A Neural Stochastic Volatility Model

Rui Luo,[†] Weinan Zhang,[‡] Xiaojun Xu,[‡] Jun Wang[†]
[†]University College London
[‡]Shanghai Jiao Tong University
{r.luo,j.wang}@cs.ucl.ac.uk, {wnzhang,xuxj}@apex.sjtu.edu.cn

Abstract

In this paper, we show that the recent integration of statistical models with deep recurrent neural networks provides a new way of formulating volatility (the degree of variation of time series) models that have been widely used in time series analysis and prediction in finance. The model comprises a pair of complementary stochastic recurrent neural networks: the generative network models the joint distribution of the stochastic volatility process; the inference network approximates the conditional distribution of the latent variables given the observables. Our focus here is on the formulation of temporal dynamics of volatility over time under a stochastic recurrent neural network framework. Experiments on real-world stock price datasets demonstrate that the proposed model generates a better volatility estimation and prediction that outperforms mainstream methods, e.g., deterministic models such as GARCH and its variants, and stochastic models namely the MCMC-based stochvol as well as the Gaussian-process-based, on average negative log-likelihood.

Introduction

The volatility of the price movements reflects the ubiquitous uncertainty within financial markets. It is critical that the level of risk (aka, the degree of variation), indicated by volatility, is taken into consideration before investment decisions are made and portfolio are optimised (Hull 2006); volatility is substantially a key variable in the pricing of derivative securities. Hence, estimating and forecasting volatility is of great importance in branches of financial studies, including investment, risk management, security valuation and monetary policy making (Poon and Granger 2003).

Volatility is measured typically by employing the standard deviation of price change in a fixed time interval, such as a day, a month or a year. The higher the volatility is, the riskier the asset should be. One of the primary challenges in designing volatility models is to identify the existence of latent stochastic processes and to characterise the underlying dependences or interactions between variables within a certain time span. A classic approach has been to handcraft the characteristic features of volatility models by imposing assumptions and constraints, given prior knowledge and observations. Notable examples include autoregressive

conditional heteroscedasticity (ARCH) model (Engle 1982) and the extension, generalised ARCH (GARCH) (Bollerslev 1986), which makes use of autoregression to capture the properties of time-varying volatility within many time series. As an alternative to the GARCH model family, the class of stochastic volatility (SV) models specify the variance to follow some latent stochastic process (Hull and White 1987). Heston (Heston 1993) proposed a continuous-time model with the volatility following an Ornstein-Uhlenbeck process and derived a closed-form solution for options pricing. Since the temporal discretisation of continuous-time dynamics sometimes leads to a deviation from the original trajectory of system, those continuous-time models are seldom applied in forecasting. For practical purposes of forecasting, the canonical model (Jacquier, Polson, and Rossi 2002; Kim, Shephard, and Chib 1998) formulated in a discrete-time fashion for regularly spaced data such as daily prices of stocks is of great interest. While theoretically sound, those approaches require strong assumptions which might involve detailed insight of the target sequences and are difficult to determine without a thorough inspection.

In this paper, we take a fully data driven approach and determine the configurations with as few exogenous input as possible, or even purely from the historical data. We propose a neural network re-formulation of stochastic volatility by leveraging stochastic models and recurrent neural networks (RNNs). In inspired by the work from Chung et al. (Chung et al. 2015) and Fraccaro et al. (Fraccaro et al. 2016), the proposed model is rooted in variational inference and equipped with the latest advances of stochastic neural networks. The model inherits the fundamentals of SV model and provides a general framework for volatility modelling; it extends previous sequential frameworks with autoregressive and bidirectional architecture and provide with a more systematic and volatility-specific formulation on stochastic volatility modelling for financial time series. We presume that the latent variables follow a Gaussian autoregressive process, which is then utilised to model the variance process. Our neural network formulation is essentially a general framework for volatility modelling, which covers two major classes of volatility models in financial study as the special cases with specific weights and activations on neurons.

Experiments with real-world stock price datasets are performed. The result shows that the proposed model produces

more accurate estimation and prediction, outperforming various widely-used deterministic models in the GARCH family and several recently proposed stochastic models on average negative log-likelihood; the high flexibility and rich expressive power are validated.

Related Work

A notable framework for volatility is autoregressive conditional heteroscedasticity (ARCH) model (Engle 1982): it can accurately identify the characteristics of time-varying volatility within many types of time series. Inspired by ARCH model, a large body of diverse work based on stochastic process for volatility modelling has emerged (Bollerslev, Engle, and Nelson 1994). Bollerslev (Bollerslev 1986) generalised ARCH model to the generalised autoregressive conditional heteroscedasticity (GARCH) model in a manner analogous to the extension from autoregressive (AR) model to autoregressive moving average (ARMA) model by introducing the past conditional variances in the current conditional variance estimation. Engle and Kroner (Engle and Kroner 1995) presented theoretical results on the formulation and estimation of multivariate GARCH model within simultaneous equations systems. The extension to multivariate model allows the covariance to present and depend on the historical information, which are particularly useful in multivariate financial models. An alternative to the conditionally deterministic GARCH model family is the class of stochastic volatility (SV) models, which first appeared in the theoretical finance literature on option pricing (Hull and White 1987). The SV models specify the variance to follow some latent stochastic process such that the current volatility is no longer a deterministic function even if the historical information is provided. As an example, Heston's model (Heston 1993) characterises the variance process as a Cox-Ingersoll-Ross process driven by a latent Wiener process. While theoretically sound, those approaches require strong assumptions which might involve complex probability distributions and non-linear dynamics that drive the process. Nevertheless, empirical evidences have confirmed that volatility models provide accurate prediction (Andersen and Bollerslev 1998) and models such as ARCH and its descendants/variants have become indispensable tools in asset pricing and risk evaluation. Notably, several models have been recently proposed for practical forecasting tasks: Kastner et al. (Kastner and Frühwirth-Schnatter 2014) implemented the MCMC-based framework *stochvol* where the ancillarity-sufficiency interweaving strategy (ASIS) is applied for boosting MCMC estimation of stochastic volatility; Wu et al. (Wu, Hernández-Lobato, and Ghahramani 2014) designed the *GP-Vol*, a non-parametric model which utilises Gaussian processes to characterise the dynamics and jointly learns the process and hidden states via online inference algorithm. Despite the fact that it provides us with a practical approach towards stochastic volatility forecasting, both models require a relatively large volume of samples to ensure the accuracy, which involves very expensive sampling routine at each time step. Another drawback is that those models are incapable to handle the forecasting task for multivariate time series.

