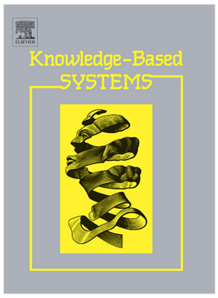
Accepted Manuscript



Sentiment-aware volatility forecasting

Frank Z. Xing, Erik Cambria, Yue Zhang

PII: S0950-7051(19)30154-6
DOI: https://doi.org/10.1016/j.knosys.2019.03.029
Reference: KNOSYS 4725

To appear in: *Knowledge-Based Systems*

Received date : 17 October 2018
Revised date : 17 February 2019
Accepted date : 24 March 2019

Please cite this article as: F.Z. Xing, E. Cambria and Y. Zhang, Sentiment-aware volatility forecasting, *Knowledge-Based Systems* (2019), https://doi.org/10.1016/j.knosys.2019.03.029

Sentiment-Aware Volatility Forecasting

Frank Z. Xing[a], Erik Cambria[a,*], Yue Zhang[b]

[a] School of Computer Science & Engineering, Nanyang Technological University, Singapore
[b] Institute of Advanced Technology, Westlake University, China

Abstract

Recent advances in the integration of deep recurrent neural networks and statistical inferences have paved new avenues for joint modeling of moments of random variables, which is highly useful for signal processing, time series analysis, and financial forecasting. However, introducing explicit knowledge as exogenous variables has received little attention. In this paper, we propose a novel model termed **s**entiment-**a**ware **v**olatil**i**ty foreca**s**t**ing** (SAVING), which incorporates market sentiment for stock return fluctuation prediction. Our framework provides an ensemble of symbolic and sub-symbolic AI approaches, that is, including grounded knowledge into a connectionist neural network. The model aims at producing a more accurate estimation of temporal variances of asset returns by better capturing the bi-directional interaction between movements of asset price and market sentiment. The interaction is modeled using Variational Bayes via the data generation and inference operations. We benchmark our model with 9 other popular ones in terms of the likelihood of forecasts given the observed sequence. Experimental results suggest that our model not only outperforms pure statistical models, e.g., GARCH and its variants, Gaussian-process volatility model, but also outperforms the state-of-the-art autoregressive deep neural nets architectures, such as the variational recurrent neural network and the neural stochastic volatility model.

Keywords: Volatility modeling; Sentiment knowledge; Time series analysis; Variational neural networks; Financial text mining.

*Corresponding author
Email addresses: `zxing001@ntu.edu.sg` (Frank Z. Xing), `cambria@ntu.edu.sg` (Erik Cambria), `yue.zhang@wias.org.cn` (Yue Zhang)

1. Introduction

Moments of asset returns carry important information in financial decision making. For example, the expected returns are calculated as the mean of return series, and volatility is measured by a covariance matrix over assets. For simplicity, the classic approach to portfolio management [1] (also called the mean-variance analysis framework) ignores higher moments and only considers the first two orders of moments of asset returns to diversify the investment. Under this framework, the constrained utility maximization problem is formulated as a quadratic optimization problem to maximize expected return and minimize the portfolio risk at the same time. Modern ways of derivatives pricing also heavily rely on the concepts of asset price and volatility, such as in the Black-Scholes model [2, 3]. Among these concepts, volatility is believed to be comparatively more difficult to approximate due to its complicated temporal dependency structure.

Traditional econometric models, such as the generalized autoregressive conditional heteroscedasticity (GARCH) group [4], formulate the time-varying volatility as a deterministic linear function of the past variable observations and its lagged items. However, financial data are characterized by chaotic behaviors and a poor signal-to-noise ratio. Observations suggest that linear modeling of volatility tends to be unstable and overfits to local randomness. Another method to model volatility in a deterministic generative manner is using a latent stochastic process as prior [5], hence called "stochastic volatility".

Recently, using deep recurrent neural networks (RNN) for sequential modeling has become popular. Successful applications of deep RNN have been reported on hand-writing recognition and speech synthesis. Compared to previous econometric research, deep RNN takes a data-driven approach to implicitly learn the approximation function. Therefore, it is claimed to have extra expressive power to capture the non-linear variation of volatility [6, 7]. The variational RNN (VRNN) is a hybrid of variational autoencoder and RNN which can be naturally used for joint modeling of a stochastic variable with its mean and variance [7], where the conditional distribution of the variable is generated from a Gaussian process determined by latent variable states. The neural stochastic volatility model (NSVM) further extends VRNN with autoregressive architecture for the hidden state, bi-directional architecture for variable encoding, and stochastic sampling techniques [8] to formalize a general form volatility model.

Both VRNN [7] and NSVM [8] take information only from the past observations. In the context of predicting expected stock returns and its volatility, the input information will be the historical prices and return sequences. One major concern of directly applying VRNN or NSVM is that, the role of investors or market participants is absent in their model settings [9]. In the real world, unfortunately, asset price fluctuations can be driven by events [10] and market sentiment [11, 12], or even irrational or collective actions that happens for no obvious reason. Implied volatility is thus very sensitive to the news and social media response [13, 14, 15, 16]. Omitting exogenous variables can thus be a defect in the attempt of constructing such a forecasting system.

To address this issue, we propose to extend VRNN volatility modeling by integrating sentiment signals from social media data. Our model is termed sentiment-aware volatility forecasting (SAVING). In this model, asset-specific sentiment time series are generated with the support of both symbolic AI and sub-symbolic AI methods. After being aligned with the return series, the sentiment variable becomes a component of both hidden states and the latent random variable for joint distribution modeling. Our contributions can be summarized as follows:

1. Joint-modeling of asset returns and market sentiment in a variational recurrent neural framework, instead of simple concatenation and normalization of the two heterogeneous

sources of knowledge.

2. Providing an interface for deep neural models to acquire explainable sentiment information from knowledge bases, bridging the gap of knowledge-based systems and pure machine learning systems.

Experimental results show that the SAVING model can achieve an improved performance on the task of volatility forecasting compared with the state-of-the-art recurrent neural models. In fact, on our dataset, the difference between the previously proposed recurrent neural models and the deterministic linear models are not significant, though they all seem to be better than naïve architectures with no effort made on adapting data features. We believe the room for improvement with an autoregressive model is limited and attribute the merit of the SAVING model to the fact that it effectively incorporates market sentiment to the predictive process.

The remainder of the article is organized as follows: Section 2 provides background for the compared methods, including econometric models and the VRNN model; Section 3 shows how the polarity score for each message is computed with the help of a knowledge base and the sentiment aggregation/quantization process to form a discrete sentiment time series; Section 4 describes the variable operations in the SAVING model; next, Section 5 reports and discusses experimental results; finally, Section 6 summarizes related research and Section 7 concludes the article with future work.

2. Background

In this section we provide some background about our baselines, including the GARCH model, the state-of-the-art VRNN model, and its variants such as NSVM.

