

Uncertainty quantification for variational Bayesian dropout based deep bidirectional LSTM networks

Sardar Iqra, Noor Farzana, Iqbal Muhammad Javed, Alsanad Ahmed, Akbar Muhammad Azeem

This is a Author's accepted manuscript (AAM) version of a publication
published by Springer Nature
in Stochastic Environmental Research and Risk Assessment

DOI: 10.1007/s00477-025-02956-8

Copyright of the original publication:

© 2025 Springer Nature

Please cite the publication as follows:

Sardar, I., Noor, F., Iqbal, M.J. et al. Uncertainty quantification for variational Bayesian dropout based deep bidirectional LSTM networks. *Stoch Environ Res Risk Assess* (2025). <https://doi.org/10.1007/s00477-025-02956-8>

This version of the article has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: <http://dx.doi.org/10.1007/s00477-025-02956-8>

**This is a parallel published version of an original publication.
This version can differ from the original published article.**

Uncertainty quantification for variational Bayesian dropout based deep bidirectional LSTM networks

Iqra Sardar¹ · Farzana Noor¹ · Muhammad Javed Iqbal² · Ahmed Alsanad³ · Muhammad Azeem Akbar⁴

Abstract

Time series classification is a critical task in various domains, requiring robust models to handle inherent uncertainties in temporal data. These uncertainties, categorized as aleatoric and epistemic, pose significant challenges in achieving accurate predictions. In real-world applications, models often encounter unseen data that were not present during training process. Bayesian inference has been widely utilized for uncertainty quantification in statistics and machine learning. In this study, we proposed a Bayesian Deep Bi-LSTM model incorporating Variational Bayesian dropout with a Gaussian prior and Variational Autoencoder (VAE). The proposed technique efficiently handles uncertainty in both the model and data while VAE reducing the dimensionality of model parameters. We apply this framework to univariate time series datasets from the UCR repository and compare its performance with four traditional machine learning methods and four sequential deep learning models. Experimental results demonstrate that the Bayesian deep Bi-LSTM model effectively improves overall classification performance. In particular, the model benefits significantly from data augmentation using SMOTE when handling imbalanced dataset. The Variational Bayesian dropout model exhibits lower total uncertainty across both datasets, indicating more stable and reliable predictions compared to the VAE-based model. Future research should explore additional datasets from the UCR repository and investigate advanced uncertainty modeling techniques to further enhance performance and scalability.

Keywords Time series classification · Bayesian deep Bi-LSTM · Aleatoric and epistemic uncertainty · Variational Bayesian dropout · Variational autoencoder

1 Introduction

Time series classification (TSC) plays an important role in pattern recognition due to the growing availability of temporal data across diverse domains. Traditional machine learning methods often struggle with sequential dependencies in time series data, making TSC a complex problem. With advancements in sensor technologies, Information and Communication Technology (ICT), and reduced storage costs, high-dimensional time series data is now collected across diverse fields, requiring sophisticated models for accurate classification. TSC has critical applications in healthcare (e.g., EEG and ECG analysis) (Wang et al. 2013a, b), finance (Fisher and Krauss 2018), human activity recognition (Lara and Labrador 2013; Singh et al. 2017), industrial monitoring (Alwan and Roberts 1988), and earthquake detection (Kuyuk and Susumu 2018). The UCR/UEA time series archive (Chen et al. 2015; Bagnall et al. 2017), one of the largest repositories for time series datasets, demonstrate

✉ Iqra Sardar
iqra.phdst13@iiu.edu.pk

✉ Ahmed Alsanad
aasanad@ksu.edu.sa

✉ Muhammad Azeem Akbar
azeem.akbar@lut.fi

¹ Department of Mathematics and Statistics, International Islamic University, Islamabad 44000, Pakistan

² Department of Computer Science, University of Engineering and Technology Taxila, Taxila 47050, Pakistan

³ STC's Artificial Intelligence Chair, Department of Information Systems, College of Computer and Information Sciences, King Saud University, 11451 Riyadh, Saudi Arabia

⁴ Software Engineering Department, Lappeenranta-Lahti University of Technology, 53851 Lappeenranta, Finland

the wide range of real-world applications for TSC. Recent developments in computing power have enabled the rise of advanced TSC algorithms (Bagnall et al. 2017), including feature-based, model-based, ensemble, and deep learning approaches, addressing the computational demands of processing and classifying raw time series data.

Traditional TSC algorithms categorized into three primary types: model-based, feature-based, and distance-based approaches. Model-based methods build statistical or neural network models, such as Gaussian, Poisson, Autoregressive (Kini and Sekhar 2013), Markov, Hidden Markov Models (HMMs) (Antonucci et al. 2015), and Recurrent Neural Networks (RNNs), for each class using raw time series data. Classification is performed by identifying the class model that best fits the new data. Probabilistic distance measures are often used in these approaches, with applications ranging from text classification (Kim et al. 2006) using Naive Bayes to biological sequence analysis using HMMs and temporal data classification with RNNs. Feature-based time series classification involves extracting meaningful features from time series data and transforming it into feature vectors for classification using traditional machine learning algorithms. The selection of appropriate features is crucial in this approach, with techniques like Recursive Feature Elimination and zero-norm optimization used to simplify high-dimensional data (Lal et al. 2004; Chakraborty 2007). Recent advancements include the use of time series shapelets—distinctive subsequences of the series—as features (Ye and Keogh 2009). Additionally, transformation techniques like Fourier and Wavelet analysis, and unsupervised feature selection methods such as common principal component analysis (PCA), have been proposed by Yoon et al. (2005). Classification is performed using distance metrics on the feature-based representation of the time series.

