both approaches by introducing a Gaussian SSM whose state and evolution operators can be learnt from the data. In order to deal with the ill-posedness of this parameter estimation problem, we propose an innovative factorized form of both the state and observation operators reminiscent of deep nonnegative matrix factorization models. An expectation-maximization method combined with a block alternating strategy is introduced to estimate each of the involved positive latent factors, while jointly performing the probabilistic state inference. Our resulting formulation and inference tool is deep recurrent dictionary learning (DRDL). We then specialize DRDL for the problem of stock forecasting by proposing an online training strategy and a probabilistic assessment of the trading decision. Numerical experiments on a problem of stock market data inference show its superiority among several state-of-the-art dynamic modeling tools.

*Corresponding author.

¹
*Email addresses: shalinis@iiitd.ac.in (Shalini Sharma),
emilie.chouzenoux@inria.fr (Émilie Chouzenoux), victor.elvira@ed.ac.uk (Víctor Elvira), angshul@iiitd.ac.in (Angshul Majumdar)*

21 *Keywords:* Time series analysis; State space models; Deep nonnegative
22 matrix factorization; Kalman filtering; Bayesian smoothing; EM algorithm;
23 Stock forecasting; Stock trading.

24 **1. Introduction**

25 Modeling dynamical systems has been a topic of interest to signal processing,
26 machine learning and control engineering researchers for more than five decades.
27 Applications range in areas as diverse as financial market analysis to electric
28 demand forecasting. We propose a new dynamical recurrent modeling technique
29 that combines the advantages of state-of-the-art deep learning tools with those
30 of traditional state-space models. The proposed tool is then particularized to
31 processing stock market time series. In the following, we review the literature
32 of dynamic modeling around this particular application and we describe the
33 contributions of the paper.

34 *1.1. State-of-the-art review*

35 Modeling the stock market is a well-known challenging problem Yang &
36 Wu (2006). The difficulty lies in the non-stationary and nonlinearity of the
37 underlying dynamical process. Moreover, financial markets are not only in-
38 fluenced by consumer behavior but also by a myriad of external factors like
39 natural disasters, administrative policies, political decisions, international rela-
40 tions, etc., to name a few. Therefore developing reliable algorithmic models for
41 stock trading still remains a challenging yet interesting topic from the point of
42 view of both finance and machine learning/signal processing Fama (2021); Shah
43 et al. (2019). Auto-regressive moving average (ARMA) models have been used
44 to model stock market Zhao-yang (2010); Atsalakis & Valavanis (2010); Moon
45 et al. (2021); Nowicka-Zagrajek & Weron (2002). ARMA assumes the stochastic
46 process to be stationary; this turns out to be too simplistic and consequently
47 unrealistic for the stock market. This limitation was partially overcome by au-
48 toregressive integrated moving average (ARIMA) Ariyo et al. (2014) models

49 (also referred as Box-Jenkins model). ARIMA has been used in the past for
50 stock forecasting and trading Devi et al. (2013), Petrica et al. (2016). However,
51 Box-Jenkins/ARIMA methods could not model non-smooth variations in time
52 series Makridakis & Hibon (1997); O'Donovan (1983). ARIMA with regressors
53 were introduced to overcome the limitations, leading to ARIMAX Chadsuthi
54 et al. (2012); Ababio (2012). Unfortunately, ARIMAX introduced other prob-
55 lems such as over/under-fitting because of the handling of extra predictors and
56 variables.

57 SSM is another powerful approach for modeling and analysing time-series
58 Särkkä (2013); Newman et al. (2023). Many studies used SSMs for stock fore-
59 casting, and analysis Östermark (1991); Saini et al. (2014); Elvira & Chouzenoux
60 (2022). The celebrated Kalman filter is a solution to the inference of a linear
61 SSM where the noise is assumed to be Gaussian Kalman (1960). The literature
62 illustrates its minimal use in stock forecasting Choudhry & Wu (2009) but found
63 its application in other financial analyses, see for example Wells (2013). To over-
64 come the restrictive linearity assumption, extended Kalman filter (EKF) Ljung
65 (1979), unscented Kalman filter (UKF) Wan & Van Der Merwe (2000) were in-
66 troduced. Particle filters Djuric et al. (2003); Doucet & Johansen (2009); Elvira
67 et al. (2019); Ntemi & Kotropoulos (2021) further relaxed the Gaussianity as-
68 sumption. The advantage of the SSM approach is that it can model uncertainty
69 in the estimate Doucet & Johansen (2009); Bach & Jordan (2004). Uncertainty
70 is crucial for financial markets since it gives a measure of the associated risk
71 Rigotti & Shannon (2005).

72 The main drawback of the aforesaid signal processing-based forecasting ap-
73 proaches is that they need the model's specification. Unfortunately, specifying
74 an underlying model for the stock market is difficult, if not impossible. Several
75 works in the literature have thus investigated the learning of model parameters
76 in SSMs. In the case of linear-Gaussian state-space models (LG-SSMs), see for
77 instance the methods in Sharma et al. (2020); Shumway & Stoffer (1982); Khan
78 & Dutt (2007), and (Särkkä, 2013, Chapter 12). All these works consider the
79 observation and state operators to be unknown and estimated from data using

80 expectation-maximization (EM) methods. However, the aforementioned works
81 can only consider linear models and do not account for any prior knowledge
82 on the involved operators. Inferring model parameters for non-linear SSMs has
83 been explored in more generic algorithms, e.g., particle MCMC methods An-
84 drieu et al. (2010), SMC² Chopin et al. (2013), and nested PFs Crisan & Miguez
85 (2018). In all these cases, the inference is costly, as they use Monte-Carlo sam-
86 pling methods and thus do not generally scale well. The problem of scalability
87 in SSM model inference has been mildly explored. Let us mention our two re-
88 cent works Chouzenoux & Elvira (2020); Sharma et al. (2021), both focusing
89 on LG-SSMs. In Chouzenoux & Elvira (2020), a sparsity prior is introduced on
90 the linear matrices to infer, providing an interpretable and compressible model.
91 Though this method is promising, it does not allow explicit control of the final
92 dimension of the model easily, and as such, still requires an increased compu-
93 tational time at inference. In Sharma et al. (2021), we proposed an online (still
94 EM-based) estimation approach in the context of stock market time series pro-
95 cessing. The online processing allows a reduced complexity and memory burden
96 while being beneficial to capturing non-linear phenomena in such volatile time
97 series. However, the parametric estimation step lacked of robustness and lack
98 of sufficient imposed structure on the estimated factors.

99 Neural network (NN) models represent another family of approaches for time
100 series modeling. By construction, these methods excel when the model spec-
101 ification is missing, as they implicitly learn the model from the data through
102 the training phase. In particular, the approximation capability of recurrent
103 neural networks (RNNs) for dynamical systems allows to learn the underlying
104 phenomena given enough training data Gonzalez-Olvera & Tang (2010); Won
105 et al. (2010); Yin et al. (2022). RNN and its subsequent versions are used in
106 several studies for stock price forecasting Saad et al. (1998); Tino et al. (2001).
107 Deeper neural network architectures are known to yield better results than their
108 shallow counterparts Bengio et al. (2007); Shao et al. (2014). They are engi-
109 neered to approximate highly non-linear function in high-dimensional spaces
110 and are supposed to be more suitable for challenging problems Cheridito et al.

111 (2021); Bianchini & Scarselli (2014). 1-D CNN performs better when compared
112 to LSTM and RNN owing to their ease of training. There are studies, such as
113 Sezer & Ozbayoglu (2018), that use them as financial forecasting. However, 1-D
114 CNNs cannot process streaming data. Hence some works have recently proposed
115 the combination of RNN with 1-D CNN in order to model time-series signals
116 Long et al. (2019). It must be noted that deep neural networks are computa-
117 tionally intensive Abbe & Sandon (2018). Furthermore, deep neural networks
118 only provide data estimates for each time step and do not provide uncertainty
119 quantification, while such information would be necessary for stock forecasting
120 to assess risks. Therefore, recent works have been dedicated to combine proba-
121 bilistic forecasting and deep learning techniques, so as to predict the probability
122 distribution of future events in the time series given its past/historical recordings
123 Salinas et al. (2020); Jiang (2021). Deep factor model based on dropout-based
124 heuristic and complex semantics have also been considered in Chauhan et al.
125 (2020). These techniques provide probability distributions as outputs, thanks
126 to specific learning strategies inherited from the Bayesian NN literature. How-
127 ever, these works, up to our knowledge, do not mention any explicit strategy
128 to estimate uncertainty/confidence score on future predictions/decisions that
129 would help to assess their reliability. Standard (non deep) machine learning
130 models have also been combined to statistical time series modeling tools. For
131 instance, the work Pai & Lin (2005) uses ARIMA and support vector machine.
132 Another work uses a combination of ARIMA and random forests for the same
133 task Kumar & Thenmozhi (2014). Finally, several works, such as Ding et al.
134 (2015); Chong et al. (2017); Fischer & Krauss (2018), use information mined
135 from news articles and blogs via natural language processing for stock forecast-
136 ing Cheng et al. (2022); Li et al. (2023). Strictly speaking, this is not artificial
137 intelligence since these are dependent on human cognizance.

138 *1.2. Contributions compared to existing literature*

139 As a summary, SSMs are valuable tools for probabilistic time series model-
140 ing and inference. But there is a crucial need for new strategies to cope with

141 the curse of dimensionality in learning SSM model parameters. In this work,
 142 we propose to impose a structured prior on the observation/state operators in-
 143 volved in an LG-SSM. We introduce deep nonnegative matrix factorized (deep
 144 NMF) models for both operators. Deep NMF De Handschutter et al. (2021) is
 145 a generalized form of NMF Cichocki et al. (2009) that models latent representa-
 146 tions from complex data through a product of a (usually small) number of linear
 147 operators (called latent factors) satisfying positivity constraints. Deep NMF has
 148 been employed with success on various unsupervised machine learning tasks Yu
 149 et al. (2015); Trigeorgis et al. (2016); Xue et al. (2017); Chen et al. (2021);
 150 Flenner & Hunter (2017). When embedded into an NN structure, it leads to
 151 the so-called deep ReLU networks Liu & Liang (2021); Daubechies et al. (2022);
 152 Chen et al. (2019). Deep NMF shares connections with the recently introduced
 153 deep dictionary learning (deep DL) Tariyal et al. (2016); Mahdizadehghadam
 154 et al. (2019), the main difference being in the priors imposed in the latent fac-
 155 tors (positivity, in the case of deep NMF, low-rank/sparsity in the case of deep
 156 DL). In our work, deep NMF is neither used for unsupervised representation
 157 learning nor in an NN framework. In contrast, it is embedded into a Gaussian
 158 SSM to model, allowing to track and predict complex latent phenomena in time
 159 series. A novel algorithm is proposed, that learns the positive latent factors
 160 jointly with the probabilistic state inferential task induced by our SSM. We call
 161 this modelling and inference tool *Deep Recurrent Dictionary Learning (DRDL)*.
 162 We further specialize this tool, to make it practically efficient in the context of
 163 large and volatile time series arising in stock market data. In particular, we
 164 perused the online training strategy we previously introduced in Sharma et al.
 165 (2021).

166 **Contributions in a nutshell.** In this work, we:

- 167 • Introduce an LG-SSM model involving deep positive latent factors;
- 168 • Propose a new EM-based inference method to jointly perform the time
 169 series prediction task and the deep linear positive factors estimation;
- 170 • Devise efficient implementation strategies for practical use of the method

171 in the context of stock market time series analysis;

- 172 • Investigate through experiments and benchmark comparisons on real fi-
173 nancial data the performance of the novel DRDL approach.

174 1.3. Paper organization

175 The rest of the paper is organized as follows. The proposed DRDL model
176 and its inference is presented in the following Section 2. The practical imple-
177 mentation of DRDL in the context of stock market data analysis is discussed in
178 Section 3. The experimental results and their analysis are described in Section
179 4. The conclusions of this work are discussed in section 5.

180 1.4. Paper organization

181 The rest of the paper is organized as follows. The proposed DRDL model
182 and its inference is presented in the following Section 2. The practical imple-
183 mentation of DRDL in the context of stock market data analysis is discussed in
184 Section 3. The experimental results and their analysis are described in Section
185 4. The conclusions of this work are discussed in section 5.