2.1. Linear Volatility Modeling

Time-varying variance is a common phenomenon for financial time series, that is, strong fluctuations are clustered in certain time periods. Constant-variance models, e.g., the Autoregressive Moving Average (ARMA) model are not suitable for modeling such time series. Consequently, deterministic linear modeling of volatility is proposed. The GARCH model [4] is one of the most recognized among them, which models a time series x_t by a Gaussian process and its time-varying variance $\sigma^2_{x,t}$, as in Eq. (1) and Eq. (2):

$$\sigma^2_{x,t} = \alpha_0 + \sum_{i=1}^{p} \alpha_i x^2_{t-i} + \sum_{j=1}^{q} \beta_j \sigma^2_{t-j}, \tag{1}$$

$$x_t \sim \mathcal{N}(0, \sigma^2_{x,t}). \tag{2}$$

where p is the moving average order and q is the autoregressive order. Together, they characterize the number of parameters a GARCH model would have. Since each variable x_t is sampled from a local Gaussian distribution with zero mean, the residuals will always follow $\epsilon_t^2 \equiv x_t^2$. In this context, Eq. (1) is de facto an ARMA model of $\sigma^2_{x,t}$.

Variants of the GARCH model include ARCH [17], where autoregressive coefficients $\boldsymbol{\beta}$ are set to zero; EGARCH [18], where $\log(\sigma^2_{x,t})$ is used instead of $\sigma^2_{x,t}$ and a bias is imposed on x_t to address asymmetric volatility; GJR-GARCH [19], where asymmetric volatility is expressed by adding an indicator function of the sign of x_{t-1}, namely adding $\delta x^2_{t-1} I_{t-1}$ to the right side of Eq. (1) where $I_{t-1} = 0$ if $x_{t-1} \geq 0$ and $I_{t-1} = 1$ if $x_{t-1} < 0$.

We reconsider random variable x_t in Eq. (1) by decomposing it to a deterministic time-dependent $\sigma_{x,t}$ and a latent *standard* Gaussian process $z_t \sim \mathcal{N}(0,1)$:

$$x_t = \sigma_{x,t} \cdot z_t. \tag{3}$$

This decomposition inverses the idea of the reparameterization trick [20] to introduce z_t, though the stochastic item z_t is eliminated from the GARCH model by taking square. However, in stochastic volatility models z_t can be a separate item where x_t is drawn. Subsequently, a new set of parameters $\boldsymbol{\gamma}$ may be introduced. In real-world data, though, parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ may have more complicated form. Therefore, a general form volatility model can be abstracted as Eq. (4),

$$\sigma_{x,t}^2 = f_{[\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}]}(\sigma_{x,<t}^2, x_{<t}, z_{\leq t}) \tag{4}$$

where $\sigma_{x,<t}^2$ can be further eliminated [8]. In this case, the volatility is specified from this process:

1. generation of autoregressive latent process $z_{\leq t}$;
2. generation of past observations $x_{<t} = g(z_{\leq t})$;
3. generation of $\sigma_{x,t}^2$.

Note that the analytical solution of function f in Eq. (4) may not be easy to specify as the number of parameters grows large. Fortunately, recent advances in deep learning provide us a new way to parameterize f.

2.2. VRNN for Sequence Modeling

Suppose we have a sequence of observations $\boldsymbol{x} = (x_1, x_2, ..., x_t)$, a basic RNN learns the parameters θ of a neural network f and keeps updating its hidden state h_t as in Eq. (5):

$$h_t = f_\theta(x_t, h_{t-1}). \tag{5}$$

The neural network f_θ may consist of any neuron structures, such as the long short-term memory (LSTM) [21] and gated recurrent unit (GRU) [22]. In a standard RNN [23], the computation follows Eq. (6) and Eq. (7):

$$x_t = W^{ho}h_{t-1} + b^o \tag{6}$$
$$h_t = \tanh(W^{hh}h_{t-1} + W^{hx}x_t + b^h) \tag{7}$$

where W^{**} are transition matrices spanning timesteps and b^* are biases that define θ. A high-level illustration of the computation process is provided in Figure 1.

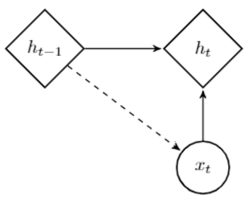


Figure 1: A basic RNN model that outputs x_t (dashed arrow) and keep updating its hidden state h_{t-1} (solid arrow).

The RNN computation in Eq. (5) implicitly models a regressor for x_t based on all past observations $x_{<t}$ through substituting $h_{<t}$. However, $\sigma_{x,t}$ does not appear. A naive volatility forecasting technique to induce $\sigma_{x,t}$ is to estimate σ_t^2 with σ_{t-1}^2, that is to write Eq. (8):

$$\sigma_{x,t}^2 = \mathrm{Var}(x_{<t}) = \frac{1}{(t-1)^2} \sum_{i=1}^{t-1} \sum_{j>i}^{t-1} (x_i - x_j)^2. \tag{8}$$

However, the expressive power of Eq. (8) is restricted. It will be clearer if we look at the lagged-form of Eq. (8):

$$\sigma_{x,t}^2 = [1 + (\frac{t-2}{t-1})^2]\sigma_{x,t-2}^2 + \frac{\sum_{i=1}^{t-1}(x_{t-1} - x_i)^2}{(t-1)^2}. \tag{9}$$

The autoregressive coefficients are not *learned* as in deterministic linear volatility models, but will only be a fixed function of t. For this reason, we argue that the standard RNN does not suffice complicated volatility modeling. Or, in other words, the standard RNN cannot generate sequences with certain types of time-varying volatility.

To solve this problem, a VRNN integrates the architecture of a variational autoencoder (VAE) [20]. Using the probabilistic parameterization of the joint distribution of $\boldsymbol{x}$ and the latent variables $\boldsymbol{z}$, the hidden state h as in RNNs is *by design* characterized by mean and variance information in a Gaussian case. For this reason, we can write the generative model as Eq. (10):

$$p(\boldsymbol{x}|\boldsymbol{z}) \sim \mathcal{N}(h^{\mu}(\boldsymbol{z}), h^{\sigma}(\boldsymbol{z})) \tag{10}$$

where volatility item $\sigma_{x,t}$ is embodied in the hidden state h.

We expand Eq. (10) across time steps.

$$p(x_t|z_{\leq t}, x_{<t}) \sim \mathcal{N}(h^{\mu,\sigma}(g_\tau(z_t), h_{t-1})) \tag{11}$$

$$p(z_t|z_{<t}, x_{<t}) \sim \mathcal{N}(h^{\mu,\sigma}(g_\upsilon(x_t), h_{t-1})) \tag{12}$$

where g_τ and g_υ are neural networks to approximate the non-linear functions. The parameterization of VRNN is therefore:

$$p(x_{\leq t}, z_{\leq t}) = \prod_{i=1}^{t} p(x_i|z_{\leq i}, x_{<i}) p(z_i|z_{<i}, x_{<i}). \tag{13}$$

where $p(x_{\leq t}, z_{\leq t})$ is joint probability of all the observations. Like in Eq. (5), hidden state of a VRNN is updated after generation of z_t and x_t (see Eq. (11) and Eq. (12)):

$$h_t = f_\theta(g_\upsilon(x_t), g_\tau(z_t), h_{t-1}). \tag{14}$$

3. Sentiment Time Series

Our goal being to integrate sentiment signals from social message streams into a volatility model, we first transform the streams into a sentiment time series. In this section, we discuss how a sentiment variable s_t is defined. We represent the sentiment information for a specific asset A as a quadruple, that is, to calculate sentiment polarity and intensity for each relevant message and aggregate them in a discrete-time axis. We have:

$$s_t(A) = (s_t^I(+), s_t^I(-), s_t^V(+), s_t^V(-)). \tag{15}$$

where $s_t^I(+)$, e.g., is the average intensity for all the positive messages, and $s_t^V(-)$ the count of negative messages regarding A. One can obtain the polarity score of a message by many techniques, e.g., training a neural network for sentiment analysis. In the SAVING model, however, we employ a knowledge-based method to embrace high interpretability and bridge the gap between connectionist models and many other disciplines, such as cognitive linguistics and commonsense reasoning [24, 25].