On the other hand, deep learning (LeCun, Bengio, and Hinton 2015; Schmidhuber 2015) that utilises nonlinear structures known as deep neural networks, powers various applications. It has triumph over pattern recognition challenges, such as image recognition (Krizhevsky, Sutskever, and Hinton 2012), speech recognition (Chorowski et al. 2015), machine translation (Bahdanau, Cho, and Bengio 2014) to name a few.

Time-dependent neural networks models include RNNs with neuron structures such as long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997), bidirectional RNN (BRNN) (Schuster and Paliwal 1997), gated recurrent unit (GRU) (Cho et al. 2014) and attention mechanism (Xu et al. 2015). Recent results show that RNNs excel for sequence modelling and generation in various applications (van den Oord, Kalchbrenner, and Kavukcuoglu 2016; Cho et al. 2014; Xu et al. 2015). However, despite its capability as non-linear universal approximator, one of the drawbacks of neural networks is its deterministic nature. Adding latent variables and their processes into neural networks would easily make the posteriori computationally intractable. Recent work shows that efficient inference can be found by variational inference when hidden continuous variables are embedded into the neural networks structure (Kingma and Welling 2013; Rezende, Mohamed, and Wierstra 2014). Some early work has started to explore the use of variational inference to make RNNs stochastic: Chung et al. (Chung et al. 2015) defined a sequential framework with complex interacting dynamics of coupling observable and latent variables whereas Fraccaro et al. (Fraccaro et al. 2016) utilised heterogeneous backward propagating layers in inference network according to its Markovian properties.

In this paper, we apply the stochastic neural networks to solve the volatility modelling problem. In other words, we model the dynamics and stochastic nature of the degree of variation, not only the mean itself. Our neural network treatment of volatility modelling is a general one and existing volatility models (e.g., the Heston and GARCH models) are special cases in our formulation.

Preliminaries: Volatility Models

Volatility models characterise the dynamics of volatility processes, and help estimate and forecast the fluctuation within time series. As it is often the case that one seeks for prediction on quantity of interest with a collection of historical information at hand, we presume the conditional variance to have dependency – either deterministic or stochastic – on history, which results in two categories of volatility models.

Deterministic Volatility Models: the GARCH Model Family

The GARCH model family comprises various linear models that formulate the conditional variance at present as a linear function of observations and variances from the past. Bollerslev's extension (Bollerslev 1986) of Engle's primitive ARCH model (Engle 1982), referred as generalised ARCH (GARCH) model, is one of the most well-studied

and widely-used volatility models:

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{p} \alpha_i x_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2, \qquad (1)$$

$$x_t \sim \mathcal{N}(0, \sigma_t^2), \qquad (2)$$

where Eq. (2) represents the assumption that the observation x_t follows from the Gaussian distribution with mean 0 and variance σ_t^2; the (conditional) variance σ_t^2 is fully determined by a linear function (Eq. (1)) of previous observations $\{x_{<t}\}$ and variances $\{\sigma_{<t}^2\}$. Note that if $q = 0$ in Eq. (1), GARCH model degenerates to ARCH model.

Various variants have been proposed ever since. Glosten, Jagannathan and Runkle (Glosten, Jagannathan, and Runkle 1993) extended GARCH model with additional terms to account for asymmetries in the volatility and proposed GJR-GARCH model; Zakoian (Zakoian 1994) replaced the quadratic operators with absolute values, leading to threshold ARCH/GARCH (TARCH) models. The general functional form is formulated as

$$\sigma_t^d = \alpha_0 + \sum_{i=1}^{p} \alpha_i |x_{t-i}|^d + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^d$$
$$+ \sum_{k=1}^{o} \gamma_k |x_{t-k}|^d I\{x_{t-k} < 0\}, \qquad (3)$$

where $I\{x_{t-k} < 0\}$ denotes the indicator function: $I\{x_{t-k} < 0\} = 1$ if $x_{t-k} < 0$, and 0 otherwise, which allows for asymmetric reactions of volatility in terms of the sign of previous observations.

Many variants of the GARCH model can be expressed by assigning values to parameters p, o, q, d in Eq. (3):

1. ARCH(p): $p \in \mathbb{N}^+$; $q \equiv 0$; $o \equiv 0$; $d \equiv 2$
2. GARCH(p, q): $p \in \mathbb{N}^+$; $q \equiv 0$; $o \equiv 0$; $d \equiv 2$
3. GJR-GARCH(p, o, q): $p \in \mathbb{N}^+$; $q \in \mathbb{N}^+$; $o \in \mathbb{N}^+$; $d \equiv 2$
4. AVARCH(p): $p \in \mathbb{N}^+$; $q \equiv 0$; $o \equiv 0$; $d \equiv 2$
5. AVGARCH(p, q): $p \in \mathbb{N}^+$; $q \in \mathbb{N}^+$; $o \equiv 0$; $d \equiv 2$
6. TARCH(p, o, q): $p \in \mathbb{N}^+$; $q \in \mathbb{N}^+$; $o \in \mathbb{N}^+$; $d \equiv 1$

Another fruitful specification shall be Nelson's exponential GARCH (EGARCH) model (Nelson 1991), which instead formulates the dependencies in log-variance $\log(\sigma_t^2)$:

$$\log(\sigma_t^2) = \alpha_0 + \sum_{i=1}^{p} \alpha_i g(x_{t-i}) + \sum_{j=1}^{q} \beta_j \log(\sigma_{t-j}^2), \quad (4)$$

$$g(x_t) = \theta x_t + \gamma(|x_t| - \mathbb{E}[|x_t|]), \qquad (5)$$

where $g(x_t)$ (Eq. (5)) accommodates the asymmetric relation between observations and volatility changes. If we set $q \equiv 0$ in Eq. (4), the EGARCH(p, q) model degenerates to EARCH(p) model.