186 2. Proposed Work

187 2.1. Considered model

188 Let us consider an observed sequence $(\mathbf{x}_k)_{1 \leq k \leq K}$ of vectors of size $N_x \geq 1$.
189 We aim to estimate $(\mathbf{z}_k)_{1 \leq k \leq K}$, a sequence of unknown hidden/latent vectors
190 of size $N_z \geq 1$. The DRDL approach relies on the following re-parametrized
191 LG-SSM:

192 For every $k \in \{1, \dots, K\}$,

$$\begin{cases} \mathbf{z}_k &= \mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2 \mathbf{z}_{k-1} + \mathbf{v}_{1,k}, \\ \mathbf{x}_k &= \mathbf{H}_0 \mathbf{H}_1 \mathbf{H}_2 \mathbf{z}_k + \mathbf{v}_{2,k}. \end{cases} \quad (1)$$

193 where $\mathbf{D}_0 \in [0, +\infty)^{N_z \times N_z}$, $\mathbf{D}_1 \in [0, +\infty)^{N_z \times N_z}$, and $\mathbf{D}_2 \in [0, +\infty)^{N_z \times N_z}$
194 are three positive-valued linear factors leading to a multi-linear state oper-
195 ator $\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2$. Similarly, $\mathbf{H}_0 \in [0, +\infty)^{N_x \times N_z}$, $\mathbf{H}_1 \in [0, +\infty)^{N_x \times N_z}$, $\mathbf{H}_2 \in$

196 $[0, +\infty)^{N_z \times N_z}$ are three positive-valued linear factors yielding the multi-linear
 197 observation model $\mathbf{H}_0 \mathbf{H}_1 \mathbf{H}_2$.¹ The process noise $(\mathbf{v}_{1,k})_{1 \leq k \leq K}$ is assumed to
 198 have a Gaussian distribution with zero-mean and symmetric definite positive
 199 covariance matrix $\mathbf{Q} \in \mathbb{R}^{N_z \times N_z}$. The observation noise $(\mathbf{v}_{2,k})_{1 \leq k \leq K}$, is also as-
 200 sumed zero-mean Gaussian with symmetric definite positive covariance matrix
 201 $\mathbf{R} \in \mathbb{R}^{N_x \times N_x}$. We consider $\mathbf{z}_0 \sim \mathcal{N}(\bar{\mathbf{z}}_0, \mathbf{P}_0)$ as the initial state, with $\bar{\mathbf{z}}_0 \in \mathbb{R}$
 202 and $\mathbf{P}_0 \in \mathbb{R}^{N_z \times N_z}$ defined as definite symmetric positive matrix. The model
 203 in Eq. (1) can be interpreted as a multi-linear Gaussian model involving a se-
 204 quence of K hidden states represented by $(\mathbf{z}_k)_{1 \leq k \leq K}$. As discussed earlier, clas-
 205 sical inference approaches for SSM require specifying every model parameters.
 206 In the model above, this would mean setting the positive latent factor matri-
 207 ces matrices $\{\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2\}$ involved in both state and observation
 208 models. In practical applications such as stock market analysis, these param-
 209 eters are unknown and must be learnt from the observed data. The objective
 210 is thus to provide a point-wise estimate of the positive latent factor matrices
 211 $\{\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2\}$ and a probabilistic estimate of sequence $(\mathbf{z}_k)_{1 \leq k \leq K}$,
 212 given the observed sequence $(\mathbf{x}_k)_{1 \leq k \leq K}$. This can be seen as solving jointly (i)
 213 two deep NMF problems, (ii) a filtering/smoothing problem.

214 2.2. Discussion on the model

215 We now discuss the main characteristics of the proposed DRDL method.
 216 The model is mathematically described in Eq. (1) and displayed in Fig. 1. The
 217 top equation describes the hidden state evolution, assuming Markovianity be-
 218 tween two consecutive hidden states. The second equation links the hidden and
 219 observed states. A first interesting aspect, inherited from the SSM paradigm,
 220 is that two Gaussian noise terms are explicitly introduced in DRDL to cope
 221 with model uncertainty, which is in contrast with most deep learning models for

¹Throughout the paper, we consider three-terms factorizations, for the sake of readability.
 The 3-layers modeling and inference methodology has the great advantage of being generic
 enough to be straightforwardly extended to any number, greater or equals to one, of factors.

222 time series processing (e.g., LSTM). A second novel feature of (1) lies in using
 223 deep NMF models instead of generic matrices (in the linear case) or functions
 224 (in the non-linear case), as it is usually the case in SSMs Andrieu et al. (2010);
 225 Chopin et al. (2013); Crisan & Miguez (2018), taking advantage on the acknowl-
 226 edged representation power of deep NMF De Handschutter et al. (2021). One
 227 important benefit of the proposed approach w.r.t. most existing methods in the
 228 literature is that we avoid Monte Carlo simulation or complex optimization pro-
 229 cedure, which is known to suffer more severely the curse of dimensionality. In
 230 our method, each latent factor can be understood as representations in abstract
 231 spaces of the phenomena occurring between both pairs of variables. Third, in
 232 contrast with the typical usage of deep NMF in machine learning, relying on
 233 backpropagation for their model training Chen et al. (2021); Flenner & Hunter
 234 (2017), DRDL model allows the construction of an handcrafted training strat-
 235 egy (see the next section), which benefits from a low computational cost, sound
 236 optimality guarantees (in terms of Bayesian estimator), and enables uncertainty
 quantification.

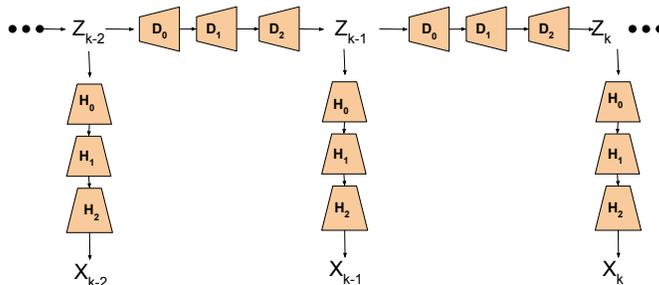


Figure 1: Schematic Diagram for Deep Recurrent Dictionary Learning

237

238 2.3. DRDL inference algorithm

239 Using SSM models for time series processing (e.g., for a prediction task)
 240 amounts to solving the so-called smoothing/filtering problem, i.e., the proba-
 241 bilistic estimation of the hidden state $(\mathbf{z}_k)_{1 \leq k \leq K}$. In our context, as the deep
 242 NMF factors $\{\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2\}$ involved in the construction of the state

243 transition and the observation transition models are most often unknown, we
 244 must also infer them from the observed data, jointly with the hidden states
 245 (through the aforementioned filtering/smoothing procedure). To do so, we propose
 246 an expectation-maximization (EM) approach (see (Särkkä, 2013, chap.12)
 247 and Shumway & Stoffer (1982)). The EM method alternates iteratively between
 248 the probabilistic inference of the state $(\mathbf{z}_k)_{1 \leq k \leq K}$, while the positive
 249 factors $\{\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2\}$ are fixed (E-step), and the update of these
 250 factors, assuming fixed state (M-step). More precisely, the E-step consists in
 251 fixing the linear operators obtained in the previous M-step and applying the
 252 classical Kalman/RTS recursions, for obtaining the filtered/smooth distributions
 253 $p(\mathbf{z}_k | \mathbf{x}_{1:k})$ and $p(\mathbf{z}_k, \mathbf{x}_{1:K})$, respectively. Then, the M-step updates the
 254 operators $\{\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2\}$ by maximizing an upper bound of:

$$\begin{aligned}
 \varphi_K(\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2) \\
 = \log p(\mathbf{x}_{1:K} | \mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2). \quad (2)
 \end{aligned}$$

255 We explicit hereafter the construction of the $i+1$ -th EM update, given estimates
 256 from the previous iteration i .

257 2.3.1. E-step: Kalman/RTS inference

258 At this step, we considered the factors $\mathbf{D}_0^{[i]}, \mathbf{D}_1^{[i]}, \mathbf{D}_2^{[i]}, \mathbf{H}_0^{[i]}, \mathbf{H}_1^{[i]}, \mathbf{H}_2^{[i]}$ to be
 259 fixed (either from the previous M-step or from the initialization at the first
 260 iteration), and the goal is the probabilistic estimation of the latent states. As we
 261 aforementioned, (1) is a multi-linear Gaussian model whose observation operator
 262 $\mathbf{H}_0 \mathbf{H}_1 \mathbf{H}_2$, evolution/state operator $\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2$, and hidden state $(\mathbf{z}_k)_{1 \leq k \leq K}$ must
 263 be estimated. For each $k \in \{1, \dots, K\}$, the probabilistic estimate of the latter
 264 conditioned to all available data up to time k , is provided by the Kalman filter
 265 through the following Gaussian filtering distribution:

$$p(\mathbf{z}_k | \mathbf{x}_{1:k}) = \mathcal{N}(\mathbf{z}_k; \bar{\mathbf{z}}_k, \mathbf{P}_k). \quad (3)$$

266 For every k , the mean $\bar{\mathbf{z}}_k$ and the covariance \mathbf{P}_k are given by the Kalman
 267 iterations:

268 For $k = 1, \dots, K$:

269 *Predict state:*

$$\begin{cases} \mathbf{z}_{k|k-1} &= \mathbf{D}_0^{[i]} \mathbf{D}_1^{[i]} \mathbf{D}_2^{[i]} \bar{\mathbf{z}}_{k-1}, \\ \mathbf{P}_{k|k-1} &= \mathbf{D}_0^{[i]} \mathbf{D}_1^{[i]} \mathbf{D}_2^{[i]} \mathbf{P}_{k-1} (\mathbf{D}_0^{[i]} \mathbf{D}_1^{[i]} \mathbf{D}_2^{[i]})^\top + \mathbf{Q}. \end{cases} \quad (4)$$

270 *Update state:*

$$\begin{cases} \mathbf{y}_k &= \mathbf{x}_k - \mathbf{H}_0^{[i]} \mathbf{H}_1^{[i]} \mathbf{H}_2^{[i]} \mathbf{z}_{k|k-1}, \\ \mathbf{S}_k &= \mathbf{H}_0^{[i]} \mathbf{H}_1^{[i]} \mathbf{H}_2^{[i]} \mathbf{P}_{k|k-1} (\mathbf{H}_0^{[i]} \mathbf{H}_1^{[i]} \mathbf{H}_2^{[i]})^\top + \mathbf{R}, \\ \mathbf{K}_k &= \mathbf{P}_{k|k-1} (\mathbf{H}_0^{[i]} \mathbf{H}_1^{[i]} \mathbf{H}_2^{[i]})^\top \mathbf{S}_k^{-1}, \\ \bar{\mathbf{z}}_k &= \mathbf{z}_{k|k-1} + \mathbf{K}_k \mathbf{y}_k, \\ \mathbf{P}_k &= \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^\top. \end{cases} \quad (5)$$

271 Hereabove, \mathbf{y}_k represents the measurement pre-fit residual, \mathbf{S}_k represents the
 272 pre-fit covariance, \mathbf{K}_k represents Kalman gain, $\bar{\mathbf{z}}_k$ represents the updated (a
 273 posteriori) state estimate, \mathbf{P}_k represents the updated (a posteriori) covariance
 274 estimate. The backward recursion from the RTS smoother allow to build the
 275 smoothing distribution $p(\mathbf{z}_k | \mathbf{x}_{1:K})$.