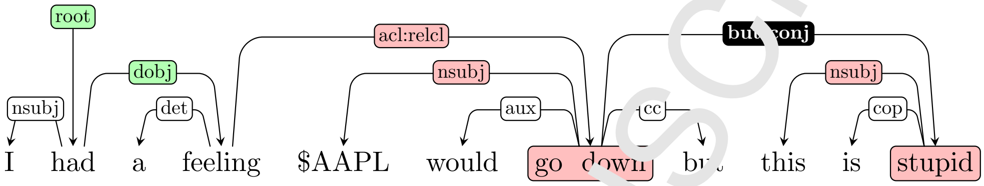


Figure 2: Polarity computing via reasoning through a typed dependency tree.

3.1. Polarity Computing

We compute the polarity score of each message using augmented sentic computing [26]. This approach relies on a group of linguistic rules to explicitly catch the long-term dependency in texts. Unlike its predecessor [27], which leverages polarity algebra, augmented sentic computing implements modificatory functions for different pivot types. A message is first parsed into multiple relation tuples with the Stanford typed dependency parser [28]. Later, a semantic parser will look at each uni-gram and bi-gram[1] and attempt to acquire the polarity score from a concept-level sentiment knowledge base. We employ the latest version of SenticNet [29], which contains 100,000 natural language concepts.

Figure 2 provides an example of how polarity scores of concepts are propagated along the dependency structure. Two concepts `go_down` and `stupid` are identified with polarity score of -0.07 and -0.93 respectively. The polarity of `stupid` is passed through a nominal subject relation so that the multi-word expression `this_is_stupid` inherits the score of -0.93. Similarly, the other high level relative clause `feeling_(that)_$AAPL_would_go_down` triggers an amplified intensity of $-\sqrt{|-0.07|} = -0.27$. Note that the two structures are linked by an adversative but-conjunction, the overall polarity is thus calculated as $\sqrt{|[(-0.27) + (-0.93)]/2|} = +0.78$ and passed to the root of the sentence. The computing process is backed-up with a multi-layered perceptron (MLP) network to solve zero-shot problem: if no concept is acquired from the knowledge base, the polarity is derived from supervised learning. Finally, if a message consists of multiple sentences, the overall polarity score will be an average of each sentence polarity score.

Messages arrive continuously with a timestamp T_i. Discrete-time operations, e.g., trading, will entail daily sentiment quantization. Taking market closure into consideration, we aggregate market sentiment from the previous trading day to one hour before the closing time. Algorithm 1 elaborates the process for sentiment time series construction.

[1] Only meaningful part-of-speech tag pairs, such as `ADJ+NOUN`, `VERB+NOUN`, and `VERB+ADV` are considered and lemmatized.

Algorithm 1: Constructing sentiment time series.

Data: message stream of a specific asset $\{m_i, T_i\}$
Result: sentiment time series $s_t(A)$
for $i = 1, 2, ...$ **do**
  if $T_i < t$ **then**
    $C(m_i) \leftarrow$ parse concepts from m_i;
    if $C(m_i) \bigcup KB \neq \emptyset$ **then**
      $s(m_i) \leftarrow$augmented sentic computing m_i;
    else
      $s(m_i) \leftarrow$MLP(m_i, Θ);
    end
    if $s(m_i) > 0$ **then**
      $s_t^I(+) \leftarrow \frac{n-1}{n} s_t^I(+) + \frac{1}{n} s(m_i)$;
      $s_t^V(+) \leftarrow s_t^V(+) + 1$;
    else if $s(m_i) < 0$ **then**
      $s_t^I(-) \leftarrow \frac{n-1}{n} s_t^I(-) + \frac{1}{n} s(m_i)$;
      $s_t^V(-) \leftarrow s_t^V(-) + 1$;
    $n \leftarrow n + 1$;
  else
    $t \leftarrow t + 1;\ [s_t(A), n] \leftarrow \mathbf{0}$;
  end
end
return $s_t(A) \leftarrow (s_t^I(+), s_t^I(-), s_t^V(+), s_t^V(-))$;

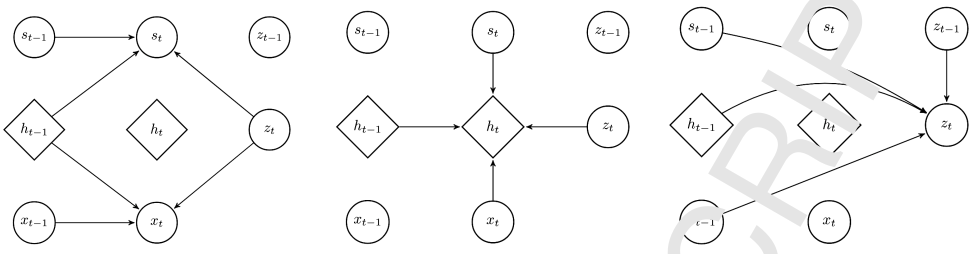
4. Sentiment-Aware Volatility Forecasting

We describe the three types of variable operations (generation-recurrence-inference) in the SAVING model with the presence of sentiment variable s_t (see Figure 3). To incorporate s_t, the latent variables will be shared by x_t and s_t, and the hidden state will be a concatenation of dimensions including return $\boldsymbol{x}$, sentiment $\boldsymbol{s}$ and latent variables $\boldsymbol{z}$. In this sense, we use bold notation $\boldsymbol{z}_t$ and $\boldsymbol{h}_t$ slightly different from that in VRNN. In VRNN, the bold notations $\boldsymbol{z}_t$ and $\boldsymbol{h}_t$ denote vector representations for each time period t, whereas in the SAVING model, for example, $\boldsymbol{z}$ denotes the full history of $z_t, z_{t-1}, z_{t-2}, \ldots$.