Stochastic Volatility Models

An alternative to the (conditionally) deterministic volatility models is the class of stochastic volatility (SV) models. First introduced in the theoretical finance literature, earliest SV models such as Hull and White's (Hull and White 1987) as well as Heston model (Heston 1993) are formulated by stochastic differential equations in a continuous-time fashion for analysis convenience. In particular, Heston model instantiates a continuous-time stochastic volatility model for univariate processes:

$$\mathrm{d}\sigma(t) = -\beta\sigma(t)\,\mathrm{d}t + \delta\,\mathrm{d}w^\sigma(t), \qquad (6)$$

$$\mathrm{d}x(t) = (\mu - 0.5\sigma^2(t))\,\mathrm{d}t + \sigma(t)\,\mathrm{d}w^x(t). \qquad (7)$$

where $x(t) = \log s(t)$ is the logarithm of stock price s_t at time t, $w^x(t)$ and $w^\sigma(t)$ represent two correlated Wiener processes and the correlation between $\mathrm{d}w^x(t)$ and $\mathrm{d}w^\sigma(t)$ is expressed as $\mathbb{E}[\mathrm{d}w^x(t) \cdot \mathrm{d}w^\sigma(t)] = \rho\,\mathrm{d}t$.

For practical use, empirical versions of the SV model, typically formulated in a discrete-time fashion, are of great interest. The canonical model (Jacquier, Polson, and Rossi 2002; Kim, Shephard, and Chib 1998) for regularly spaced data is formulated as

$$\log(\sigma_t^2) = \eta + \phi(\log(\sigma_{t-1}^2) - \eta) + z_t, \qquad (8)$$

$$z_t \sim \mathcal{N}(0, \sigma_z^2), \quad x_t \sim \mathcal{N}(0, \sigma_t^2). \qquad (9)$$

Equation (8) indicates that the (conditional) log-variance $\log(\sigma_t^2)$ depends on not only the historical log-variances $\{\log(\sigma_t^2)\}$ but a latent stochastic process $\{z_t\}$. The latent process $\{z_t\}$ is, according to Eq. (9), white noise process with i.i.d. Gaussian variables.

Notably, the volatility σ_t^2 is no longer conditionally deterministic (i.e. deterministic given the complete history $\{\sigma_{<t}^2\}$) but to some extent stochastic in the setting of SV models: Heston model involves two correlated continuous-time Wiener processes while the canonical model is driven by a discrete-time Gaussian white-noise process.

Volatility Models in a General Form

Hereafter we denote the sequence of observations as $\{x_t\}$ and the latent stochastic process as $\{z_t\}$. As seen in previous sections, the dynamics of volatility process $\{\sigma_t^2\}$ can be abstracted as

$$\sigma_t^2 = f(\sigma_{<t}^2, x_{<t}, z_{\le t}) = \Sigma^x(x_{<t}, z_{\le t}). \qquad (10)$$

The latter equality follows as we recursively substitute σ_τ^2 with $f(\sigma_{<\tau}^2, x_{<\tau}, z_{\le \tau})$ for all $\tau < t$. For models in the GARCH family, we discard $z_{\le t}$ in the specification of $\Sigma^x(x_{<t}, z_{\le t})$ (Eq. (10)); on the other hand, for the SV model, $x_{<t}$ is ignored instead. We can loosen the constraint that x_t is zero-mean to a time-varying mean $\mu^x(x_{<t}, z_{\le t})$ for more flexibility.

Recall that the latent stochastic process $\{z_t\}$ (Eq. (9)) in the SV model is an i.i.d. Gaussian white noise process. We may extend the white noise process to a more flexible one with inherent autoregressive dynamics: the mean $\mu^z(z_{<t})$ and variance $\Sigma^z(z_{<t})$ are functions of an autoregressive form on the historical values. Thus, the generalised model can be formulated as

$$z_t | z_{<t} \sim \mathcal{N}(\mu^z(z_{<t}), \Sigma^z(z_{<t})), \qquad (11)$$

$$x_t | x_{<t}, z_{\le t} \sim \mathcal{N}(\mu^x(x_{<t}, z_{\le t}), \Sigma^x(x_{<t}, z_{\le t})), \qquad (12)$$

where we have presumed that both the observation x_t and the latent variable z_t are normally distributed. Note that the autoregressive process degenerates to i.i.d. white noise process when $\mu^z(z_{<t}) \equiv 0$ and $\Sigma^z(z_{<t}) \equiv \sigma_z^2$. It should be emphasised that the purpose of reinforcing an autoregressive structure (11) of the latent variable z_t is that we believe such formulation fits better to real scenarios from financial aspect compared with the i.i.d. convention: the price fluctuation of a certain stock is the consequence of not only its own history but also the influence from the environment, e.g. its competitors, up/downstream industries, relevant companies in the market, etc. Such external influence is ever-changing and may preserve memory and hence hard to characterise if restricted to i.i.d. noise. The latent variable z_t with an autoregressive structure provides a possibility of decoupling the internal influential factors from the external ones, which we believe is the essence of introducing z_t.

Neural Stochastic Volatility Models

In this section, we establish the *neural stochastic volatility model* (NSVM) for volatility estimation and prediction.