276 For $k = K, \dots, 1$

277 *Backward Recursion:*

$$\begin{cases} \mathbf{z}_{k+1}^- &= \mathbf{D}_0^{[i]} \mathbf{D}_1^{[i]} \mathbf{D}_2^{[i]} \bar{\mathbf{z}}_k, \\ \mathbf{P}_{k+1}^- &= \mathbf{D}_0^{[i]} \mathbf{D}_1^{[i]} \mathbf{D}_2^{[i]} \mathbf{P}_k (\mathbf{D}_0^{[i]} \mathbf{D}_1^{[i]} \mathbf{D}_2^{[i]})^\top + \mathbf{Q}, \\ \mathbf{G}_k &= \mathbf{P}_k (\mathbf{D}_0^{[i]} \mathbf{D}_1^{[i]} \mathbf{D}_2^{[i]})^\top [\mathbf{P}_{k+1}^-]^{-1}, \\ \mathbf{z}_k^s &= \bar{\mathbf{z}}_k + \mathbf{G}_k [\mathbf{z}_{k+1}^s - \mathbf{z}_{k+1}^-], \\ \mathbf{P}_k^s &= \mathbf{P}_k - \mathbf{G}_k [\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-] \mathbf{G}_k^\top. \end{cases} \quad (6)$$

278 Consequently, for every time step $k \in \{1, \dots, K\}$, the RTS smoother provides:

$$p(\mathbf{z}_k | \mathbf{x}_{1:K}) = \mathcal{N}(\mathbf{z}_k; \mathbf{z}_k^s, \mathbf{P}_k^s). \quad (7)$$

279 *2.3.2. M-step: Evolution operators update*

280 The M-step performs an optimization step to increase the likelihood of the
 281 positive latent factors $\{\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2\}$, given the smoothing distribu-
 282 tion obtained in the E-step. It proceeds by building the upper-bound:

$$\begin{aligned} \varphi_k(\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2) \\ \geq \mathcal{Q}(\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2; \Theta^{[i]}). \end{aligned} \quad (8)$$

283 Hereabove, $\Theta^{[i]} = \{\Sigma^{[i]}, \Phi^{[i]}, \mathbf{B}^{[i]}, \mathbf{C}^{[i]}, \Delta^{[i]}\}$ gathers five quantities defined from
 284 the outputs of the E-step described in Sec. 2.3.1):

$$\begin{aligned} \mathcal{Q}(\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2; \Theta^{[i]}) = \\ - \frac{K}{2} \text{tr} \left(\mathbf{Q}^{-1} \Sigma^{[i]} - \mathbf{C}^{[i]} (\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2)^\top - \mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2 (\mathbf{C}^{[i]})^\top \right. \\ \left. + \mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2 \Phi^{[i]} (\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2)^\top \right) \\ - \frac{K}{2} \text{tr} \left(\mathbf{R}^{-1} \Delta^{[i]} - \mathbf{B}^{[i]} (\mathbf{H}_0 \mathbf{H}_1 \mathbf{H}_2)^\top - \mathbf{H}_0 \mathbf{H}_1 \mathbf{H}_2 (\mathbf{B}^{[i]})^\top \right. \\ \left. + \mathbf{H}_0 \mathbf{H}_1 \mathbf{H}_2 \Sigma^{[i]} (\mathbf{H}_0 \mathbf{H}_1 \mathbf{H}_2)^\top \right), \end{aligned} \quad (9)$$

285 with:

$$\begin{aligned} \Sigma^{[i]} &= \frac{1}{K} \sum_{k=1}^K \mathbf{P}_k^s + \mathbf{z}_k^s (\mathbf{z}_k^s)^\top, \\ \Phi^{[i]} &= \frac{1}{K} \sum_{k=1}^K \mathbf{P}_{k-1}^s + \mathbf{z}_{k-1}^s (\mathbf{z}_{k-1}^s)^\top, \\ \mathbf{B}^{[i]} &= \frac{1}{K} \sum_{k=1}^K \mathbf{x}_k (\mathbf{z}_k^s)^\top, \\ \mathbf{C}^{[i]} &= \frac{1}{K} \sum_{k=1}^K \mathbf{P}_k^s \mathbf{G}_{k-1}^\top + \mathbf{z}_k^s (\mathbf{z}_{k-1}^s)^\top, \\ \Delta^{[i]} &= \frac{1}{K} \sum_{k=1}^K \mathbf{x}_k \mathbf{x}_k^\top. \end{aligned} \quad (10)$$

286 The updates $\{\mathbf{D}_0^{[i+1]}, \mathbf{D}_1^{[i+1]}, \mathbf{D}_2^{[i+1]}, \mathbf{H}_0^{[i+1]}, \mathbf{H}_1^{[i+1]}, \mathbf{H}_2^{[i+1]}\}$ given the knowledge
 287 of $\Theta^{[i]}$, amounts to maximizing $\mathcal{Q}(\cdot; \Theta^{[i]})$ under positivity constraints on the
 288 factors. In contrast with the linear unconstrained model case studied in (Särkkä,
 289 2013, Chapter 12), the maximization problem here does not have a closed-form
 290 solution. It is highly non-convex due to the multi-linearity of our model. Luckily,
 291 it happens to be convex with respect to each of the factors. We thus propose

292 to resort to the following alternating maximization step:

$$\begin{aligned}
\mathbf{D}_0^{[i+1]} &= \operatorname{argmax}_{\mathbf{D}_0 \geq 0} \mathcal{Q}(\mathbf{D}_0, \mathbf{D}_1^{[i]}, \mathbf{D}_2^{[i]}, \mathbf{H}_0^{[i]}, \mathbf{H}_1^{[i]}, \mathbf{H}_2^{[i]}; \Theta^{[i]}) \\
\mathbf{D}_1^{[i+1]} &= \operatorname{argmax}_{\mathbf{D}_1 \geq 0} \mathcal{Q}(\mathbf{D}_0^{[i+1]}, \mathbf{D}_1, \mathbf{D}_2^{[i]}, \mathbf{H}_0^{[i]}, \mathbf{H}_1^{[i]}, \mathbf{H}_2^{[i]}; \Theta^{[i]}) \\
\mathbf{D}_2^{[i+1]} &= \operatorname{argmax}_{\mathbf{D}_2 \geq 0} \mathcal{Q}(\mathbf{D}_0^{[i+1]}, \mathbf{D}_1^{[i+1]}, \mathbf{D}_2, \mathbf{H}_0^{[i]}, \mathbf{H}_1^{[i]}, \mathbf{H}_2^{[i]}; \Theta^{[i]}) \\
\mathbf{H}_0^{[i+1]} &= \operatorname{argmax}_{\mathbf{H}_0 \geq 0} \mathcal{Q}(\mathbf{D}_0^{[i+1]}, \mathbf{D}_1^{[i+1]}, \mathbf{D}_2^{[i+1]}, \mathbf{H}_0, \mathbf{H}_1^{[i]}, \mathbf{H}_2^{[i]}; \Theta^{[i]}) \\
\mathbf{H}_1^{[i+1]} &= \operatorname{argmax}_{\mathbf{H}_1 \geq 0} \mathcal{Q}(\mathbf{D}_0^{[i+1]}, \mathbf{D}_1^{[i+1]}, \mathbf{D}_2^{[i+1]}, \mathbf{H}_0^{[i+1]}, \mathbf{H}_1, \mathbf{H}_2^{[i]}; \Theta^{[i]}) \\
\mathbf{H}_2^{[i+1]} &= \operatorname{argmax}_{\mathbf{H}_2 \geq 0} \mathcal{Q}(\mathbf{D}_0^{[i+1]}, \mathbf{D}_1^{[i+1]}, \mathbf{D}_2^{[i+1]}, \mathbf{H}_0^{[i+1]}, \mathbf{H}_1^{[i+1]}, \mathbf{H}_2; \Theta^{[i]})
\end{aligned}$$

293 This approach ensures by construction the following inequality:

$$\begin{aligned}
\mathcal{Q}(\mathbf{D}_0^{[i+1]}, \mathbf{D}_1^{[i+1]}, \mathbf{D}_2^{[i+1]}, \mathbf{H}_0^{[i+1]}, \mathbf{H}_1^{[i+1]}, \mathbf{H}_2^{[i+1]}; \Theta^{[i]}) \\
\geq \mathcal{Q}(\mathbf{D}_0^{[i]}, \mathbf{D}_1^{[i]}, \mathbf{D}_2^{[i]}, \mathbf{H}_0^{[i]}, \mathbf{H}_1^{[i]}, \mathbf{H}_2^{[i]}; \Theta^{[i]}), \quad (11)
\end{aligned}$$

294 which is key to guarantee the convergence properties for the EM iteration. In-
295 deed, the proposed updates yield an increase of the lower bound of the marginal
296 likelihood, so as a consequence, an increase of the marginal log-likelihood itself.
297 The overall procedure is thus guaranteed to yield a monotonic increase of the
298 marginal log-likelihood function φ_K and classical results about majorization-
299 minimization methods allow to ensure convergence guarantees to a stationary
300 point of φ_K Jacobson & Fessler (2007). The six sub-problems are quadratic pro-
301 gramming (convex) problems and can be solved through several available solvers.
302 We decided to use the simple and fast alternating least squares approach Ci-
303 chocki et al. (2009), reminiscent from the literature of deep nonnegative matrix
304 factorization Chen et al. (2021), and the deep ReLU neural networks models
305 Daubechies et al. (2022), both showing a satisfactory behavior in preliminary
306 experiments. We start by computing each subproblem solution ignoring the pos-
307 itivity constraints, and then capped the negative entries of the obtained factors.
308 This yields the following analytic updates:

$$\begin{aligned}
\mathbf{D}_0^{[i+1]} &= \text{ReLu} \left(\mathbf{C}^{[i]} (\mathbf{D}_2^{[i]})^\top (\mathbf{D}_1^{[i]})^\top (\mathbf{D}_1^{[i]} \mathbf{D}_2^{[i]} \boldsymbol{\Phi}^{[i]} (\mathbf{D}_2^{[i]})^\top (\mathbf{D}_1^{[i]})^\top)^\dagger \right), \\
\mathbf{D}_1^{[i+1]} &= \text{ReLu} \left(((\mathbf{D}_0^{[i+1]})^\top \mathbf{Q}^{-1} \mathbf{D}_0^{[i+1]})^\dagger (\mathbf{D}_0^{[i+1]})^\top \mathbf{Q}^{-1} \mathbf{C}^{[i]} (\mathbf{D}_2^{[i]})^\top \right. \\
&\quad \left. \times (\mathbf{D}_2^{[i]} \boldsymbol{\Phi}^{[i]} (\mathbf{D}_2^{[i]})^\top)^\dagger \right), \\
\mathbf{D}_2^{[i+1]} &= \text{ReLu} \left(((\mathbf{D}_1^{[i+1]})^\top (\mathbf{D}_0^{[i+1]})^\top \mathbf{Q}^{-1} \mathbf{D}_0^{[i+1]} \mathbf{D}_1^{[i+1]})^\dagger (\mathbf{D}_1^{[i+1]})^\top \right. \\
&\quad \left. \times (\mathbf{D}_0^{[i+1]})^\top \mathbf{Q}^{-1} \mathbf{C}^{[i]} (\boldsymbol{\Phi}^{[i]})^{-1} \right), \\
\mathbf{H}_0^{[i+1]} &= \text{ReLu} \left(\mathbf{B}^{[i]} (\mathbf{H}_2^{[i]})^\top (\mathbf{H}_1^{[i]})^\top (\mathbf{H}_1^{[i]} \mathbf{H}_2^{[i]} \boldsymbol{\Sigma}^{[i]} (\mathbf{H}_2^{[i]})^\top (\mathbf{H}_1^{[i]})^\top)^\dagger \right), \\
\mathbf{H}_1^{[i+1]} &= \text{ReLu} \left(((\mathbf{H}_0^{[i+1]})^\top \mathbf{R}^{-1} \mathbf{H}_0^{[i+1]})^\dagger (\mathbf{H}_0^{[i+1]})^\top \mathbf{R}^{-1} \mathbf{B}^{[i]} (\mathbf{H}_2^{[i]})^\top \right. \\
&\quad \left. \times (\mathbf{H}_2^{[i]} \boldsymbol{\Sigma}^{[i]} (\mathbf{H}_2^{[i]})^\top)^\dagger \right), \\
\mathbf{H}_2^{[i+1]} &= \text{ReLu} \left(((\mathbf{H}_1^{[i+1]})^\top (\mathbf{H}_0^{[i+1]})^\top \mathbf{R}^{-1} \mathbf{H}_0^{[i+1]} \mathbf{H}_1^{[i+1]})^\dagger \right. \\
&\quad \left. \times (\mathbf{H}_1^{[i+1]})^\top (\mathbf{H}_0^{[i+1]})^\top \mathbf{R}^{-1} \mathbf{B}^{[i]} (\boldsymbol{\Sigma}^{[i]})^{-1} \right). \tag{12}
\end{aligned}$$

310

311 Hereabove, $(\cdot)^\dagger$ denotes the pseudo-inverse operator. Moreover, $\text{ReLu}(\cdot)$
312 states for the rectified linear unit function, that projects each entry of its input
313 to the positive orthant.

314 2.4. The DRDL algorithm summarized

315 We summarize in Alg. 1 the DRDL algorithm, for the probabilistic inference
316 of the sequence of hidden state $(\mathbf{z}_k)_{1 \leq k \leq K}$, jointly with the point-wise estimation
317 of the latent factors $\{\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2\}$, assuming the data follows the
318 DRDL model (1). In practice, DRDL algorithm is ran for a maximum number
319 of iterations i_{\max} , set so as to reach stabilisation of the latent factors.