Our goal is to learn the complicated dynamics of variable interactions with an implicit function $\mathcal{F}$ of Eq. (16):

$$\mathcal{F}(\sigma_t, x_{<t}, s_{<t}, \boldsymbol{z}_{\leq t}) = 0. \tag{16}$$

With the shared latent variables and their autoregressive nature, namely $p(\boldsymbol{z}_t|\boldsymbol{z}_{<t}) \sim \mathcal{N}(\mu_{z,<t}, \sigma^2_{z,<t})$, two symmetric causal chains $s_{t-1} \rightarrow \boldsymbol{z_t} \rightarrow x_t$ and $x_{t-1} \rightarrow \boldsymbol{z_t} \rightarrow s_t$ are built up to model the bi-directional interaction between movements of asset price and market sentiment. Although this interaction has been justified by Granger causality test from both sides [30, 14, 15], the SAVING model is the first deep RNN architecture to elegantly capture this feature to the best of our knowledge (see Figure 4).



(a) Generation (b) Recurrence (c) Inference

Figure 3: Graphical illustration of the SAVING model: 3(a) joint generating process for asset return and sentiment from hidden state and latent variables; 3(b) updating hidden state of neurons; 3(c) inferencing posterior distribution of latent variables using time-lagged items. The hidden variables are denoted by diamonds and observable variables by circles.

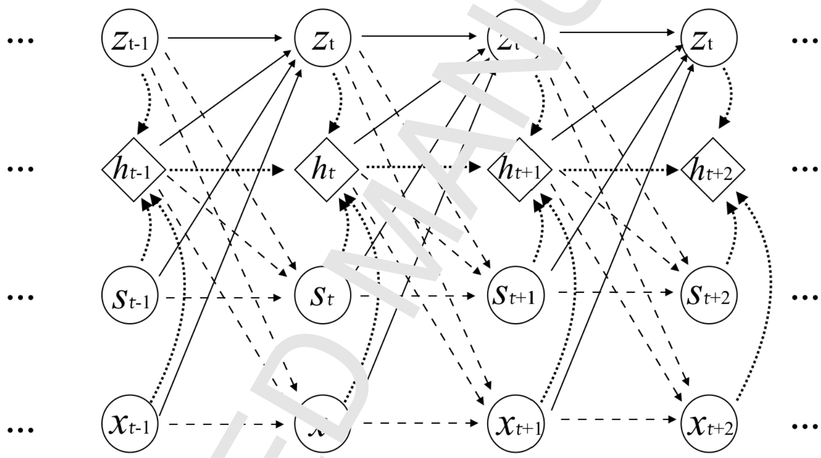


Figure 4: The full architecture of the SAVING model expanding on the time arrow. Generation operations are denoted by dashed arrows; recurrence operations are denoted by dotted arrows; inference operations are denoted by solid arrows.

4.1. Generation

The major difference between the SAVING model and the VRNN [7] is that for both return and sentiment variables, the conditional distribution of $\boldsymbol{z_t}$ is no longer an autoregressive Gaussian distribution, but takes into consideration the past input observations. For instance, generation of x_t involves three arguments:

$$p(x_t|x_{<t}, \boldsymbol{z}_t) \sim \mathcal{N}(\mu_{x,t}, \sigma^2_{x,t}), \tag{17}$$

$$\text{where } [\mu_{x,t}, \sigma_{x,t}] = \phi_x(\varphi^x(x_{t-1}), \varphi^z(\boldsymbol{z}_t), \boldsymbol{h}_{t-1}), \tag{18}$$

$\mu_{x,t}$ and $\sigma_{x,t}$ denote the parameters of the Gaussian distribution where x_t is sampled, φ^x and φ^z are neural networks that extract information from x_{t-1} and $\boldsymbol{z}_t$. Decoder ϕ_x maps the dimension back to $\dim(\mu_x + \sigma_x)$.

Similarly, s_t is synchronized with $\boldsymbol{z}_t$, denoting the sentiment accumulated between the current and the previous market closing time:

$$p(s_t|s_{<t}, \boldsymbol{z}_t) \sim \mathcal{N}(\mu_{s,t}, \sigma_{s,t}^2), \tag{19}$$

$$\text{where } [\mu_{s,t}, \sigma_{s,t}] = \phi_s(\varphi^s(s_{t-1}), \varphi^z(\boldsymbol{z}_t), \boldsymbol{h}_{t-1}) \tag{20}$$

$\mu_{s,t}$ and $\sigma_{s,t}$ denote the parameters of the Gaussian distribution where s_t is sampled, φ^s is a neural network that extracts information from s_{t-1} and ϕ_s is a decoder. To initialize the generation process, we assume the first pair of observation (x_t, s_t) is drawn from a standard Gaussian distribution.

4.2. Recurrence

The recurrent process updates the hidden state. In the SAVING model, h_t is a concatenation of three heterogeneous memories for the latent variables, return series and sentiment series respectively. Since $\boldsymbol{z}_t$ already contains joint information, its update is independent from other parts of the model:

$$\boldsymbol{h}_t = [h_t^z, \boldsymbol{h}_t^{x,s}] \tag{21}$$

$$h_t^z = f_\theta(\varphi^z(\boldsymbol{z}_t), h_{t-1}^z). \tag{22}$$

However, observed x_t and s_t are not integrated in the generation phase. Consequently, the hidden state will be shared by return and sentiment variables to facilitate forecasting through the completeness of $\boldsymbol{z}_t$:

$$\boldsymbol{h}_t^{x,s} = f_\theta(\varphi^x(x_t), \varphi^s(s_t), \boldsymbol{h}_{t-1}^{x,s}). \tag{23}$$

where f_θ is the neural network to join the new observations and the previous model (see Eq. (5)).

4.3. Inference

After the hidden state is updated by the latest x_t and s_t, the joint distribution of observable variables will be used to infer conditional distribution of $\boldsymbol{z}_{t+1}$. Variational inference [31, 20] is employed to stochastically optimize an approximation $q(\boldsymbol{z}_t|x_t, s_t)$ for the posterior $p(\boldsymbol{z}_t|x_{<t}, s_{<t})$, because of the difficulties in calculating the joint marginal distribution. According to Bayes' theorem, the conditional probability distribution of $\boldsymbol{z}_t$ is

$$p(\boldsymbol{z}_t|x_{<t}, s_{<t}) = \frac{p(x_{<t}, s_{<t}|\boldsymbol{z})p(\boldsymbol{z})}{p(x_{<t}, s_{<t})}, \tag{24}$$

where the integral $p(x_{<t}, s_{<t}) = \int p(x_{<t}, s_{<t}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z}$ is computationally intractable since $\boldsymbol{z}$ is unknown. Therefore, we learn parameters $\mu_{z,t}$ and $\sigma_{z,t}$, such that

$$q(\boldsymbol{z}_t|x_t, s_t) \sim \mathcal{N}(\mu_{z,t}, \sigma_{z,t}^2), \tag{25}$$

$$\text{where } [\mu_{z,t}, \sigma_{z,t}] = \psi(\varphi^z(\boldsymbol{z}_{t-1}), \varphi^{x,s}(\boldsymbol{x}, \boldsymbol{s}), \boldsymbol{h}_{t-1}). \tag{26}$$

Encoder ψ maps the dimension to $\dim(\mu_z + \sigma_z)$.