Generating Observable Sequence

Recall that the observable variable x_t (Eq. (12)) and the latent variable z_t (Eq. (11)) are described by autoregressive models (as x_t also involves an exogenous input $z_{\leq t}$). Let $p_\Phi(x_t|x_{<t}, z_{\leq t})$ and $p_\Phi(z_t|z_{<t})$ denote the probability distributions of x_t and z_t at time t. The factorisation on the joint distributions of sequences $\{x_t\}$ and $\{z_t\}$ applies as follow:

$$p_\Phi(Z) = \prod_t p_\Phi(z_t|z_{<t})$$
$$= \prod_t \mathcal{N}(z_t; \mu_\Phi^z(z_{<t}), \Sigma_\Phi^z(z_{<t})), \qquad (13)$$

$$p_\Phi(X|Z) = \prod_t p_\Phi(x_t|x_{<t}, z_{\leq t})$$
$$= \prod_t \mathcal{N}(x_t; \mu_\Phi^x(x_{<t}, z_{\leq t}), \Sigma_\Phi^x(x_{<t}, z_{\leq t})), \quad (14)$$

where $X = \{x_t\}_{1:T}$ and $Z = \{z_t\}_{1:T}$ represents the sequences of observable and latent variables, respectively, whereas Φ stands for the collection of parameters of generative model. The unconditional generative model is defined as the joint distribution:

$$p_\Phi(X, Z) = \prod_t p_\Phi(x_t|x_{<t}, z_{\leq t}) p_\Phi(z_t|z_{<t}). \qquad (15)$$

It can be observed that the mean and variance are conditionally deterministic: given the historical information $\{z_{<t}\}$, the current mean $\mu_t^z = \mu_\Phi^z(z_{<t})$ and variance $\Sigma_t^z = \Sigma_\Phi^z(z_{<t})$ of z_t is obtained and hence the distribution $\mathcal{N}(z_t; \mu_t^z, \Sigma_t^z)$ of z_t is specified; after sampling z_t from the specified distribution, we incorporate $\{x_{<t}\}$ and calculate the current mean $\mu_t^x = \mu_\Phi^x(x_{<t}, z_{\leq t})$ and variance $\Sigma_t^x = \Sigma_\Phi^x(x_{<t}, z_{\leq t})$ of x_t and determine its distribution $\mathcal{N}(x_t; \mu_t^x, \Sigma_t^x)$ of x_t. It is natural and convenient to present such a procedure in a recurrent fashion because of its autoregressive nature. Since RNNs can essentially approximate arbitrary function of recurrent form, the means and variances,

which may be driven by complex non-linear dynamics, can be efficiently computed using RNNs.

The unconditional generative model consists of two pairs of RNN and multi-layer perceptron (MLP), namely $\text{RNN}_G^z/\text{MLP}_G^z$ for the latent variable and $\text{RNN}_G^x/\text{MLP}_G^x$ for the observable. We stack those two RNN/MLP pairs together according to the causal dependency between variables. The unconditional generative model is implemented as the *generative network*:

$$\{\mu_t^z, \Sigma_t^z\} = \text{MLP}_G^z(h_t^z; \Phi), \qquad (16)$$
$$h_t^z = \text{RNN}_G^z(h_{t-1}^z, z_{t-1}; \Phi), \qquad (17)$$
$$z_t \sim \mathcal{N}(\mu_t^z, \Sigma_t^z), \qquad (18)$$
$$\{\mu_t^x, \Sigma_t^x\} = \text{MLP}_G^x(h_t^x; \Phi), \qquad (19)$$
$$h_t^x = \text{RNN}_G^x(h_{t-1}^x, x_{t-1}, z_t; \Phi), \qquad (20)$$
$$x_t \sim \mathcal{N}(\mu_t^x, \Sigma_t^x), \qquad (21)$$

where h_t^z and h_t^x denote the hidden states of the corresponding RNNs. The MLPs map the hidden states of RNNs into the means and deviations of variables of interest. The collection of parameters Φ is comprised of the weights of RNNs and MLPs. NSVM relaxes the conventional constraint that the latent variable z_t is $\mathcal{N}(0, 1)$ in a way that z_t is no longer i.i.d noise but a time-varying signal from external process with self-evolving nature. As discussed above, this relaxation will benefit the effectiveness in real scenarios.

One should notice that when the latent variable z_t is obtained, e.g. by inference (see details in the next subsection), the conditional distribution $p_\Phi(X|Z)$ (Eq. (14)) will be involved in generating the observable x_t instead of the joint distribution $p_\Phi(X, Z)$ (Eq. (15)). This is essentially the scenario of predicting future values of the observable variable given its history. We will use the term "generative model" and will not discriminate the unconditional generative model or the conditional one as it can be inferred in context.

Inferencing the Latent Process

As the generative model involves the latent variable z_t, of which the true values are inaccessible even we have observed x_t, the marginal distribution $p_\Phi(X)$ becomes the key that bridges the model and the data. However, the calculation of $p_\Phi(X)$ itself or its complement, the posterior distribution $p_\Phi(Z|X)$, is often intractable as complex integrals are involved. We are unable to learn the parameters by differentiating the marginal log-likelihood $\log p_\Phi(X)$ or to infer the latent variables through the true posterior. Therefore, we consider instead a restricted family of tractable distributions $q_\Psi(Z|X)$, referred to as the approximate posterior family, as approximations to the true posterior $p_\Phi(Z|X)$ such that the family is sufficiently rich and of high capacity to provide good approximations.