320

321 3. Application to Stock Trading

322 We now particularize the DRDL inference algorithm to the stock trading
323 applications. In particular, we address the forecasting/trading tasks given a set
324 of K daily (i.e., k is a day index) observations of stock market data.

Algorithm 1. DRDL (3 layers) inference algorithm.

Inputs. Prior parameters $(\bar{\mathbf{z}}_0, \mathbf{P}_0)$; model noise covariance matrices \mathbf{Q}, \mathbf{R} ; set of observations $\{\mathbf{x}_k\}_{1 \leq k \leq K}$.

Initialization. Set positive latent factors

$$\{\mathbf{D}_0^{(0)}, \mathbf{D}_1^{(0)}, \mathbf{D}_2^{(0)}, \mathbf{H}_0^{(0)}, \mathbf{H}_1^{(0)}, \mathbf{H}_2^{(0)}\}.$$

Recursive step. For $i = 0, 1, \dots, i_{\max}$:

(E step) Run the Kalman filter (4)-(5) and RTS smoother (6) using latent factors $\{\mathbf{D}_0^{(i)}, \mathbf{D}_1^{(i)}, \mathbf{D}_2^{(i)}, \mathbf{H}_0^{(i)}, \mathbf{H}_1^{(i)}, \mathbf{H}_2^{(i)}\}$.

Calculate $(\boldsymbol{\Sigma}^{(i)}, \boldsymbol{\Phi}^{(i)}, \mathbf{B}^{(i)}, \mathbf{C}^{(i)}, \boldsymbol{\Delta}^{(i)})$ using (10).

(M step) Compute $\{\mathbf{D}_0^{(i+1)}, \mathbf{D}_1^{(i+1)}, \mathbf{D}_2^{(i+1)}, \mathbf{H}_0^{(i+1)}, \mathbf{H}_1^{(i+1)}, \mathbf{H}_2^{(i+1)}\}$ using (12).

Output. State filtering/smoothing pdfs (3) and (7) along with point-wise estimates of the latent factor from (12).

325 **3.1. Online implementation**

326 First, in order to better cope with high volatility of stock market quantities
327 and allow immediate feedback to the users for on-the-fly trading, we propose
328 here an online implementation of our DRDL approach. We make use of sliding
329 windows of size of $\tau \in \{1, \dots, K\}$ time steps. The model parameters are inferred
330 for every $k \in \{0, \dots, K - \tau\}$ using the last τ data points observed in the window,
331 i.e. $\mathcal{X}_k = \{\mathbf{x}_j\}_{j=k+1}^{k+\tau}$, then followed by the EM approach described in detail
332 above. The sliding window approach leverages two advantages. First, it helps
333 in faster processing of the sequence as one can choose a small number τ of
334 data points in the window. Second, it might provide better modeling, since the

335 constant linear factors assumption is likely to better model the time series if τ
 336 is small. However, a too small τ might also degrades the inference capabilities.
 337 Hence, it is essential to find a tradeoff in finding an optimal τ , as we will
 338 show in our experiments. When implementing the online strategy, a warm start
 339 approach is employed for the Kalman filter initialization. The observation/state
 340 factors are set to their most recent values, and the mean/covariance of the state
 341 for processing \mathcal{X}_{k+1} are initialized using the results of the processing of \mathcal{X}_k . Let
 342 us note that, when we set $\tau = K$, we retrieve the offline version of the algorithm,
 343 where the EM inference tool is ran only once.

344 3.2. Modeling and post-processing for stock market analysis tasks

345 Stock market data processing typically amounts to solving two distinct ap-
 346 plicative problems, namely daily stock price forecasting and stock trading deci-
 347 sion (among 3 options: buy/hold/sell) estimation. We hereafter explain how to
 348 post-process DRDL results to tackle both above-stated problems.

349 3.2.1. Stock forecasting

350 Let us first specify the observation model in stock forecasting. For a given
 351 window size $\tau > 0$, for each $k \in \{0, \dots, K - \tau\}$, we observe $(\mathbf{x}_j)_{k+1 \leq j \leq k+\tau} \in \mathbb{R}^{15}$,
 352 gathering 14 technical indicators ² as well as the adjusted close price. Running
 353 our DRDL on the considered window yields the following mean estimate of the
 354 15 quantities for the next time step indexed as $k + \tau + 1$:

$$\hat{\mathbf{x}}_{k+\tau+1} = \mathbf{H}_0 \mathbf{H}_1 \mathbf{H}_2 \mathbf{z}_{k+\tau|k+\tau-1}, \quad (13)$$

²We retained the relative strength index (RSI), the William percentage range, the absolute price oscillator (APO), the commodity channel index, the Chande momentum oscillator (CMO), the directional movement Indicator (DMI), the ultimator oscillator, the WMA, the exponential moving average (EMA), the Simple Moving Average (SMA), the triple EMA, the moving average convergence (MAC), the percentage price oscillator, the rate of change (ROC). Detailed definitions can be found in <https://www.investopedia.com/terms/t/technicalindicator.asp>

355 with the covariance matrix

$$\mathbf{S}_{k+\tau+1} = \mathbf{H}_0 \mathbf{H}_1 \mathbf{H}_2 (\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2 \mathbf{P}_{k+\tau} (\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2)^\top + \mathbf{Q}) + \mathbf{R}. \quad (14)$$

356 Hereabove, $\{\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2\}$ are the factors estimated during the M-
 357 step of our EM-based inference method, and $(\mathbf{z}_{k+\tau|k+\tau-1}, \mathbf{P}_{k+\tau})$ are byproducts
 358 of the Kalman prediction step (4)-(5), computed during the E-step of the EM.
 359 The proposed methodology aims at predicting the entire 15-dimensional vector.
 360 However, stock forecasting is typically focused on the prediction of a single
 361 quantity such as the adjusted close price.

362 3.2.2. Stock trading

363 In stock trading, a different set of inputs are passed to the model. For each
 364 window index $k \in \{0, \dots, K - \tau\}$, the observed data points $(\mathbf{x}_j)_{k+1 \leq j \leq k+\tau} \in$
 365 \mathbb{R}^{17} , $x_j[i]$, for $i \in \{1, \dots, 14\}$, are the same 14 technical indicators as in stock
 366 forecasting. Additionally, $[x_j[15], x_j[16], x_j[17]] \in \{0, 1\}^3$ gathers the decisions
 367 “hold”, “buy”, or “sell”, which are calculated for each stocks for every day so as
 368 to maximize the annualized returns. The labels are further turned into soft hot
 369 encoded vectors as explained in (Sharma et al., 2021, Sec. 3.3.2). The mean
 370 and covariance of these 17 quantities can be estimated for next day following
 371 (13) and (14). We then define our class label for next time step as

$$\ell_{k+\tau+1} = \operatorname{argmax}_{i \in \{1,2,3\}} \hat{x}_{k+\tau+1}[i + 14]. \quad (15)$$

372 3.3. Probabilistic assessment of stock trading decision

373 We now describe the procedure to assess the uncertainty quantification as-
 374 sociated to the DRDL predictions. Let $k \in \{0, \dots, K - \tau\}$ be the window index
 375 on which Algorithm 1 has been run. The probabilistic estimation of the quan-
 376 tities of interest for the next time step (i.e., one-day ahead prediction) $\mathbf{x}_{k+\tau+1}$
 377 conditioned to the data observed in the window $\mathbf{x}_{k:k+\tau}$, reads as a multivariate
 378 Gaussian distribution

$$p(\mathbf{x}_{k+\tau+1} | \mathbf{x}_{k:k+\tau}) = \mathcal{N}(\mathbf{x}_{k+\tau+1}; \hat{\mathbf{x}}_{k+\tau+1}, \mathbf{S}_{k+\tau+1}), \quad (16)$$

379 with mean and covariance ($\hat{\mathbf{x}}_{k+\tau+1}, \mathbf{S}_{k+\tau+1}$), given by (13) and (14), respec-
380 tively. Equation (16) assigns a probability score to any decision (e.g., trading)
381 based on the prediction output of the DRDL method. Let us focus on the
382 particular example of assessing the uncertainty of the stock trading decision at
383 time index $k + \tau + 1$, given observations at indexes $j \in \{k + 1, \dots, k + \tau\}$. The
384 trading decision relies on the discrete maximization step (15). Let us express
385 the probability mass function (pmf) of this decision, from the gaussian predic-
386 tive probability density function (pdf) of the observed data points in Eq. (16).
387 The pmf can here be summarized as $\mathbf{p}_{k+\tau+1} \in [0, 1]^3$ where each $p_{k+\tau+1}[i]$,
388 $i \in \{1, 2, 3\}$ is a probability, and $\sum_{i=1}^3 p_{k+\tau+1}[i] = 1$. Each $p_{k+\tau+1}[i]$ represents
389 the probability inferred by DRDL that the true value $x_{k+\tau+1}[i + 14]$ is greater
390 than $x_{k+\tau+1}[j + 14]$, for $j = \{1, 2, 3\} \setminus i$. According to Eqs. (16) and (15),
391 $\mathbf{p}_{k+\tau+1}$ can be obtained through

$$(\forall i \in \{1, 2, 3\}) \quad p_{k+\tau+1}[i] = \int_{\mathcal{Y}_i} \mathcal{N}(\mathbf{y}; \hat{\mathbf{x}}_{k+\tau+1}[15 : 17], \mathbf{S}_{k+\tau+1}[15 : 17, 15 : 17]) d\mathbf{y}, \quad (17)$$

392 with

$$\mathcal{Y}_i = \{\mathbf{y} \in \mathbb{R}^3 \mid \mathbf{y}[i] \geq \mathbf{y}[j], j = \{1, 2, 3\} \setminus i\}. \quad (18)$$

393 Due to the intricate form of the constrained set in (18), the integral in (17)
394 is intractable. It can be easily approximated with high precision by direct
395 simulation. In practice, we sampled 10^4 three-dimensional sample from a normal
396 standard distribution. The samples can be re-used for all time steps using
397 coloring and shifting according to the covariance and mean, respectively. Thanks
398 to this procedure, we can infer $\mathbf{p}_{k+\tau+1}$ for every k , and then assess the next day
399 stock trading outcome by using the standard cross-entropy loss:

$$\text{log-loss} = \frac{1}{K - \tau + 1} \sum_{k=0}^{K-\tau} \sum_{i=1}^3 -(L_{k+\tau+1}[i] \log(p_{k+\tau+1}[i])), \quad (19)$$

400 where the true labels are denoted $\mathbf{L}_{k+\tau+1} \in \{0, 1\}^3$ for each time $k + \tau + 1$
401 (hereagain, we use soft hot encoding representation).

402 *3.4. Summarized pipeline*

403 We provide in Alg. 2 the summary of our proposed pipeline for applying
 404 DRDL, in Algorithm 1, in the context of stock forecasting (steps a-b with $N_x =$
 405 15) and trading (steps a-b-c-d with $N_x = 17$).

Algorithm 2. *DRDL (3 layers) method for stock forecasting and trading.*

Inputs. *Prior parameters $(\bar{\mathbf{z}}_0, \mathbf{P}_0)$; model noise covariance matrices \mathbf{Q}, \mathbf{R} ; set of observations $\{\mathbf{x}_k\}_{1 \leq k \leq K}$; windows size τ .*

Initialization. *Set positive latent factors*

$\{\mathbf{D}_0^{(0)}, \mathbf{D}_1^{(0)}, \mathbf{D}_2^{(0)}, \mathbf{H}_0^{(0)}, \mathbf{H}_1^{(0)}, \mathbf{H}_2^{(0)}\}$.