4.4. Learning

Parameters of the encoding and decoding neural networks, i.e. ϕ_x, ϕ_s, and ψ are co-trained using stochastic gradient descent. Recall that our objective is to minimize the Kullback-Leibler divergence between $q(\boldsymbol{z}_t|x_t, s_t)$ and $p(\boldsymbol{z}_t|x_{<t}, s_{<t})$:

$$KL(q_t||p_t) = -\sum_{i=1}^{t} p(x_{\leq i}, s_{\leq i}) \ln \frac{q(\boldsymbol{z}_i|x_i, s_i)}{p(x_{\leq i}, s_{\leq i}, \boldsymbol{z}_i)} \tag{27}$$

Since the integral of joint distribution $\int p(x_{\leq i}, s_{\leq i})dxds$ is fixed with a series of observations, minimizing the Kullback-Leibler divergence is equivalent to maximizing the Evidence Lower Bound (ELBO). After independent factorization of the generation and inference processes by substituting conditioning variables with time-lagged items, the loss function can be written as a negative ELBO [32]:

$$\ell(q) = \sum_{i=1}^{t}[KL(q_i||p_i) - \ln p(x_i, s_i|x_{<i}, s_{<i}, \boldsymbol{z}_i)]. \tag{28}$$

Then we apply gradient descent until convergence for:

$$\begin{cases} \phi_x & \leftarrow \phi_x - \rho\, \partial\ell(q)/\partial\phi_x \\ \phi_s & \leftarrow \phi_s - \rho\, \partial\ell(q)/\partial\phi_s \\ \psi & \leftarrow \psi - \rho\, \partial\ell(q)/\partial\psi \end{cases} \tag{29}$$

where gradients over the variational distribution q is actually intractable. However, we can eliminate q via Monte Carlo sampling [33]. We draw S samples of the latent variables $\boldsymbol{z}$ to approximate the gradient of the loss function:

$$\partial\ell = \frac{1}{S}\sum_{i=1}^{S} \ln \frac{p(x_{\leq i}, s_{\leq i}, \boldsymbol{z}_i)}{q(\boldsymbol{z}_i|x_i, s_i)} \partial \ln q(\boldsymbol{z}_i|x_i, s_i). \tag{30}$$

In the SAVING model, all the samples are assumed to be Gaussian. Due to its recursive nature, the forecasting horizon can be arbitrarily set once the learning phase is accomplished. Nevertheless, it is always recommended to re-train the model once new observations are received. In our experiment settings, the volatility forecasting is one-step-forward. This setting also ensures that the testing is strictly out-of-sample and over-fitting is reduced to a minimum level.

5. Experiments

We empirically investigate the effectiveness of the SAVING model over a range of baselines in the literature using historical stock price data and social media streams.

5.1. Data Preparation

A high-quality source for constructing sentiment time series is crucial to the performance of the SAVING model. We employ StockTwits[2], a social media platform for sharing ideas between

[2]http://stocktwits.com/

investors, traders, and entrepreneurs. We believe that the professional platform contains less noisy information compared to general-purpose venues, such as Twitter. Due to the limited accessibility of historical data, we test the SAVING model on 10 US stocks with relatively big market capitalization for over one year (from August 14, 2017 to August 22, 2018). In total, 82.2MB of textual data are collected, out of which a small portion is user-labeled. Table 1 lists the stock tickers, user-labeled numbers of positive and negative messages, positive/negative ratios, and the total numbers of messages in this period.

The messages associated with one specific company are filtered by cashtags[3]. To align sentiment time series and return series, we cut off the messages after 3:00 P.M. for the next trading day. This configuration allows for one hour of trading operations before market closure in practice. The price series are transformed to daily log returns $x_t = \log(price_t/price_{t-1})$ and normalized before feeding into the SAVING model. For both return series and sentiment time series, the missing values are filled by the closest previous record.

5.2. Model Settings

In our experiments, hyperparameters are set as following: the dimension of sentiment variable is set to $\dim s_t = 4$ to include both intensity and volume of daily message streams [9]; return variable is univariate and stock-specific, though 10 stock pairs (x_t, s_t) are concatenated for training to allow for modeling relationships of connected stocks; GRU cells are used for RNN function f_θ with 20% dropout and other neural networks such as ϕ_x, ϕ_s, ψ, φ^x, φ^s, and φ^z are implemented with two-layered MLPs, where each layer has 10 neurons.

Table 1: Basic data statistics. 2017-08-14 to 2018-08-22.

Ticker	#Pos.	#Neg.	Pos./Neg. ratio	Total
AAPL	28,4[illegible]	12,040	2.37	130,425
AGN	1,432	592	2.42	8,622
AMZN	27,902	7,029	3.97	97,580
BABA	35,607	4,488	7.93	97,253
GOOG	5,824	1,684	3.46	23,371
GS	3,6[illegible]	1,177	3.07	19,142
PFE	1,414	115	12.30	8,946
SBUX	2,618	1,461	1.79	14,873
STMP	485	112	4.33	3,202
TSLA	44,398	28,882	1.54	153,060

The embedding dimension for hidden state is set to 15: equally partitioned for return, sentiment, and latent variables to create a bottleneck for the input dimension. Adagrad optimizer [36] is found to be exceptional for training of the SAVING model parameters, whereas when using other popular optimizers such as Adam or stochastic gradient descent the converged NLLs are much higher. We set learning rate to $lr = 5 \times 10^{-4}$ and save the trained model each 10 epochs for 500 epochs in sum. The final model is empirically chosen with NLLs on the training set as the criterion. Then, this final model is used for one-step-forward forecasting only. As the

[3]A cashtag combines a dollar sign and some catchy word, in our context usually a stock ticker, e.g., $AAPL.

Table 2: Performance of the SAVING model and other compared benchmarks measured by NLL; only the SAVING model and s+LSTM employ sentiment information, other models are autoregressive.

Ticker	SAVING	GARCH [4]	EGARCH [18]	TARCH [34]	GJR [19]	GP-vol [35]	VRNN [7]	NSVM [8]	LSTM [21]	s+LSTM
AAPL	**-3.2798**	-3.1010	-3.1041	-3.1280	-3.1255	-3.1184	-2.9900	-2.9561	-0.7271	-2.1504
AGN	**-3.0387**	-3.0113	-3.0113	-3.0102	-3.0129	-3.0158	-3.0296	[illegible]	-0.9030	-1.2742
AMZN	**-2.9296**	-2.8183	-2.8183	-2.8194	-2.8187	-2.7918	-2.8559	[illegible]	-0.2887	-2.1488
BABA	**-3.2003**	-2.7253	-2.7253	-2.7228	-2.7292	-2.7229	-2.7240	[illegible]	-0.4521	-1.6865
GOOG	**-3.2319**	-3.0670	-3.0670	-3.0823	-3.0851	-3.0207	-2.9752	[illegible]	-0.4657	-2.1528
GS	**-3.2609**	-3.1267	-3.1267	-3.1245	-3.1333	-3.1160	-3.0121	[illegible]	-0.4832	-0.6010
PFE	-2.8548	-3.4011	-3.4078	-3.3921	**-3.4111**	-3.3911	[illegible]	[illegible]	-0.3844	-0.0159
SBUX	-2.9606	-3.1579	-3.1580	-3.1647	**-3.1656**	-3.1046	-3.0814	-2.9911	-0.5805	-1.5442
STMP	**-3.1081**	-2.3556	-2.3556	-2.4437	-2.4412	-2.3738	-2.3985	-2.2883	-0.4343	-1.2197
TSLA	**-2.7775**	-2.1776	-2.3483	-2.3735	-2.3493	-2.3005	[illegible]	-2.0769	-0.2006	0.9380
Average	**-3.0642**	-2.8942	-2.9122	-2.9261	-2.9272	-2.8956	-2.8749	-2.7617	-0.4920	-1.1856

new data come in, the training set is updated and the outdated model is discarded with a re-train procedure.