It is straightforward to verify that given a sequence of observations $X = \{x_{1:T}\}$, for any $1 \leq t \leq T$, z_t is dependent on the entire observation sequences. Hence, we define the inference model with the spirit of mean-field approximation where the approximate posterior is Gaussian and the follow-

ing factorisation applies:

$$q_\Psi(Z|X) = \prod_{t=1}^{T} q_\Psi(z_t|z_{<t}, x_{1:T})$$

$$= \prod_t \mathcal{N}(z_t; \tilde{\mu}_\Psi^z(z_{<t}, x_{1:T}), \tilde{\Sigma}_\Psi^z(z_{<t}, x_{1:T})), \quad (22)$$

where $\tilde{\mu}_\Psi^z(z_{t-1}, x_{1:T})$ and $\tilde{\Sigma}_\Psi^z(z_{t-1}, x_{1:T})$ are functions of the given observation sequence $\{x_{1:T}\}$, representing the approximated mean and variance of the latent variable z_t; Ψ denotes the collection of parameters of inference model.

The neural network implementation of the model, referred to as the *inference network*, is designed to equip a cascaded architecture with an autoregressive RNN and a bidirectional RNN, where the bidirectional RNN incorporates both the forward and backward dependencies on the entire observations whereas the autoregressive RNN models the temporal dependencies on the latent variables:

$$\{\tilde{\mu}_t^z, \tilde{\Sigma}_t^z\} = \text{MLP}_I^z(\tilde{h}_t^z; \Psi), \quad (23)$$

$$\tilde{h}_t^z = \text{RNN}_I^z(\tilde{h}_{t-1}^z, z_{t-1}, [\tilde{h}_t^\rightarrow, \tilde{h}_t^\leftarrow]; \Psi), \quad (24)$$

$$\tilde{h}_t^\rightarrow = \text{RNN}_I^\rightarrow(\tilde{h}_{t-1}^\rightarrow, x_{t-1}; \Psi), \quad (25)$$

$$\tilde{h}_t^\leftarrow = \text{RNN}_I^\leftarrow(\tilde{h}_{t+1}^\leftarrow, x_{t+1}; \Psi), \quad (26)$$

$$z_t \sim \mathcal{N}(\tilde{\mu}_t^z, \tilde{\Sigma}_t^z; \Psi), \quad (27)$$

where $\tilde{h}_t^\rightarrow$ and $\tilde{h}_t^\leftarrow$ represent the hidden states of the forward and backward directions of the bidirectional RNN. The autoregressive RNN with hidden state $\tilde{h}_t^z$ takes the joint state $[\tilde{h}_t^\rightarrow, \tilde{h}_t^\leftarrow]$ of the bidirectional RNN and the previous value of z_{t-1} as input. The inference mean $\tilde{\mu}_t^z$ and variance $\tilde{\Sigma}_t^z$ is computed by an MLP from the hidden state $\tilde{h}_t^z$ of the autoregressive RNN. We use the subscript I instead of G to distinguish the architecture used in inference model in contrast to that of the generative model. It should be emphasised that the inference network will collaborates with the generative network on conditional generating procedure.

Algorithm 1 Recursive Forecasting
___
1: **loop**
2:      $\{z_{1:t}^{\langle s \rangle}\} \leftarrow$ draw S paths from $q(z_{1:t}|x_{1:t})$
3:      $\{z_{1:t+1}^{\langle s \rangle}\} \leftarrow$ extend $\{z_{1:t}^{\langle s \rangle}\}$ for 1 step via $p(z_{t+1}|z_{1:t})$
4:      $\hat{p}(x_{\tau+1}|x_{1:\tau}) \leftarrow 1/S \times \sum_s p(x_{\tau+1}|z_{1:\tau+1}^{\langle s \rangle}, x_{1:\tau})$
5:      $\hat{\sigma}_{t+1}^2 \leftarrow \text{var}\{\hat{x}_{t+1}^{1:S}\}, \{\hat{x}_{t+1}^{1:S}\} \sim \hat{p}(x_{\tau+1}|x_{1:\tau})$
6:      $\{x_{1:t+1}\} \leftarrow [\{x_{1:t}\}, x_{t+1}]$ with new x_{t+1}
7:      $t \leftarrow t + 1$, (optionally) retrain the model
___

Forecasting Observations in Future

For a volatility model to be practically applicable in forecasting, the generating procedure conditioning on the history is of essential interest. We start with 1-step-ahead prediction, which serves as building block of multi-step forecasting.

Given the historical observations $\{x_{1:T}\}$ up to time step T, 1-step-ahead prediction of either Σ_{T+1}^x or x_{T+1} is fully

depicted by the conditional predictive distribution:

$$p(x_{T+1}|x_{1:T}) = \int_z p(x_{T+1}|z_{1:T+1}, x_{1:T})$$
$$\cdot p(z_{T+1}|z_{1:T})p(z_{1:T}|x_{1:T}) \, dz, \quad (28)$$

where the distributions on the right-hand side refer to those in the generative model with the generative parameters Φ omitted. As the true posterior $p(z_{1:T}|x_{1:T})$ involved in Eq. (28) is intractable, the exact evaluation of conditional predictive distribution $p(x_{T+1}|x_{1:T})$ is difficult.

A straightforward solution is that we substitute the true posterior $p(z_{1:T}|x_{1:T})$ with the approximation $q(z_{1:T}|x_{1:T})$ (see Eq. (22)) and leverage $q(z_{1:T}|x_{1:T})$ to inference S sample paths $\{z_{1:T}^{\langle 1:S \rangle}\}$ of the latent variables according to the historical observations $\{x_{1:T}\}$. The approximate posterior from a well-trained model is presumed to be a good approximation to the truth; hence the sample paths shall be mimics of the true but unobservable path. We then extend the sample paths one step further from T to $T+1$ using the autoregressive generative distribution $p(z_{T+1}|z_{1:T})$ (see Eq. (13)). The conditional predictive distribution is thus approximated as

$$\hat{p}(x_{T+1}|x_{1:T}) \approx \frac{1}{S} \sum_s p(x_{T+1}|z_{1:T+1}^{\langle s \rangle}, x_{1:T}), \quad (29)$$

which is essentially a mixture of S Gaussians. In the case of multi-step forecasting, a common solution in practice is to perform a recursive 1-step-ahead forecasting routine with model updated as new observation comes in; the very same procedure can be applied except that more sample paths should be evaluated due to the accumulation of uncertainty. Algorithm 1 gives the detailed rolling scheme.