Window processing. *For $k = 0, 1, \dots, K - \tau$:*

- a. *Run DRDL algorithm 1 on sequence $(\mathbf{x}_j)_{k+1 \leq j \leq k+\tau}$, initialized with estimates from $k - 1$ th window (warm start).*
- b. *Calculate one-step ahead predicted mean $\hat{\mathbf{x}}_{k+\tau+1}$ and its covariance $\mathbf{S}_{k+\tau+1}$ using (13)-(14).*
- c. *Compute one-step ahead predicted label $\ell_{k+\tau+1}$ using (15).*
- d. *Compute $\mathbf{p}_{k+\tau+1}$ using (17)-(18).*

Output. *Forecasting/trading predictions and log-loss value (19).*

406

407 **4. Experimental Results**

408 *4.1. Dataset*

409 The finance dataset used for experiments is curated from Yahoo finance
410 repository.³ We curated data for 180 stocks which comprises stocks from USA,
411 UK, India and China. The data is prepared by scrapping daily adjusted close
412 prices, open price, volume, high price, low price for a span of twenty years (i.e.,
413 01/01/1998 to 01/10/2019) using yahoo finance API for Python. Having stocks
414 from different market cap is always advisable by the traders, as it gives them
415 breadth while investing in advanced as well as emerging markets Fawaz et al.
416 (2019). The diversification also allows to assess the model robustness to various
417 trends Kumar & Shah (2009). From the knowledge of the close prices, we build
418 two observation sequences associated to the resolution of two specific problems,
419 namely stock forecasting and stock trading, as described in Sec. 3.2. The data
420 is scaled by normalising the 14 technical indicator values. For both problems,
421 we will compare DRDL and several state-of-the-art methods arising from signal
422 processing and machine learning literature. In all experiments, each of the 180
423 observed time series is split into two parts, namely a train phase made of the
424 first recorded 2546 days, and a test phase made of the next 2882 days. The train
425 phase is used to learn the models parameters (for instance, the linear factors
426 involved in DRDL), while the test phase is used to evaluate the performance of
427 the learnt models, their parameters being fixed. More details about DRDL and
428 the retained benchmark methods setting are provided in the next subsection.
429 Figure 2 displays the evolution of 4 of the 14 technical indicators used as input
430 of the inference tools, during the test phase. One can notice the high volatility
431 in the observed data.

³<https://yahoo.finance.com>

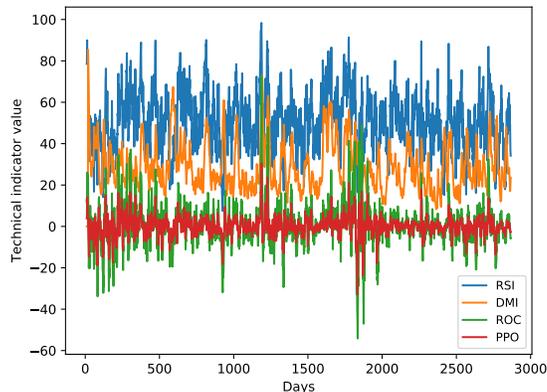


Figure 2: Evolution of four (among 14) observed technical indicators during the test phase.

432 *4.2. Practical Settings*

433 *4.2.1. DRDL settings*

434 As described in Sec. 3.2, DRDL can be specified to tackle both stock fore-
 435 casting problem, in which case $N_x = 15$, and stock trading problem where
 436 $N_x = 17$. Three variants of DRDL will be compared, depending on the number
 437 of linear factors (i.e., layers) in the multi-linear model. More specifically, we
 438 will distinguish in our experiments:

439 **DRDL (1 layer):** $\mathbf{D}_0 = \mathbf{D}_1 = \mathbf{D}_2 = \mathbf{Id}$ and $\mathbf{H}_1 = \mathbf{H}_2 = \mathbf{Id}$ fixed and $\{\mathbf{H}_0\}$ is
 440 estimated;

441 **DRDL (2 layers):** $\mathbf{D}_0 = \mathbf{D}_1 = \mathbf{D}_2 = \mathbf{Id}$ and $\mathbf{H}_2 = \mathbf{Id}$ fixed and $\{\mathbf{H}_0, \mathbf{H}_1\}$
 442 are estimated;

443 **DRDL (3 layers):** $\mathbf{D}_0 = \mathbf{D}_1 = \mathbf{D}_2 = \mathbf{Id}$ and $\{\mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2\}$ are estimated.

444 Note that, ignoring the positivity constraint, DRDL (1 layer) would identify with
 445 our previously published method RDL Sharma et al. (2021). We implement the
 446 sliding window approach described in Sec. 3.1, for various choices of τ described
 447 hereafter. In all experiments, the initial value are set as ; $\bar{\mathbf{z}}_0$ is an all zero

448 vector; $\mathbf{P}_0=10^{-7}\text{Id}$, $\mathbf{Q}=10^{-2}\text{Id}$ and $\mathbf{R}=10^{-2}\text{Id}$, where Id states for the identity
 449 matrix. Moreover, we set the dimension of the state as $N_z = 14$, which also
 450 corresponds to the number of measured technical indicators, we observed better
 451 performance of the model. The entries of the linear factors to estimate are
 452 initialized at time 0 using independent realizations of a uniform distribution on
 453 $[0, 10^{-1}]$. As mentioned in Sec. 3.1, to initialize the next processed windows, we
 454 used warm start strategy. The estimation of the linear factors (i.e. M-step of
 455 the EM method) is only conducted during the training phase. A maximum of
 456 50 iterations of the EM loop are used in Alg. 1, which was observed sufficient to
 457 reach stability of the estimated factors. During the test phase, the linear latent
 458 factors are fixed, and only the Kalman/RTS inference is ran (i.e., we inhibit
 459 M-step in Alg. 1). The scores shown are arithmetic means of 10 random trials,
 460 and are computed only during the test phase.

461 4.2.2. Compared methods

462 The proposed DRDL approach is analyzed by comparing with state-of-the-
 463 art-methods deep learning namely Multi-filter neural network Long et al. (2019),
 464 Long short term memory Fischer & Krauss (2018), 2-D deep CNN (CNN-TA)
 465 Sezer & Ozbayoglu (2018) and ARIMA Ariyo et al. (2014). We select the state-
 466 of-the-art methods for each task for a fair comparison. For stock forecasting, the
 467 comparison is done with ARIMA and LSTM. The ARIMA parameter value are
 468 set to $(p, d, q) = (5, 1, 5)$. The LSTM is customized from its original version to
 469 carry out regression tasks by replacing the softmax output layer with an affine
 470 layer. The Adam optimizer is used with learning rate 10^{-4} and 200 epochs.
 471 We used a mini-batch strategy where batch-size is fixed to 16 to reduce the
 472 objective function’s mean square error (MSE). The evaluation of the methods
 473 is done using metrics like mean absolute error (MAE), root means square error
 474 (RMSE), SMAPE (Symmetric mean absolute percentage error), and Pearson
 475 correlation factor.

476 For the stock trading task, the comparison is made with CNN-TA, Multi-
 477 filter neural network, and LSTM implemented with their original architecture

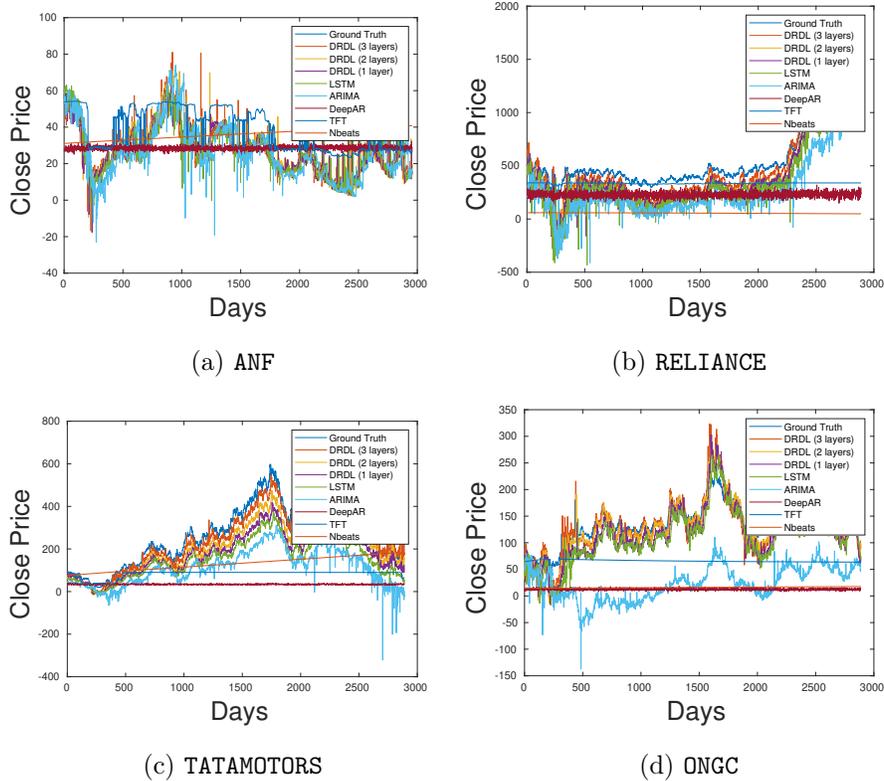


Figure 3: Ground truth and inferred adjusted close price on the test phase for four different stocks, using DRDL with 1 to 3 layers, LSTM or ARIMA.

478 and parameter values set to the specified ones in the respective paper Sezer
 479 & Ozbayoglu (2018), Long et al. (2019) and Fischer & Krauss (2018). The
 480 evaluation of the methods is done using classification metrics such as the F1-
 481 score, recall, precision for each class. We also performed trading simulation
 482 experiments in terms of annualized returns. We additionally present the log-
 483 loss values provided by DRDL (see Sec. 3.3) to illustrate how the probabilistic
 484 assessment can be used to let the researchers analyze market sentiments and
 485 diversify a balanced portfolio.

486 *4.2.3. Hardware and Software descriptions*

487 We curated the data using the python API. The data curation and experi-
488 mental results for the DRDL model are computed using Python 3.6 code. The
489 implementation is done using the potential python libraries like NumPy, scikit-
490 learn and pandas. In contrast, CNN-TA is implemented in its original version
491 using the keras. MFNN and LSTM are implemented with PyTorch. The tech-
492 nical indicators are evaluated using the libraries Ta-lib⁴ and Ta4j⁵. Provided
493 computational times correspond to codes running on Xeon E3-1225V5 clocked
494 at 3.3GHz, with a 4Gb GPU (GeForce GT 730), 16GB RAM, 200GB HDD and
495 Ubuntu OS.

496 *4.3. Numerical results for stock forecasting problem*

497 *4.3.1. Influence of window size*

498 The choice of window size is an essential aspect as it can enhance and limit
499 the methodology’s potential. To understand the model behavior, we present
500 Table 1 which provides detailed information on the performance of the model on
501 varying window sizes. The table offers an analysis of various metrics like Pearson
502 correlation (r), RMSE (Root Mean Square Error), MAE (Mean Absolute Error),
503 and SMAPE (Symmetric mean absolute percentage error) for different window
504 sizes τ . The model’s performance improves as the window size increases till
505 a stabilization point. We can see that a balanced choice is $\tau = 650$ to reach
506 stabilized performance on this particular task and dataset. We further use this
507 value in upcoming experiments.

508 *4.3.2. Comparison with benchmark models*

509 To understand better, we present table 2 which provides comprehensive anal-
510 ysis on performance estimation on the stock forecasting problem using DRDL,
511 LSTM, ARIMA, DeepAR Salinas et al. (2020), Nbeats Oreshkin et al. (2019),
512 and TFT Lim et al. (2019). Table 2 presents comparison in terms of Pearson

⁴<http://ta-lib.org>

⁵<http://www.ta4j.org>

Window size τ	r	RMSE \downarrow	MAE (%) \downarrow	SMAPE (%) \downarrow
250	0.45	29.43	0.47	31.8
300	0.49	27.81	0.23	28.6
350	0.53	21.61	0.29	25.5
500	0.69	13.79	0.13	23.4
650	0.71	13.35	0.11	18.4
700	0.72	13.62	0.10	18.5

Table 1: Results of DRDL (3 layers) on stock forecasting problem for different window size. Scores averaged on test phase, on all the 180 stocks.

Model	r	RMSE \downarrow	MAE(%)	SMAPE(%)
ARIMA	0.13	78.6 (1.89)	1.23(0.56)	65.5
LSTM	0.24	297.5 (2.64)	6.12 (0.65)	47
DeepAR	0.40	73.89 (1.85)	0.43(0.34)	58.13
Nbeats	0.38	95.52 (1.99)	0.57 (0.12)	63.75
TFT	0.52	35.79(1.56)	0.35(0.019)	38.25
DRDL (1 layer)	0.65	23.24 (1.64)	0.19(0.02)	35.3
DRDL (2 layers)	0.69	14.2 (1.43)	0.15(0.006)	23.2
DRDL (3 layers)	0.71	13.35(0.37)	0.11(0.003)	18.4

Table 2: Comparative analysis of DRDL against state-of-the-art methods for stock forecasting problem: Pearson correlation score (r), MAE, RMSE, SMAPE scores and their respective std. deviation on the estimation of next time step adjusted close price, averaged over the data in the test set and the stocks.