The SAVING model is trained in an online fashion on an Nvidia Tesla M60 GPU. The training time, in theory, will increase with the growth of historical data and the number of stocks considered. Empirically, convergence can be reached in minute-level for one year's data.

5.3. Compared Methods

Following the previous work [8], we adopt multiple benchmarks from different model groups, including deterministic linear models, stochastic volatility models, and deep recurrent neural models optionally equipped with sentiment information:

1. GARCH(1,1), where only x_{t-1} and σ^2_{t-1} are included on right hand side of Eq. (1); EGARCH(1,1), where the volatility variables are in their log form; TARCH(1,1,1) [34], where the power of σ is set to 1; GJR-GARCH(1,1,1), where one lagged asymmetric shock is added.
2. Gaussian-process volatility model (GP-vol) [35], where x_t is generated from a Gaussian process parameterized by $\mu = 0$ and σ_t^2.
3. Variational neural models including VRNN [7], where the prior on latent variables is autoregressive and can be understood as a time-varying VAE, and NSVM [8], where the prior also depends on past observations.
4. LSTM [21], which consists of two layers: a recurrent layer with 10 LSTM cells, 20% dropout and rectified linear unit activations (ReLU), and a dense layer of 10 neurons, summing up to 590 trainable parameters. Additionally we have s+LSTM, where sentiment and return variables are concatenated to form an input. The naïve volatility forecasting based on a sliding window of returns is implemented as with Eq. (8).

5.4. Results and Discussion

We observe some interesting facts after a quick pass through Table 1. Generally, people post far more positive messages than negative ones on social media, at least for the scope of stocks studied. For hotspot stocks, such as AAPL and TSLA, the ratio is about 2. Less discussed stock like PFE has an amazingly high ratio of around 12.

It also occurs that frequently discussed stocks are more volatile, while stocks without much news exposure have relatively stable returns. To evaluate the performance of different models we use negative log-likelihood (NLL), which assumes the predicted return and volatility form the ground truth distribution, and calculate the likelihood of observations. Under the Gaussian distribution hypothesis of x_t, NLL can also be interpreted as a mean squared error (MSE) with a volatility-based regularization [37] (see Eq. (31)):

$$\begin{aligned}\mathrm{NLL} &= -\sum_{i=1}^{t} \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp \frac{-(x_i-\mu_i)^2}{2\sigma_i^2} \\ &= \frac{1}{2}[t \ln 2\pi + \sum_{i=1}^{t} \ln \sigma_i^2 + \sum_{i=1}^{t} \frac{(x_i-\mu_i)^2}{2\sigma_i^2}]. \end{aligned} \tag{31}$$

Table 2 reports our experimental results. Furthermore, statistical significance analysis is summarized in Table 3. Since the distribution of NLLs is difficult to solve, we conduct paired one-tailed t-tests to investigate the improvement of using the SAVING model against other benchmark methods. The tests only require the differences to be roughly normally distributed, which is justified by the Shapiro-Wilk's W for all the pairs. Though we still list results of the Wilcoxon signed-rank test for reference.

Our null hypothesis is that the average performance of the SAVING model among different stocks is identical to or worse than the performance of the benchmark method. The hypothesis is rejected for VRNN, NSVM, LSTM and s+LSTM at a significance level of 0.05 and for GARCH and GP-vol at a significance level of 0.1. No sufficient evidence suggest that the SAVING model performs better than modified GARCH variants, though the relatively small p-values still encourage to prefer the SAVING model over EGARCH, TARCH, and GJR-GARCH.

Table 3: Pairwise statistical analysis of the performance of the SAVING model against other benchmarks, measured by NLLs.

Statistic	GARCH [4]	EGARCH [18]	TARCH [34]	GJR [19]	GP-vol [35]	VRNN [7]	NSVM [8]	LSTM [21]	s+LSTM
Shapiro-Wilk's W	0.96	0.96	0.95	0.95	0.96	0.98	0.97	0.90	0.92
p-value	0.78	0.82	0.70	0.66	0.76	0.96	0.91	0.22	0.37
paired one-tailed t	1.42	1.33	1.28	1.25	1.51	2.15	2.86	40.44	6.43
p-value	0.09*	0.11	0.12	0.12	0.08*	0.03**	0.01**	0.00**	0.00**
Wilcoxon's W	14.0	15.0	15.0	15.0	12.0	8.0	5.0	0.0	0.0
p-value	0.17	0.20	0.2[illegible]	0.20	0.11	0.05**	0.02**	0.01**	0.01**

The SAVING model outperforms other compared methods on 8 out of 10 stocks in terms of NLL. We compare deterministic linear models and deep neural models. Unlike previous studies, we do not find deep neural models, such as VRNN and NSVM to be superior to linear models. This may due to the use of different datasets and the difficulty of model tuning. In fact, all the linear model variants exhibit similar and stable behavior, though GJR-GARCH and TARCH perform slightly better than EGARCH and follows the original GARCH. It is worth mentioning that the GJR-GARCH model even produces better results on two stocks and also obtained very similar results on other five stocks. This observation justifies the necessity of considering asymmetric shock for volatility modeling.

LSTM and s+LSTM show very different results because these two models actually forecast only returns, thus a great amount of information is lost. The volatility is later derived using a naïve variance estimation based on both observed and forecasted values. Furthermore, the way to incorporate sentiment time series is by simple concatenation. While experiments of the SAVING model suggest this configuration of simple concatenation is not favored because we need to simultaneously minimize error for sentiment prediction. In some extreme cases (2 out

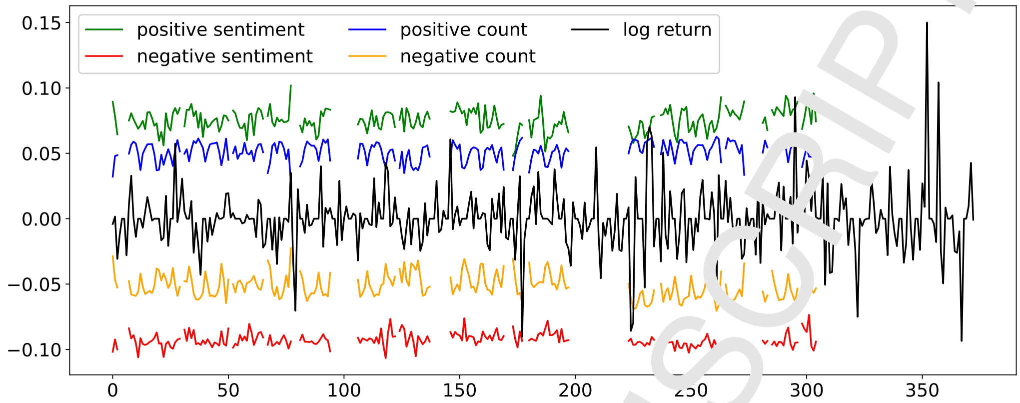


Figure 5: Return and sentiment series of TSLA. (Colors can be better displayed in the web version of this article.)