Experiment

In this section, we present the experiment on real-world stock price time series to validate the effectiveness and to evaluate the performance of the prosed model.

Dataset and Pre-processing

The raw dataset comprises 162 univariate time series of the daily closing stock price, chosen from China's A-shares and collected from 3 institutions. The choice is made by selecting those with earlier listing date of trading (from 2006 or earlier) and fewer suspension days (at most 50 suspension days within the entire period of observation), such that the undesired noises introduced by insufficient observation or missing values – highly influential on the performance but essentially irrelevant to the purpose of volatility modelling – can be reduced to the minimum. The raw price series is cleaned by aligning and removing abnormalities: we manually aligned the mismatched part and interpolated the missing value by stochastic regression imputation (Little and Rubin 2014) where the imputed value is drawn from a Gaussian distribution with mean and variance calculated by regression on the empirical value within a short interval of 20 recent days. The series is then transformed from actual prices s_t into log-returns $x_t = \log(s_t/s_{t-1})$ and normalised. Moreover, we combinatorically choose a predefined number d out

of 162 univariate log-return series and aggregate the selected series at each time step to form a d-dimensional multivariate time series, the choice of d is in accordance with the rank of correlation, e.g. $d = 6$ in our experiments. Theoretically, it leads to a much larger volume of data as $\binom{162}{6} > 2 \times 10^{10}$. Specifically, the actual dataset for training and evaluation comprises a collection of 2000 series of d-dimensional normalised log-return vectors of length 2570 (~ 7 years) with no missing values. We divide the whole dataset into two subsets for training and testing along the time axis: the first 2000 time steps of each series have been used as training samples whereas the rest 570 steps of each series as the test samples.

Baselines

We select several deterministic volatility models from the GARCH family as baselines:

1. Quadratic models
   - ARCH(1); GARCH(1,1); GJR-GARCH(1,1,1);
2. Absolute value models
   - AVARCH(1); AVGARCH(1,1); TARCH(1,1,1);
3. Exponential models.
   - EARCH(1); EGARCH(1,1);

Moreover, two stochastic volatility models are compared:

1. MCMC volatility model: *stochvol*;
2. Gaussian process volatility model *GP-Vol*.

For the listed models, we retrieve the authors' implementations or tools: stochvol[1], GP-Vol[2] (the hyperparameters are chosen as suggested in (Wu, Hernández-Lobato, and Ghahramani 2014)) and implement the models, such as GARCH, EGARCH, GJR-GARCH, etc., based on several widely-used packages[3][4][5] for time series analysis. All baselines are evaluated in terms of the negative log-likelihood on the test samples, where 1-step-ahead forecasting is carried out in a recursive fashion similar to Algorithm 1.

Model Implementation

In our experiments, we predefine the dimensions of observable variables to be $\dim x_t = 6$ and the latent variables $\dim z_t = 4$. Note that the dimension of the latent variable is smaller than that of the observable, which allows us to extract a compact representation. The NSVM implementation in our experiments is composed of two neural networks, namely the generative network (see Eq. (16)-(21)) and inference network (see Eq. (23)-(27)). Each RNN module contains one hidden layer of size 10 with GRU cells; MLP modules are 2-layered fully-connected feedforward networks, where the hidden layer is also of size 10 whereas the output layer splits into two equal-sized sublayers with different activation functions: one applies exponential function to ensure the non-negativity for variance while the other uses

[1] https://cran.r-project.org/web/packages/stochvol

[2] http://jmhl.org

[3] https://pypi.python.org/pypi/arch/4.0

[4] https://www.kevinsheppard.com/MFE_Toolbox

[5] https://cran.r-project.org/web/packages/fGarch

linear function to calculate mean estimates. Thus MLP_I^z's output layer is of size $4 + 4$ for $\{\tilde{\mu}^z, \tilde{\Sigma}^z\}$ whereas the size of MLP_G^x's output layer is $6 + 6$ for $\{\mu^x, \Sigma^x\}$. During the training phase, the inference network is connected with the conditional generative network (see, Eq. (16)-(18)) to establish a bottleneck structure, the latent variable z_t inferred by variational inference (Kingma and Welling 2013; Rezende, Mohamed, and Wierstra 2014) follows a Gaussian approximate posterior; the size of sample paths is set to $S = 100$. The parameters of both networks are jointly learned, including those for the prior. We introduce Dropout (Srivastava et al. 2014) into each RNN modules and impose $L2$-norm on the weights of MLP modules as regularistion to prevent overshooting; Adam optimiser (Kingma and Ba 2014) is exploited for fast convergence; exponential learning rate decay is adopted to anneal the variations of convergence as time goes. Two covariance configurations are adopted: 1. we stick with diagonal covariance matrices configurations; 2. we start with diagonal covariance and then apply rank-1 perturbation (Rezende, Mohamed, and Wierstra 2014) during fine-tuning until training is finished. The recursive 1-step-ahead forecasting routine illustrated as Algorithm 1 is applied in the experiment for both training and test phase: during the training phase, a single NSVM is trained, at each time step, on the entire training samples to learn a holistic dynamics, where the latent shall reflect the evolution of environment; in the test phase, on the other hand, the model is optionally retrained, at every 20 time steps, on each particular input series of the test samples to keep track on the specific trend of that series. In other words, the trained NSVM predicts 20 consecutive steps before it is retrained using all historical time steps of the input series at present. Correspondingly, all baselines are trained and tested at every time step of each univariate series using standard calibration procedures. The negative log-likelihood on test samples has been collected for performance assessment. We train the model on a single-GPU (Titan X Pascal) server for roughly two hours before it converges to a certain degree of accuracy on the training samples. Empirically, the training phase can be processed on CPU in reasonable time, as the complexity of the model as well as the size of parameters is moderate.