513 correlation factor r , RMSE, MAE and SMAPE. We can see that DRDL (3 lay-
514 ers) architecture outperforms DRDL (2 layers), DRDL (1 layer) as well as the
515 other benchmarks models. We also notice that the average performance of TFT
516 is comparable to DRDL (1 layer) architecture. Fig A.4 in appendix section

Method	Train Time cost (h.)	Test Time cost (min.)
DRDL (3 layers)	2.37h	20 min
DRDL (2 layers)	2.12h	18.4 min
DRDL (1 layer)	1.78h	15.8 min
ARIMA	2.01h	36 min
LSTM	8 days	45 min
DeepAR	2.45h	20 min
TFT	2.25h	27 min
Nbeats	3.12h	25 min

Table 3: Averaged time over 10 random runs for processing the dataset (train(hrs) and test(min)), for DRDL and its competitors.

Method	Sharpe Ratio	T-test
DRDL (3 layers)	2.14	0.63
DRDL (2 layers)	1.99	0.78
DRDL (1 layer)	1.84	0.83
ARIMA	1.03	0.89
LSTM	0.88	1.34
DeepAR	2.06	0.58
TFT	1.22	0.67
Nbeats	1.33	0.54

Table 4: Comparison of Sharpe ratio and T-test score, for DRDL and its competitors averaged over 180 stocks.

517 Appendix A displays the Pearson correlation analysis between ground truth
518 daily adjusted close price time series and predicted ones along test phase, using
519 DRDL (3 layers) for four representative stock cases. Table 4 presents the statisti-
520 cal test (t-test) and stock market simulation (Sharpe Ratio) on the forecasting
521 results. We observe that the proposed method with 3 layers architecture gives
522 better performance as its average score for 185 stocks is smallest as compared to
523 other state-of-the-art methods hence we can conclude that more similarity exists

524 between the actual closing prices and predicted closing prices when compared
525 for different state-of-the art methods. We also present an average analysis of
526 Sharpe ratio for proposed method and its competitors which gives more infor-
527 mation on the risk- adjusted return on the investment. A higher Sharpe ratio
528 indicates good investment returns, likewise we see that the forecast estimation
529 from DRDL (3 layers) and DeepAR yields a higher Sharpe ratio as compared
530 to other state-of-the-art method.

Method	F1 Score			Precision			Recall			Training time (in hrs)	Testing Time (in min)
	Hold	Sell	Buy	Hold	Sell	Buy	Hold	Sell	Buy		
DRDL (3 layers)	0.61	0.30	0.29	0.85	0.26	0.29	0.51	0.54	0.54	4.12	10.17
DRDL (2 layers)	0.61	0.32	0.34	0.83	0.23	0.26	0.48	0.51	0.53	3.56	12.50
DRDL (1 layer)	0.59	0.19	0.23	0.88	0.15	0.12	0.45	0.52	0.51	2.00	11.56
MFNN	0.58	0.11	0.06	0.79	0.11	0.04	0.47	0.37	0.16	5.34	14.23
LSTM	0.86	0.05	0.05	0.84	0.07	0.06	0.89	0.05	0.05	12.53	13.50
CNN-TA	0.85	0.08	0.09	0.84	0.11	0.09	0.85	0.07	0.10	4.57	14.36

Table 5: Comparison of classification scores of different methods on the stock trading problem. All scores are averaged over 180 stocks and over the days of the test phase.

Window size (τ)	F1 Score			Precision			Recall		
	Hold	Sell	Buy	Hold	Sell	Buy	Hold	Sell	Buy
250	0.61	0.23	0.19	0.91	0.15	0.12	0.46	0.51	0.50
300	0.61	0.23	0.19	0.91	0.15	0.12	0.46	0.50	0.51
350	0.61	0.26	0.27	0.89	0.18	0.19	0.47	0.53	0.52
500	0.61	0.26	0.27	0.89	0.18	0.19	0.47	0.52	0.53
650	0.61	0.30	0.29	0.85	0.26	0.29	0.51	0.54	0.54
700	0.61	0.30	0.29	0.87	0.21	0.20	0.48	0.53	0.54

Table 6: Classification scores of DRDL (3 layers) for varying window size.

531 Table 3 presents the computational time for forecasting the next day closing
532 price for our dataset. We distinguish the time required to train the methods
533 (on the first ten years) and to test them (on the next ten years) using the walk-
534 forward method described in (Sharma et al., 2021, Section 4.2.1). We observed

Window size (τ)	F1 Score			Precision			Recall		
	Hold	Sell	Buy	Hold	Sell	Buy	Hold	Sell	Buy
250	0.61	0.21	0.19	0.91	0.14	0.12	0.47	0.49	0.48
300	0.61	0.24	0.20	0.90	0.16	0.13	0.47	0.50	0.49
350	0.62	0.25	0.24	0.89	0.17	0.16	0.48	0.51	0.52
500	0.62	0.25	0.24	0.89	0.17	0.16	0.48	0.52	0.51
650	0.61	0.32	0.34	0.83	0.23	0.26	0.48	0.51	0.53
700	0.59	0.33	0.32	0.84	0.20	0.23	0.46	0.53	0.51

Table 7: Classification scores of DRDL (2 layers) with varying window size.

Window size (τ)	F1 Score			Precision			Recall		
	Hold	Sell	Buy	Hold	Sell	Buy	Hold	Sell	Buy
250	0.84	0.10	0.10	0.85	0.10	0.10	0.85	0.12	0.12
300	0.80	0.10	0.12	0.85	0.10	0.10	0.74	0.17	0.14
350	0.68	0.15	0.15	0.82	0.10	0.10	0.62	0.30	0.31
500	0.59	0.18	0.22	0.86	0.15	0.14	0.46	0.51	0.51
650	0.59	0.19	0.23	0.88	0.15	0.12	0.45	0.52	0.51
700	0.59	0.24	0.22	0.90	0.16	0.14	0.46	0.51	0.52

Table 8: Classification scores of DRDL (1 layer) with varying window size.

535 the highest computational time with the LSTM approach. The other methods
536 have rather similar computational time, DRDL (1 layer) being the fastest. The
537 computational time of DRDL (3 layers), reaching the best performance metrics,
538 stays reasonable, and is comparable with the one of DeepAR and TFT. Here, we
539 must recall that, in contrast with most of its competitors (except ARIMA), our
540 implementation of DRDL method (for both train/test phases) does not exploit
541 GPU facilities such as PyTorch. Complexity reductions could certainly occur if
542 this was the case.

543 The stock forecasting results are presented in Fig. 3. The comparison is
544 carried out between the proposed DRDL for different layer number, LSTM,
545 ARIMA, DeepAR, TFT, Nbeats method. We observed that in some cases (c-

Stock symbols	DRDL (3 layers)	DRDL (2 layers)	DRDL (1 layer)	CNN-TA	MFNN	LSTM
WIPRO.BO	-13.89	-23.26	-29.14	-18.14	-27.81	-47.74
AAPL	19.12	11.3	10.14	0	12.92	0
AMZN	-13.23	-11.92	21.23	30.64	-20.85	-0.15
IOC.BO	-13.48	-23.28	-2.68	-3.03	-26.42	-3.1
TATACHEM.BO	1.23	3.83	2.19	-1.54	-8.32	0
SPICEJET.BO	11.92	10.17	-8.63	-24.08	-28.21	0
ATML	-4.13	-5.78	-10.19	-33.25	-27.07	-33.82
DOM.L	4.56	9.34	2.83	0.11	8.22	0.47
INDRAMEDCO.BO	-5.78	-10.34	-3.65	-14.22	-3.53	-50.86
Average on all 180 stocks	3.87	2.67	2.34	-5.08	-11.45	-13.02

Table 9: Annualized returns resulting from the stock trading decisions of different methods during the test phase.

546 d), LSTM approach failed to reach satisfying results which might be due to
547 vanishing gradient issues. In cases (a-b), ARIMA performs quite good when
548 compared to its performance in other cases (c-d). In contrast, DRDL (3 layers)
549 reaches stable and satisfactory outcomes. DRDL (2 layers) outperforms DRDL
550 (1 layer) and both benchmark methods but remains lower quality than its 3-
551 layers variant.

552 4.4. Numerical results for the stock trading problem

553 4.4.1. Influence of the window size

554 The challenging task with the DRDL approach is to preserve a balance be-
555 tween the computational time and optimal predictions. We experimented with
556 various window sizes to analyze and preserve the best parameter for our future
557 experiments. We present Table 6 which depicts the experimental performance
558 of DRDL (3 layers) architecture, Table 7 depicts the experimental performance
559 of DRDL (2 layers) architecture, Table 8 depicts the experimental performance
560 of DRDL (1 layer) architecture for different window sizes. The empirical results
561 state that the performance of the approach increases as it feeds more data to the
562 model for better understanding. We can preserve a balance parameter $\tau = 650$,

563 which indicates stabilized performance. We further use this value in upcoming
564 experiments.

565 4.4.2. Classification metrics

566 To explain the empirical analyses of the trading process (classification), we
567 present confusion matrices. The trading process involves classifying the signal
568 into three classes, namely "Buy," "hold," and "sell" classes. The summarized
569 performance for 180 stocks by DRDL and other state-of-the-art methods is pre-
570 sented in Fig. A.5 in Appendix A. Among the three classes, we see the predic-
571 tion of hold class is captured efficiently when compared to the other classes. The
572 LSTM approach predicts the best score over the other state-of-the-art methods
573 when compared to false negatives scores. However, in LSTM and CNN-TA ap-
574 proaches are highlighted many false positives for the "hold" class. It can be
575 noted that these deep learning techniques have labeled most signals as hold
576 class, jeopardizing the model behavior for the other classes ("buy," "sell"). The
577 nature of the finance market is highly volatile and non-linear; hence we get to
578 see a highly imbalanced dataset. However, we noticed that the DRDL approach
579 handles it by imposing an activation function on the operators. These opera-
580 tors are expected to evolve continuously as we grow deeper with time sequence.
581 Table. 5 and Fig. A.5 add more weight to the analysis. The results state that
582 the DRDL approach managed to predict the highly unbalanced data. The sen-
583 sitivity score (Recall) is presented well by the DRDL approach compared to the
584 state-of-the-art methods. The diagonals of the confusion matrix of the DRDL
585 approach also takes the maximum values, which a valid classifier should expect.
586 When dealing with highly imbalanced dataset such as finance dataset, it is more
587 important to study classification metrics like F1 Score, Precision and Recall for
588 each class. This helps in analyzing the model behavior for each class. The Table
589 5 presents analysis of these classification metrics for DRDL compared to other
590 deep learning state-of-the-art methods. We conclude that DRDL outperforms
591 the other methods stated.

592 In Table 5, we also present the computational times (train and test) for
593 conducting trading simulations for our dataset. Hereagain, LSTM is the more
594 demanding method at training. All methods have rather comparable test times,
595 despite DRDL is implemented on CPU only. In particular, besides its proba-
596 bilistic output, DRDL is not more costly than its competitors. Adding more
597 layers to DRDL slightly increases its train time, but does not affect much the
598 test time.

599 *4.4.3. Annualized Returns*

600 Stock market aims to analyze and evaluate the return on investment for
601 a given stock. Every trader is indeed interested in evaluating his investment
602 returns and taking risks accordingly. We simulate market scenarios Sezer &
603 Ozbayoglu (2018) by evaluating the annualized returns by the predicted stock
604 trading decisions provided by DRDL using 1 to 3 layers as well as the decisions
605 from from the benchmark models. Table 9 presents a detailed study of nine
606 stocks for DRDL methodology and state-of-the-art methods. We display only
607 empirical values for nine stocks and the average results over the 180 stocks. To
608 make it easy for readers we have highlighted best annualized returns in bold.
609 It is clearly evident that the DRDL approach yields higher returns when tested
610 for a duration of 10 years when compared to annualized returns obtained from
611 deep learning state-of-the-art methods predictions.