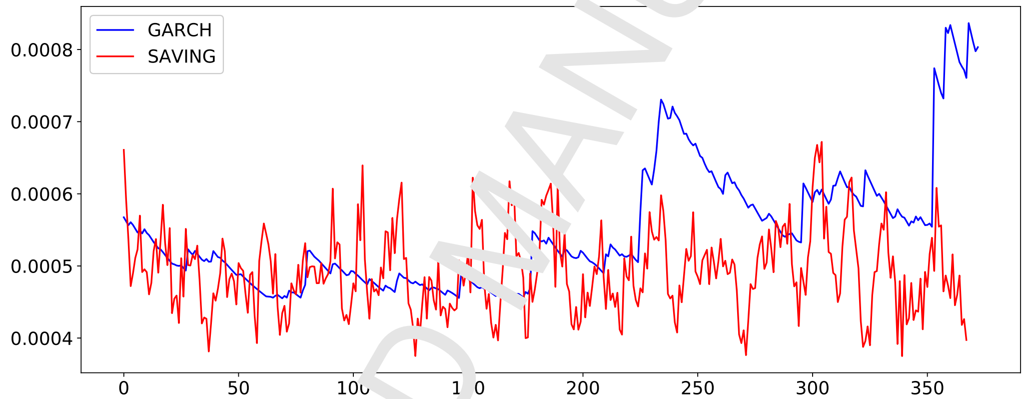


Figure 6: Forecasted volatility of TSLA by two models. (Colors can be better displayed in the web version of this article)

of 10), introducing sentiment increases the error of predictions. This not only happens to the LSTM/s+LSTM pair but also the SAVING/VRNN pair. Despite these rare cases, incorporating sentiment still significantly improves LSTM based naïve volatility forecasting, reducing average NLL by 0.6936.

The SAVING model, on the other hand, takes advantage of both sentiment information and the expressive power of a VRNN. Figure 5 shows the log returns of TSLA, where sentiment time series are scaled to fit into the chart. Although some movement segments are seemingly aligned, this complicated relation between sentiment and asset prices is hard to capture with linear models. Figure 6 provides forecasted volatility of the SAVING model and a GARCH model. We observe that the SAVING model can swiftly adapt to the current fluctuation level whereas GARCH suffers from a long recovery time from previous shocks, consistent with [8]. For this reason, the SAVING forecasting exhibits some desired properties, e.g., stationary, so that many financial models assuming a latent volatility still apply. The forecasting also shows pseudo-cyclical fluctuation, which is the case for returns as markets close regularly on weekends and holidays.

The recent peak on day 358 (August 7, 2018) corresponds to the drastic jump after Elon Musk announced his take-private plan on Twitter. We observe that the price quickly fell back in the following days after the news sentiment faded away. It is arguable that in a relatively long term the stock's volatility has really been influenced. However, the GARCH forecasting, based on the observed results from day 240 to day 290, seems to need months to digest this shock without considering market sentiment (see Figure 6).

6. Related Work

Volatility is a fundamental concept which many financial applications build on, such as asset/derivative pricing, hedging, and portfolio optimization [11, 38]. A group of deterministic linear models [4, 18, 19] pioneered time-varying volatility modeling. However, these models can be disputed by noise-contaminated data, which is unfortunately common in the financial world. Stochastic models, e.g., [39, 35] are thus developed to mitigate this weakness. In studies of computational intelligence, fuzzy logic and fuzzy time series are also widely used to to model time-varying volatility [40, 41].

Recent developments of RNN provided us the possibility of analyzing time series with models having usually hundreds or thousands times of parameters than classical models. Specifically, VRNN models that encode variance with model variables [7, 8] are suitable for volatility modeling. Other approaches, such as reinforcement learning, are explored to implicitly model risk aversion from simulated portfolio performance [42]. These models are powerful but pure data mining on past observations.

Since the relation between market sentiment and price movement has been widely testified [14, 15, 9], it would be more meaningful to seek for additional predictive power by incorporating this external information. More broadly speaking, the community witnessed a trend of grounding knowledge to connectionist models in recent years, thanks to the progress in text mining [43], text categorization [44], and text clustering [45] techniques. For instance, Ding et al. [10] extracted events from news and incorporated event embedding vectors into a deep convolutional neural network (DCNN) for stock price prediction; Luo et al. [46] incorporated manually-designed query-driven attention to employ expert knowledge for RNN-based financial sentiment analysis; Xing et al. [47] used the business classification knowledge to model stock relationships with application to portfolio construction. We believe this idea can produce more fruitful results in many scenarios.

7. Conclusion

In this work, we proposed the SAVING model to incorporate market sentiment as a form of external knowledge for volatility forecasting. The model inherits the expressive power of deep VRNNs, and further explore effective ways to align data from two different sources. The generalized model not only provides an interface to external knowledge bases but also captures the bi-directional interaction between market sentiment and asset price movements. Experiments show that the SAVING model outperforms many state-of-the-art volatility forecasting models, such as GJR-GARCH, GP-vol, VRNN, and NSVM. Future work includes quantification and visualization of sentiment-volatility interaction. We also plan to leverage sentiment knowledge bases specific to the finance domain and integrate non-Gaussian distribution hypotheses of asset returns or distribution agnostic models.

References

[1] H. Markowitz, Portfolio selection, The Journal of Finance 7 (1) (1952) 77–91.

[2] F. Black, M. Scholes, The pricing of options and corporate liabilities, Journal of Political Economy 81 (3) (1973) 637–654.

[3] P. Date, S. Islyaev, A fast calibrating volatility model for option pricing, European Journal of Operational Research 243 (2) (2015) 599–606.

[4] T. Bollerslev, Generalized autoregressive conditional heteroskedasticity, Journal of Econometrics 31 (3) (1986) 307–327.

[5] S. Heston, A closed-form solution for options with stochastic volatility with applications to bond and currency options, Review of Financial Studies 6 (2) (1993) 327–343.

[6] M. Hermans, B. Schrauwen, A recurrent latent variable model for sequential data, in: Proceedings of NIPS, Vol. 1, 2013, pp. 190–198.

[7] J. Chung, K. Kastner, L. Dinh, K. Goel, A. Courville, Y. Bengio, A recurrent latent variable model for sequential data, in: Proceedings of NIPS, Vol. 2, 2015, pp. 2980–2988.

[8] R. Luo, W. Zhang, X. Xu, J. Wang, A neural stochastic volatility model, in: Proceedings of AAAI, 2018, pp. 6401–6408.