Result and Discussion

The performance of NSVM and baselines is listed for comparison in Table 1: the performance on the first 10 individual stocks (chosen in alphacetical order but anonymised here) and the average score on all 162 stocks are reported in terms of negative log-likelihood (NLL) measure. The result shows that NSVM has achieved higher accuracy over the baselines on the task of volatility modelling and forecasting on NLL, which validates the high flexibility and rich expressive power of NSVM for volatility modelling and forecasting. In particular, NSVM with rank-1 perturbation (referred to as NSVM-corr in Table 1) beats all other models in terms of NLL, while NSVM with diagonal covariance matrix (i.e. NSVM-diag) outperforms GARCH(1,1) on 142 out of 162 stocks. Although the improvement comes at the cost of longer training time before convergence, it can be mitigated by applying parallel computing techniques as well

Table 1: The performance of the proposed model and the baselines in terms of negative log-likelihood (NLL) evaluated on the test samples of real-world stock price time series: each row from 1 to 10 lists the average NLL for a specific individual stock; the last row summarises the average NLL of the entire test samples of all 162 stocks.

Stock	NSVM-corr	NSVM-diag	ARCH	GARCH	GJR	AVARCH	AVGCH	TARCH	EARCH	EGARCH	stochvol	GP-Vol
1	**1.11341**	1.42816	1.36733	1.60087	1.60262	1.34792	1.57115	1.58156	1.33528	1.53651	1.39638	1.56260
2	**1.04058**	1.28639	1.35682	1.63586	1.59978	1.32049	1.46016	1.45951	1.35758	1.52856	1.37080	1.47025
3	**1.03159**	1.32285	1.37576	1.44640	1.45826	1.34921	1.44437	1.45838	1.33821	1.41331	1.25928	1.48203
4	**1.06467**	1.32964	1.38872	1.45215	1.43133	1.37418	1.44565	1.44371	1.35542	1.40754	1.36199	1.32451
5	**0.96804**	1.22451	1.39470	1.31141	1.30394	1.37545	1.28204	1.27847	1.37697	1.28191	1.16348	1.41417
6	**0.96835**	1.23537	1.44126	1.55520	1.57794	1.39190	1.47442	1.47438	1.36163	1.48209	1.15107	1.24458
7	**1.13580**	1.43244	1.36829	1.65549	1.71652	1.32314	1.50407	1.50899	1.29369	1.64631	1.42043	1.19983
8	**1.03752**	1.26901	1.39010	1.47522	1.51466	1.35704	1.44956	1.45029	1.34560	1.42528	1.26289	1.47421
9	**0.95157**	1.15896	1.42636	1.32367	1.24404	1.42047	1.35427	1.34465	1.42143	1.32895	1.12615	1.35478
10	**0.99105**	1.13143	1.36919	1.55220	1.29989	1.24032	1.06932	1.04675	23.35983	1.20704	1.32947	1.18123
AVG	**1.18354**	1.23521	1.27062	1.27051	1.28809	1.28827	1.27754	1.29010	1.33450	1.36465	1.27098	1.34751

(a) The volatility forecasting for Stock 37

(b) The volatility forecasting for Stock 82
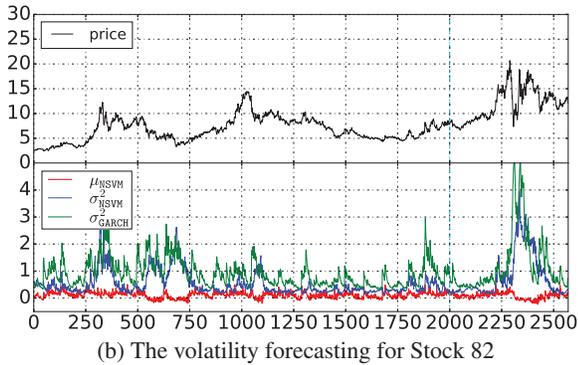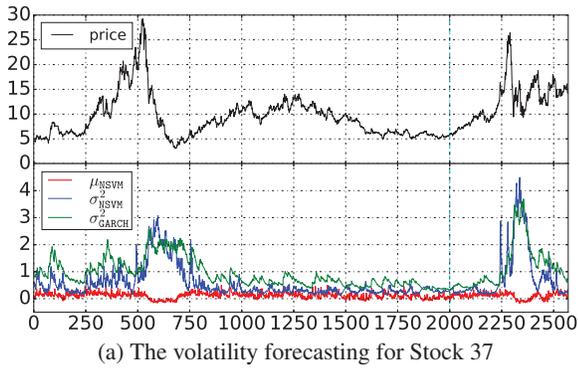
Figure 1: Case studies of volatility forecasting.

as more advanced network architecture or training methods. Apart from the higher accuracy NSVM obtained, it provides us with a rather general framework to generalise univariate time series models of any specific functional form to the corresponding multivariate cases by extending network dimensions and manipulating the covariance matrices. A case study on real-world financial datasets is illustrated in Fig. 1.

NSVM shows higher sensibility on drastic changes and better stability on moderate fluctuations: the response of NSVM in Fig. 1a is more stable in $t \in [1600, 2250]$, the period of moderate price fluctuation; while for drastic price change at $t = 2250$, the model responds with a sharper spike compared with the quadratic GARCH model. Furthermore, NSVM demonstrates the inherent non-linearity in both Fig. 1a and 1b: at each time step within $t \in [1000, 2000]$, the model quickly adapts to the current fluctuation level whereas GARCH suffers from a relatively slower decay from the previous influences. The cyan vertical line at $t = 2000$ splits the training samples and test samples. We show only one instance within our dataset due to the limitation of pages, the performance of other instances are similar.