612 *4.4.4. Portfolio diversification*

613 Many researchers and traders believe that it is essential to know the asso-
614 ciated sentiments associated with each stock to understand the stock market.
615 Traders support and recommend having a mix of stock sentiments in one’s
616 portfolio. The market is very well divided into three types of stock sentiments:
617 small-cap, mid-cap, and large-cap. To read about them in detail please refer
618 to Sharma et al. (2021)[section 4.4.4]. To evaluate this sentiment using the
619 predicted signals from the proposed approach, we calculated probabilistic quan-
620 tification, as explained in sec. 3.3. The practitioner uses this quantification

	Stock symbols	DRDL 3 layers	DRDL 2 layers	DRDL 1 layer
Small-cap	ALOKTEXT.BO	1.04	1.20	1.09
	ALKYLAMINE.BO	1.17	1.19	1.34
	ZEEMEDIA6.BO	0.89	0.99	0.23
	PVP.BO	1.34	1.98	2.78
Mid-cap	IOC.BO	1.02	1.05	0.87
	TATACHEM.BO	0.76	0.45	0.94
	SPICEJET.BO	0.34	0.65	1.20
	BHEL.BO	0.20	0.51	1.15
Large-cap	AAPL	1.13	0.98	1.11
	AMZN	0.11	0.41	0.43
	HINDZINC.BO	0.03	0.65	0.45
	ONGC.BO	0.20	0.13	0.09
	SIEMENS.NS	0.12	0.02	0.11

Table 10: Comparative analysis of uncertainty quantification provided by DRDL using 1 to 3 layers. The quantification is listed for stocks with market capitalization categories. The log-loss is computed over the test phase.

621 score to have a well-diversified portfolio. The score provides a confidence score
622 that helps the investor decide where to invest in the market to have a balance
623 of market sentiments and maximize returns.

624 To understand further, we present Table 10 which provides a log-loss score.
625 The log-loss score provides the confidence score in terms of its volatility nature,
626 where the smaller value is considered, the better and less volatile. We evaluated
627 the confidence score for the proposed approach for different configuration. The
628 market capitalization of these stocks can be found ⁶. The log-loss value provides
629 the probabilistic inference for the predictions. The inference tries to penalize

⁶<https://finance.yahoo.com/screener>

630 the events for which the method assigns a low probability. We observed that the
631 log-loss value reached a meager value which indicated good prediction accuracy
632 in large-cap stocks, which are expected to be least volatile. In contrast, we
633 achieved a higher log-loss value for predictions associated with small-cap stocks
634 as they are highly unstable and new to the market.

635 **Acknowledgment**

636 The CNRS-CEFIPRA project supported this work under grant NextGenBP
637 PRC2017. E.C. acknowledges support from the European Research Council
638 Starting Grant MAJORIS ERC-2019-STG-850925.

639 **5. Conclusion**

640 In our approach, time-series sequences are modeled with a flexible Gaus-
641 sian SSM. The transition matrices (state and observation models) are unknown,
642 and are estimated thanks to an expectation-minimization strategy, assuming a
643 particular deep NMF structure. The DRDL approach inherits advantages from
644 sophisticated modeling techniques while quantifying the uncertainty in the pre-
645 dictions. We have then adapted the DRDL approach to deal with a challenging
646 large scale financial time series problem, to target stock forecasting and trading
647 tasks. In particular, the method is able to successfully operate in an online
648 processing manner, allowing to capture piece-wise linear characteristics in the
649 data. The results show that the proposed method outperforms the state-of-the-
650 art techniques. Given these promising results, we plan as future work to delve
651 deeper into the area of financial forecasting, including the application of our
652 technique in forecasting derivatives.

653 **References**

654 Ababio, K. A. (2012). Comparative study of stock price forecasting using arima
655 and arimax models. *Kwame Nkrumah University Of Science And Technology*,

656 .

- 657 Abbe, E., & Sandon, C. (2018). Provable limitations of deep learning. *arXiv*
658 *preprint arXiv:1812.06369*, .
- 659 Andrieu, C., Doucet, A., & Holenstein, R. (2010). Particle markov chain monte
660 carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical*
661 *Methodology)*, 72, 269–342.
- 662 Ariyo, A. A., Adewumi, A. O., & Ayo, C. K. (2014). Stock price prediction
663 using the arima model. In *2014 UKSim-AMSS 16th International Conference*
664 *on Computer Modelling and Simulation* (pp. 106–112). IEEE.
- 665 Atsalakis, G., & Valavanis, K. P. (2010). Surveying stock market forecasting
666 techniques-part i: Conventional methods. *Journal of Computational Opti-*
667 *mization in Economics and Finance*, 2, 45–92.
- 668 Bach, F. R., & Jordan, M. I. (2004). Learning graphical models for stationary
669 time series. *IEEE Transactions on Signal Processing*, 52, 2189–2199.
- 670 Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H. et al. (2007). Greedy layer-
671 wise training of deep networks. *Advances in neural information processing*
672 *systems*, 19, 153.
- 673 Bianchini, M., & Scarselli, F. (2014). On the complexity of neural network clas-
674 sifiers: A comparison between shallow and deep architectures. *IEEE Trans-*
675 *actions on Neural Networks and Learning Systems*, 25, 1553–1565.
- 676 Chadsuthi, S., Modchang, C., Lenbury, Y., Iamsirithaworn, S., & Triampo, W.
677 (2012). Modeling seasonal leptospirosis transmission and its association with
678 rainfall and temperature in thailand using time-series and arimax analyses.
679 *Asian Pacific Journal of Tropical Medicine*, 5, 539–546.
- 680 Chauhan, L., Alberg, J., & Lipton, Z. (2020). Uncertainty-aware lookahead
681 factor models for quantitative investing. In H. D. III, & A. Singh (Eds.),
682 *Proceedings of the 37th International Conference on Machine Learning* (pp.
683 1489–1499). PMLR volume 119 of *Proceedings of Machine Learning Research*.

- 684 Chen, M., Jiang, H., Liao, W., & Zhao, T. (2019). Efficient approximation of
685 deep relu networks for functions on low dimensional manifolds. *Advances in*
686 *neural information processing systems*, 32.
- 687 Chen, Z., Jin, S., Liu, R., & Zhang, J. (2021). A deep non-negative matrix
688 factorization model for big data representation learning. *Frontiers in Neuro-*
689 *robotics*, 15. URL: [https://www.frontiersin.org/article/10.3389/fn](https://www.frontiersin.org/article/10.3389/fnbot.2021.701194)
690 [bot.2021.701194](https://www.frontiersin.org/article/10.3389/fnbot.2021.701194). doi:10.3389/fnbot.2021.701194.
- 691 Cheng, D., Yang, F., Xiang, S., & Liu, J. (2022). Financial time series fore-
692 casting with multi-modality graph neural network. *Pattern Recognition*, 121,
693 108218.
- 694 Cheridito, P., Jentzen, A., & Rossmannek, F. (2021). Efficient approximation
695 of high-dimensional functions with neural networks. *IEEE Transactions on*
696 *Neural Networks and Learning Systems*, .
- 697 Chong, E., Han, C., & Park, F. C. (2017). Deep learning networks for stock
698 market analysis and prediction: Methodology, data representations, and case
699 studies. *Expert Systems with Applications*, 83, 187–205.
- 700 Chopin, N., Jacob, P. E., & Papaspiliopoulos, O. (2013). SMC2: an efficient
701 algorithm for sequential analysis of state space models. *Journal of the Royal*
702 *Statistical Society: Series B (Statistical Methodology)*, 75, 397–426.
- 703 Choudhry, T., & Wu, H. (2009). Forecasting the weekly time-varying beta of
704 uk firms: Garch models vs. kalman filter method. *The European Journal of*
705 *Finance*, 15, 437–444.
- 706 Chouzenoux, E., & Elvira, V. (2020). Graphem: Em algorithm for blind kalman
707 filtering under graphical sparsity constraints. In *ICASSP 4 May 2020- 8*
708 *May 2020 IEEE International Conference on Acoustics, Speech and Signal*
709 *Processing (ICASSP), Barcelona* (pp. 5840–5844). IEEE.

- 710 Cichocki, A., Zdunek, R., Phan, A., & Amari, S. (2009). *Nonnegative Matrix and*
711 *Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis*
712 *and Blind Source Separation*. Wiley-Blackwell.
- 713 Crisan, D., & Miguez, J. (2018). Nested particle filters for online parameter
714 estimation in discrete-time state-space markov models. *Bernoulli*, *24*, 3039–
715 3086.
- 716 Daubechies, I., DeVore, R., Foucart, S., Hanin, B., & Petrova, G. (2022). Non-
717 linear approximation and (deep) relu networks. *Constructive Approximation*,
718 *55*, 127–172.
- 719 De Handschutter, P., Gillis, N., & Siebert, X. (2021). A survey on deep matrix
720 factorizations. *Computer Science Review*, *42*, 100423.
- 721 Devi, B. U., Sundar, D., & Alli, P. (2013). An effective time series analysis for
722 stock trend prediction using arima model for nifty midcap-50. *International*
723 *Journal of Data Mining & Knowledge Management Process*, *3*, 65.
- 724 Ding, X., Zhang, Y., Liu, T., & Duan, J. (2015). Deep learning for event-driven
725 stock prediction. In *Twenty-fourth international joint conference on artificial*
726 *intelligence*.
- 727 Djuric, P. M., Kotecha, J. H., Zhang, J., Huang, Y., Ghirmai, T., Bugallo, M. F.,
728 & Miguez, J. (2003). Particle filtering. *IEEE signal processing magazine*, *20*,
729 19–38.
- 730 Doucet, A., & Johansen, A. M. (2009). A tutorial on particle filtering and
731 smoothing: Fifteen years later. *Handbook of nonlinear filtering*, *12*, 3.
- 732 Elvira, V., & Chouzenoux, E. (2022). Graphical inference in linear-Gaussian
733 state-space models. *IEEE Transactions on Signal Processing*, *70*, 4757–4771.
- 734 Elvira, V., Martino, L., Bugallo, M. F., & Djuric, P. M. (2019). Elucidating
735 the auxiliary particle filter via multiple importance sampling [lecture notes].
736 *IEEE Signal Processing Magazine*, *36*, 145–152.

- 737 Fama, E. F. (2021). Efficient capital markets a review of theory and empirical
738 work. *The Fama Portfolio*, (pp. 76–121).
- 739 Fawaz, H. I., Forestier, G., Weber, J., Idoumghar, L., & Muller, P.-A. (2019).
740 Deep learning for time series classification: a review. *Data Mining and Knowl-*
741 *edge Discovery*, *33*, 917–963.
- 742 Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory
743 networks for financial market predictions. *European Journal of Operational*
744 *Research*, *270*, 654–669.
- 745 Flenner, J., & Hunter, B. (2017). A deep non-negative matrix factorization
746 neural network. *Semantic Scholar*, . [https://www1.cmc.edu/pages/facul-](https://www1.cmc.edu/pages/faculty/BHunter/papers/deep-negative-matrix.pdf)
747 [ty/BHunter/papers/deep-negative-matrix.pdf](https://www1.cmc.edu/pages/faculty/BHunter/papers/deep-negative-matrix.pdf).
- 748 Gonzalez-Olvera, M. A., & Tang, Y. (2010). Black-box identification of a class
749 of nonlinear systems by a recurrent neurofuzzy network. *IEEE Transactions*
750 *on Neural Networks*, *21*, 672–679.
- 751 Jacobson, M. W., & Fessler, J. A. (2007). An expanded theoretical treatment
752 of iteration-dependent majorize-minimize algorithms. *IEEE Transactions on*
753 *Image Processing*, *16*, 2411–2422.
- 754 Jiang, W. (2021). Applications of deep learning in stock market prediction:
755 Recent progress. *Expert Systems with Applications*, *184*, 115537. URL: [http](http://www.sciencedirect.com/science/article/pii/S0957417421009441)
756 [s://www.sciencedirect.com/science/article/pii/S0957417421009441](http://www.sciencedirect.com/science/article/pii/S0957417421009441).
757 doi:<https://doi.org/10.1016/j.eswa.2021.115537>.
- 758 Kalman, R. E. (1960). A new approach to linear filtering and prediction prob-
759 lems. *ASME, Journal Basic Eng.*, *82*, 35–45 (11 pages).
- 760 Khan, M. E., & Dutt, D. N. (2007). An expectation-maximization algorithm
761 based Kalman smoother approach for event-related desynchronization (ERD)
762 estimation from EEG. *IEEE Transactions on Biomedical Engineering*, *54*,
763 1191–1198.