[9] F. Z. Xing, E. Cambria, L. Malandri, C. Vercellis, Discovering bayesian market views for intelligent asset allocation, in: Proceedings of ECML PKDD, 2018.

[10] X. Ding, Y. Zhang, T. Liu, J. Duan, Deep learning for event-driven stock prediction, in: Proceedings of IJCAI, 2015, pp. 2327–2333.

[11] L. Malandri, F. Z. Xing, C. Orsenigo, C. Vercellis, E. Cambria, Public mood–driven asset allocation: the importance of financial sentiment in portfolio management, Cognitive Computation 10 (6) (2018) 1167–1176.

[12] S. Kelly, K. Ahmad, Estimating the impact of domain-specific news sentiment on financial assets, Knowledge-Based Systems 150 (2018) 116–126.

[13] F. Z. Xing, E. Cambria, R. E. Welsch, Natural language based financial forecasting: A survey, Artificial Intelligence Review 50 (1) (2018) 49–73.

[14] J. Bollen, H. Mao, X. Zeng, Twitter mood predicts the stock market, Journal of Computational Science 2 (1) (2011) 1–8.

[15] J. Smailović, M. Grčar, N. Lavrač, M. Žnidaršič, Stream-based active learning for sentiment analysis in the financial domain, Information Sciences 285 (2014) 181–203.

[16] E. Cambria, D. Rajagopal, D. Olsher, D. Das, Big social data analysis, in: R. Akerkar (Ed.), Big Data Computing, Chapman and Hall/CRC, 2013, Ch. 13, pp. 401–414.

[17] R. F. Engle, Autoregressive conditional heteroscedasticity with estimates of variance of united kingdom inflation, Econometrica 50 (4) (1982) 987–1008.

[18] D. B. Nelson, Conditional heteroskedasticity in asset returns: A new approach, Econometrica 59 (2) (1991) 347–370.

[19] L. R. Glosten, R. Jagannathan, D. E. Runkle, On the relation between expected value and the volatility of the nominal excess return on stocks, The Journal of Finance 48 (1993) 1779–1801.

[20] D. P. Kingma, T. Salimans, R. Józefowicz, X. Chen, I. Sutskever, M. Welling, Improving variational autoencoders with inverse autoregressive flow, in: Proceedings of NIPS, 2016, pp. 4736–4744.

[21] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780.

[22] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, in: Proceedings of EMNLP, 2014, pp. 1724–1734.

[23] J. L. Elman, Finding structure in time, Cognitive science 14 (2) (1990) 179–211.

[24] E. Cambria, A. Hussain, C. Havasi, C. Eckl, Common sense computing: From the society of mind to digital intuition and beyond, in: J. Fierrez, J. Ortega, A. Esposito, A. Drygajlo, M. Faundez-Zanuy (Eds.), Biometric ID Management and Multimodal Communication, Vol. 5707 of Lecture Notes in Computer Science, Springer, Berlin Heidelberg, 2009, pp. 252–259.

[25] E. Cambria, D. Olsher, K. Kwok, Sentic activation: A two-level affective common sense reasoning framework, in: Proceedings of AAAI, 2012, pp. 186–192.

[26] F. Z. Xing, E. Cambria, R. E. Welsch, Intelligent bayesian asset allocation via market sentiment views, IEEE Computational Intelligence Magazine 13 (4) (2018) 25–34.

[27] S. Poria, E. Cambria, A. F. Gelbukh, F. Bisio, A. Hussain, Sentiment data flow analysis by means of dynamic linguistic patterns, IEEE Computational Intelligence Magazine 10 (4) (2015) 26–36.

[28] M.-C. de Marneffe, C. D. Manning, The stanford typed dependencies representation, in: Coling: Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation, 2008, pp. 1–8.

[29] E. Cambria, S. Poria, D. Hazarika, K. Kwok, SenticNet 5: Discovering conceptual primitives for sentiment analysis

by means of context embeddings, in: Proceedings of AAAI, 2018, pp. 1795–1802.

[30] C. W. J. Granger, Investigating causal relations by econometric models and cross-spectral methods, Econometrica 37 (3) (1969) 424–438.

[31] D. J. Rezende, S. Mohamed, D. Wierstra, Stochastic backpropagation and approximate inference in deep generative models, in: Proceedings of ICML, 2014, pp. 1278–1286.

[32] C. M. Bishop, N. D. Lawrence, T. Jaakkola, M. I. Jordan, Approximating posterior distributions in belief networks using mixtures, in: Proceedings of NIPS, 1997, pp. 1–7.

[33] R. Ranganath, S. Gerrish, D. M. Blei, Black box variational inference, in: Proceedings of AISTATS, 2014, pp. 814–822.

[34] R. Rabemananjara, J. M. Zakoian, Arch models and asymmetries in volatility, Journal of Applied Econometrics 8 (1) (1993) 31–49.

[35] Y. Wu, J. M. Hernández-Lobato, Z. Ghahramani, Gaussian process volatility model, in: Proceedings of NIPS, Vol. 27, 2014, pp. 1044–1052.

[36] J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, Journal of Machine Learning Research 12 (2011) 2121–2159.

[37] S. J. Taylor, Asset Price Dynamics, Volatility, and Prediction, Princeton University Press, 2005.

[38] T. Bao, C. Diks, H. Li, A generalized capm model with asymmetric power distributed errors with an application to portfolio construction, Economic Modelling 68 (2018) 611–621.

[39] S. Kim, N. Shephard, S. Chib, Stochastic volatility: Likelihood inference and comparison with arch models, The Review of Economic Studies 65 (3) (1998) 361–393.

[40] Q. Song, B. S. Chissom, Fuzzy time series and its models, Fuzzy Sets and Systems 54 (1993) 269–277.

[41] O. Duru, A multivariate model of fuzzy integrated logical forecasting method (m-filf) and multiplicative time series clustering: A model of time-varying volatility for dry cargo freight market, Expert Systems with Applications 39 (4) (2012) 4135–4142.

[42] Y. Ding, W. Liu, J. Bian, D. Zhang, T.-Y. Liu, Investor-imitator: A framework for trading knowledge extraction, in: The ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2018, pp. 1310–1319.

[43] M. Dragoni, M. Federici, A. Rexha, Reus: a real-time unsupervised system for monitoring opinion streams, Cognitive Computation.

[44] Y. Li, Q. Pan, S. Wang, T. Yang, E. Cambria, A generative model for category text generation, Information Sciences 450 (2018) 301–315.

[45] L. M. Abualigah, A. T. Khader, E. S. Hanandeh, Hybrid clustering analysis using improved krill herd algorithm, Applied Intelligence 48 (11) (2018) 4047–4071.

[46] L. Luo, X. Ao, F. Pan, J. Wang, T. Zhao, N. Yu, Q. He, Beyond polarity: Interpretable financial sentiment analysis with hierarchical query-driven attention, in: Proceedings of IJCAI, 2018, pp. 4244–4250.

[47] F. Z. Xing, E. Cambria, R. E. Welsch, Growing semantic vines for robust asset allocation, Knowledge-Based Systems 165 (2019) 297–305.