Conclusion

In this paper, we proposed a new volatility model, referred to as NSVM, for volatility estimation and prediction. We integrated statistical models with deep neural networks, leveraged the characteristics of each model, organised the dependences between random variables in the form of graphical models, implemented the mappings among variables and parameters through RNNs and MLPs, and finally established a powerful stochastic recurrent model with universal approximation capability. The proposed architecture comprises a pair of complementary stochastic neural networks: the generative network and inference network. The former models the joint distribution of the stochastic volatility process with both observable and latent variables of interest; the latter provides with the approximate posterior i.e. an analytical approximation to the (intractable) conditional distribution of the latent variables given the observable ones. The parameters (and consequently the underlying distributions) are learned (and inferred) via variational inference, which maximises the lower bound for the marginal log-likelihood of the observable variables. NSVM has presented higher accuracy on the task of volatility modelling and forecasting on real-world financial datasets, compared with various widely-used models, such as GARCH, EGARCH, GJR-GARCH, TARCH in the GARCH family, MCMC-based stochvol model as well as Gaussian process volatility model GP-Vol. Future work on NSVM would be to investigate the modelling of time series with non-Gaussian residual distributions, in particular the heavy-tailed distributions e.g. Log-Normal $\log \mathcal{N}$ and Student's t-distribution.

References

Andersen, T. G., and Bollerslev, T. 1998. Answering the skeptics: Yes, standard volatility models do provide accurate forecasts. *International economic review* 885–905.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *CoRR* abs/1409.0473.

Bollerslev, T.; Engle, R. F.; and Nelson, D. B. 1994. Arch models. *Handbook of econometrics* 4:2959–3038.

Bollerslev, T. 1986. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics* 31(3):307–327.

Cho, K.; van Merrienboer, B.; Gülçehre, Ç.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Moschitti, A.; Pang, B.; and Daelemans, W., eds., *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, 1724–1734. ACL.

Chorowski, J.; Bahdanau, D.; Serdyuk, D.; Cho, K.; and Bengio, Y. 2015. Attention-based models for speech recognition. In Cortes et al. (2015), 577–585.

Chung, J.; Kastner, K.; Dinh, L.; Goel, K.; Courville, A. C.; and Bengio, Y. 2015. A recurrent latent variable model for sequential data. In Cortes et al. (2015), 2980–2988.

Cortes, C.; Lawrence, N. D.; Lee, D. D.; Sugiyama, M.; and Garnett, R., eds. 2015. *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada.*

Engle, R. F., and Kroner, K. F. 1995. Multivariate simultaneous generalized arch. *Econometric theory* 11(01):122–150.

Engle, R. F. 1982. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the Econometric Society* 987–1007.

Fraccaro, M.; Sønderby, S. K.; Paquet, U.; and Winther, O. 2016. Sequential neural models with stochastic layers. In Lee, D. D.; Sugiyama, M.; von Luxburg, U.; Guyon, I.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, 2199–2207.

Glosten, L. R.; Jagannathan, R.; and Runkle, D. E. 1993. On the relation between the expected value and the volatility of the nominal excess return on stocks. *The journal of finance* 48(5):1779–1801.

Heston, S. L. 1993. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of financial studies* 6(2):327–343.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Hull, J., and White, A. 1987. The pricing of options on assets with stochastic volatilities. *The journal of finance* 42(2):281–300.

Hull, J. C. 2006. *Options, futures, and other derivatives*. Pearson Education India.

Jacquier, E.; Polson, N. G.; and Rossi, P. E. 2002. Bayesian analysis of stochastic volatility models. *Journal of Business & Economic Statistics* 20(1):69–87.

Kastner, G., and Frühwirth-Schnatter, S. 2014. Ancillarity-sufficiency interweaving strategy (ASIS) for boosting MCMC estimation of stochastic volatility models. *Computational Statistics & Data Analysis* 76:408–423.

Kim, S.; Shephard, N.; and Chib, S. 1998. Stochastic volatility: likelihood inference and comparison with arch models. *The review of economic studies* 65(3):361–393.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980.

Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes. *CoRR* abs/1312.6114.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In Bartlett, P. L.; Pereira, F. C. N.; Burges, C. J. C.; Bottou, L.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, 1106–1114.

LeCun, Y.; Bengio, Y.; and Hinton, G. E. 2015. Deep learning. *Nature* 521(7553):436–444.

Little, R. J., and Rubin, D. B. 2014. *Statistical analysis with missing data*. John Wiley & Sons.

Nelson, D. B. 1991. Conditional heteroskedasticity in asset returns: A new approach. *Econometrica: Journal of the Econometric Society* 347–370.

Poon, S.-H., and Granger, C. W. 2003. Forecasting volatility in financial markets: A review. *Journal of economic literature* 41(2):478–539.

Rezende, D. J.; Mohamed, S.; and Wierstra, D. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, 1278–1286. JMLR.org.

Schmidhuber, J. 2015. Deep learning in neural networks: An overview. *Neural Networks* 61:85–117.

Schuster, M., and Paliwal, K. K. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing* 45(11):2673–2681.

Srivastava, N.; Hinton, G. E.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15(1):1929–1958.

van den Oord, A.; Kalchbrenner, N.; and Kavukcuoglu, K. 2016. Pixel recurrent neural networks. In Balcan, M., and Weinberger, K. Q., eds., *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, 1747–1756. JMLR.org.

Wu, Y.; Hernández-Lobato, J. M.; and Ghahramani, Z. 2014. Gaussian process volatility model. In Ghahramani, Z.; Welling, M.; Cortes, C.; Lawrence, N. D.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, 1044–1052.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A. C.; Salakhutdinov, R.; Zemel, R. S.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In Bach, F. R., and Blei, D. M., eds., *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, 2048–2057. JMLR.org.

Zakoian, J.-M. 1994. Threshold heteroskedastic models. *Journal of Economic Dynamics and control* 18(5):931–955.