- 764 Kumar, M., & Thenmozhi, M. (2014). Forecasting stock index returns using
765 arima-svm, arima-ann, and arima-random forest hybrid models. *International*
766 *Journal of Banking, Accounting and Finance*, 5, 284–308.
- 767 Kumar, V., & Shah, D. (2009). Expanding the role of marketing: from customer
768 equity to market capitalization. *Journal of Marketing*, 73, 119–136.
- 769 Li, Z. L., Zhang, G. W., Yu, J., & Xu, L. Y. (2023). Dynamic graph structure
770 learning for multivariate time series forecasting. *Pattern Recognition*, (p.
771 109423).
- 772 Lim, B., Arik, S. Ö., Loeff, N., & Pfister, T. (2019). Temporal fu-
773 sion transformers for interpretable multi-horizon time series forecasting.
774 *CoRR*, *abs/1912.09363*. URL: <http://arxiv.org/abs/1912.09363>.
775 *arXiv:1912.09363*.
- 776 Liu, B., & Liang, Y. (2021). Optimal function approximation with relu neural
777 networks. *Neurocomputing*, 435, 216–227.
- 778 Ljung, L. (1979). Asymptotic behavior of the extended kalman filter as a param-
779 eter estimator for linear systems. *IEEE Transactions on Automatic Control*,
780 24, 36–50.
- 781 Long, W., Lu, Z., & Cui, L. (2019). Deep learning-based feature engineering for
782 stock price movement prediction. *Knowledge-Based Systems*, 164, 163–173.
- 783 Mahdizadehghadam, S., Panahi, A., Krim, H., & Dai, L. (2019). Deep dictio-
784 nary learning: A parametric network approach. *IEEE Transactions on Image*
785 *Processing*, 28, 4790–4802.
- 786 Makridakis, S., & Hibon, M. (1997). Arma models and the box–jenkins method-
787 ology. *Journal of Forecasting*, 16, 147–163.
- 788 Moon, J., Hossain, M. B., & Chon, K. H. (2021). Ar and arma model order selec-
789 tion for time-series modeling with imagenet classification. *Signal Processing*,
790 183, 108026.

- 791 Newman, K., King, R., Elvira, V., de Valpine, P., McCrea, R. S., & Morgan,
792 B. J. (2023). State-space models for ecological time-series data: Practical
793 model-fitting. *Methods in Ecology and Evolution*, *14*, 26–42.
- 794 Nowicka-Zagrajek, J., & Weron, R. (2002). Modeling electricity loads in californ-
795 nia: Arma models with hyperbolic noise. *Signal Processing*, *82*, 1903–1915.
- 796 Ntemi, M., & Kotropoulos, C. (2021). A jump-diffusion particle filter for price
797 prediction. *Signal Processing*, *183*, 107994.
- 798 O’Donovan, T. M. (1983). Short term forecasting: An introduction to the box-
799 jenkins approach. *John Wiley & Sons, INC., 605 Third AVE., New York,*
800 *NY 10158, USA, 1983, 256, .*
- 801 Oreshkin, B. N., Carпов, D., Chapados, N., & Bengio, Y. (2019). N-beats:
802 Neural basis expansion analysis for interpretable time series forecasting. *arXiv*
803 *preprint arXiv:1905.10437, .*
- 804 Östermark, R. (1991). Vector forecasting and dynamic portfolio selection: Em-
805 pirical efficiency of recursive multiperiod strategies. *European Journal of Op-*
806 *erational Research*, *55*, 46–56.
- 807 Pai, P.-F., & Lin, C.-S. (2005). A hybrid arima and support vector machines
808 model in stock price forecasting. *Omega*, *33*, 497–505.
- 809 Petrica, A.-C., Stancu, S., & Tindeche, A. (2016). Limitation of arima models
810 in financial and monetary economics. *Theoretical & Applied Economics*, *23*.
- 811 Rigotti, L., & Shannon, C. (2005). Uncertainty and risk in financial markets.
812 *Econometrica*, *73*, 203–243.
- 813 Saad, E. W., Prokhorov, D. V., & Wunsch, D. C. (1998). Comparative study
814 of stock trend prediction using time delay, recurrent and probabilistic neural
815 networks. *IEEE Transactions on Neural Networks*, *9*, 1456–1470.

- 816 Saini, N., Mittal, A. K. et al. (2014). Forecasting volatility in indian stock
817 market using state space models. *Journal of Statistical and Econometric*
818 *Methods*, 3, 115–136.
- 819 Salinas, D., Flunkert, V., Gasthaus, J., & Januschowski, T. (2020). Deepar:
820 Probabilistic forecasting with autoregressive recurrent networks. *Interna-*
821 *tional Journal of Forecasting*, 36, 1181–1191.
- 822 Särkkä, S. (2013). *Bayesian filtering and smoothing*. 3. Cambridge University
823 Press.
- 824 Sezer, O. B., & Ozbayoglu, A. M. (2018). Algorithmic financial trading with
825 deep convolutional neural networks: Time series to image conversion ap-
826 proach. *Applied Soft Computing*, 70, 525–538.
- 827 Shah, D., Isah, H., & Zulkernine, F. (2019). Stock market analysis: A review
828 and taxonomy of prediction techniques. *International Journal of Financial*
829 *Studies*, 7, 26.
- 830 Shao, L., Wu, D., & Li, X. (2014). Learning deep and wide: A spectral method
831 for learning deep networks. *IEEE Transactions on Neural Networks and*
832 *Learning Systems*, 25, 2303–2308.
- 833 Sharma, S., Elvira, V., Chouzenoux, E., & Majumdar, A. (2021). Recurrent
834 dictionary learning for state-space models with an application in stock fore-
835 casting. *Neurocomputing*, 450, 1–13.
- 836 Sharma, S., Majumdar, A., Elvira, V., & Chouzenoux, E. (2020). Blind kalman
837 filtering for short-term load forecasting. *IEEE Transactions on Power Sys-*
838 *tems*, 35, 4916–4919.
- 839 Shumway, R. H., & Stoffer, D. S. (1982). An approach to time series smoothing
840 and forecasting using the em algorithm. *Journal of Time Series Analysis*, 3,
841 253–264.

- 842 Tariyal, S., Majumdar, A., Singh, R., & Vatsa, M. (2016). Deep dictionary
843 learning. *IEEE Access*, *4*, 10096–10109.
- 844 Tino, P., Schittenkopf, C., & Dorffner, G. (2001). Financial volatility trading
845 using recurrent neural networks. *IEEE Transactions on Neural Networks*, *12*,
846 865–874.
- 847 Trigeorgis, G., Bousmalis, K., Zafeiriou, S., & Schuller, B. W. (2016). A deep
848 matrix factorization method for learning attribute representations. *IEEE*
849 *Transactions on Pattern Analysis and Machine Intelligence*, *39*, 417–429.
- 850 Wan, E. A., & Van Der Merwe, R. (2000). The unscented kalman filter for
851 nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems*
852 *for Signal Processing, Communications, and Control Symposium (Cat. No.*
853 *00EX373)* (pp. 153–158). Ieee.
- 854 Wells, C. (2013). *The Kalman filter in finance* volume 32. Springer Science &
855 Business Media.
- 856 Won, S. H., Song, I., Lee, S. Y., & Park, C. H. (2010). Identification of finite
857 state automata with a class of recurrent neural networks. *IEEE Transactions*
858 *on Neural Networks*, *21*, 1408–1421.
- 859 Xue, H.-J., Dai, X., Zhang, J., Huang, S., & Chen, J. (2017). Deep matrix
860 factorization models for recommender systems. In *IJCAI* (pp. 3203–3209).
861 Melbourne, Australia volume 17.
- 862 Yang, Q., & Wu, X. (2006). 10 challenging problems in data mining research.
863 *International Journal of Information Technology & Decision Making*, *5*, 597–
864 604.
- 865 Yin, T., Liu, C., Ding, F., Feng, Z., Yuan, B., & Zhang, N. (2022). Graph-based
866 stock correlation and prediction for high-frequency trading systems. *Pattern*
867 *Recognition*, *122*, 108209.

- 868 Yu, L., Liu, C., & Zhang, Z.-K. (2015). Multi-linear interactive matrix factor-
869 ization. *Knowledge-Based Systems*, 85, 307–315.
- 870 Zhao-yang, W. (2010). Forecasting stock indexes based on a revised grey model
871 and the arma model. *CAAI Transactions on Intelligent Systems*, 3.

872 **Appendix A. Results**

873 This section displays results from Confusion matrices and Pearson correlation
874 graph. Due to space constraints we have attached additional results in appendix.

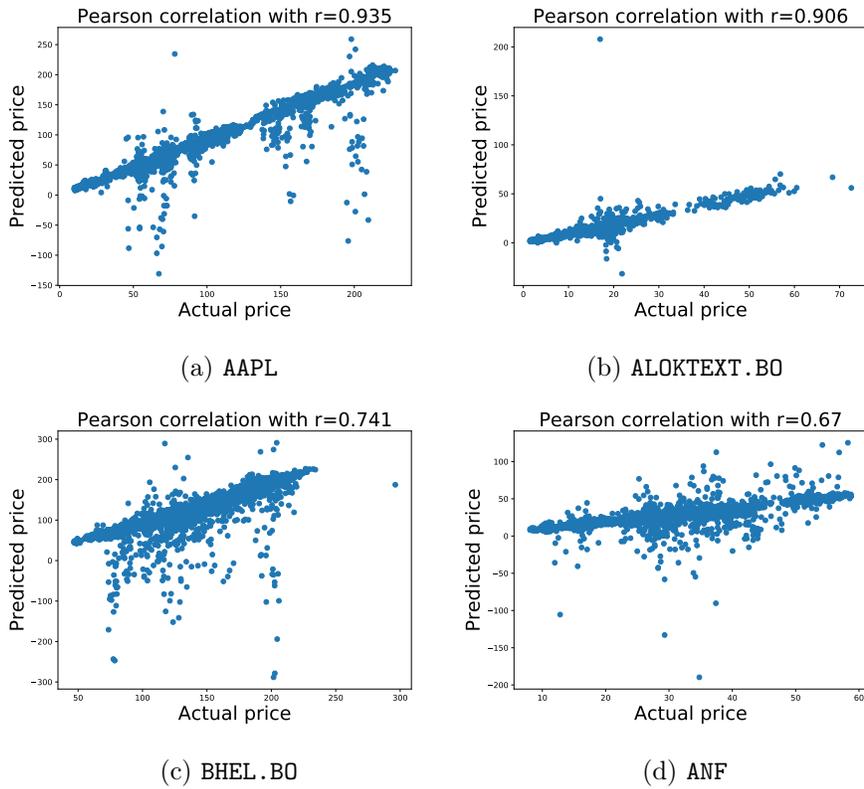
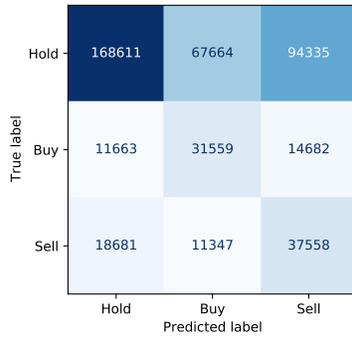
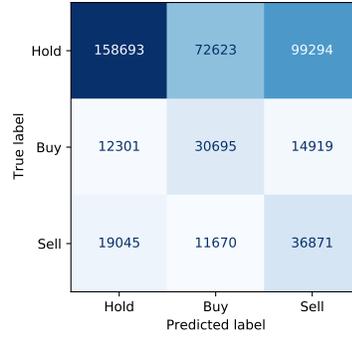


Figure A.4: Pearson correlation graph between ground truth adjusted close price and predicted one with DRDL (3 layers), during test phase, for four different stocks.

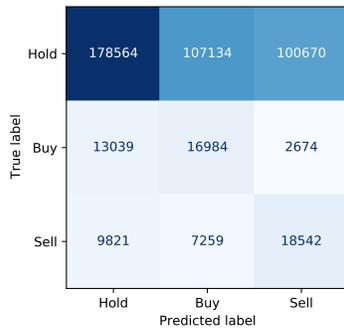
875



(a) DRDL (3 layers)



(b) DRDL (2 layers)



(c) DRDL (1 layer)



(d) MFNN



(e) CNN-TA



(f) LSTM

Figure A.5: Confusion matrices on stock trading classification task (averaged over days in test phase and over stocks) for DRDL with 1 to 3 layers, and deep learning techniques MFNN, CNN-TA and LSTM.