Distance-based time series classification relies on measuring the similarity or dissimilarity between raw time series using efficient distance functions and applying traditional classifiers like k-nearest neighbors (k-NN). Euclidean distance (ED) is the simplest and most commonly used measure but requires equal-length series and is sensitive to time distortions. Elastic similarity measures, such as Dynamic Time Warping (DTW) (Berndt and Clifford 1994) and its variants, address these limitations and are widely regarded as the most effective for time series classification despite their computational cost. The combination of DTW with k-NN has historically been one of the most efficient approaches. A comparative study of different distance measures can be found (Wang et al. 2013a, b). Ensemble-based approaches for time series classification combine multiple classifiers to improve accuracy by integrating diverse feature sets or models. Examples include Elastic Ensemble (PROP), time series forest (TSF) (Lines and Bagnall 2015),

which uses 11 classifiers based on elastic distance measures, and Collective of Transformation Ensembles (COTE) (Bagnall et al. 2015), which integrates 35 classifiers using features from time and frequency domains. HIVE-COTE, an extension of COTE, further enhances performance (Lines et al. 2018) and becomes hugely computationally intensive and impractical to run on a real big data mining problem (Bagnall et al. 2017). The method requires training 37 classifiers as well as cross-validating each hyperparameter of algorithm, which makes its infeasible to train in some situation. However, ensemble methods are computationally intensive, requiring significantly more processing time than single classifiers, even with high-performance computing resources. The comprehensive review in state-of-the-art deep learning applications, challenges and future prospects for time series forecasting (Kumar et al. 2023; Ansari et al. 2024; Puri et al. 2024). TSC problems could be solved using a pure feature based algorithm (Bagnall et al. 2017; Neamtu et al. 2018) but not used deep learning models. While these traditional methods have shown success, they often struggle with long-range dependencies, noise sensitivity, and high computational costs.

To address these challenges deep learning models (LeCun et al. 2015) in various classification tasks, particularly GPU-accelerated ones, have gained popularity for time series due to their ability to process complex temporal patterns (Wang et al. 2017). Deep convolutional neural networks (CNNs) has revolutionized the field of computer vision (Krizhevsky et al. 2012). As in 2015, CNNs were used to reach human level performance in image recognition tasks (Szegedy et al. 2015). Comparative analysis of the most current Deep Neural Networks (DNNs) by Ismail Fawaz et al. (2019) aims to execute TSC task. Particularly with the development of new deeper architectures as Residual and CNNs, DNNs transformed the field of computer vision. Apart from images, sequential data such text and audio may also be handled using DNNs to obtain state-of-the-art performance for document classification and speech recognition. Xia et al. (2024) evaluated a forward echo state convolutional network (FESCN) against six classical approaches and four neural network models on the univariate time series dataset UCR. FESCN has a forward topology echo state network encoder which are a type of recurrent neural networks (RNNs) with large randomly linked hidden layer termed "reservoir" and a convolutional and max-pooling layer decoder. FESCN exceeds other approaches in classification accuracy, according to experiments.

Handling sequential data, such as series or sequences, an RNN is a specific kind of artificial neural network. Simple RNN modules comprise a simple structure with one "tanh" layer. Their limited memory, however, makes it challenging to remember information from past time steps—especially

in longer sequential data. Modern sequential deep learning architectures help to efficiently overcome these constraints as Long Short-Term Memory (LSTM) represents an advance type of standard RNNs by using cells with gates to selectively retain or forget information over time. LSTMs have been successfully applied to time series tasks like human activity recognition and stock classification. Karim et al. (2019) adding a squeeze and excitation block for the fully convolutional block, LSTM Fully Convolutional Network (LSTM-FCN) and Attention LSTM-FCN into a multivariate time series classification model is further improved accuracy. These models perform effectively on several challenging multivariate TSC tasks including action recognition or activity recognition. Bidirectional LSTM (BiLSTM) enhance this architecture by processing data in both forward and backward directions, leveraging long-term memory and mitigating vanishing gradients. By incorporating both forward and backward hidden states, Bi-LSTMs maintain a comprehensive view of both past and future information throughout the sequences (Jahangir et al. 2020; Toubeau et al. 2018). Deep BiLSTM architecture, which use multiple LSTM layers, can generally acquire good representations of high-dimensional, complex, and strong nonlinear sequential data.

TSC is inherently challenging due to its sequential nature, temporal dependencies, large volumes, and high-dimensional features. Traditional sequential deep learning models, while effective in feature extraction, often yield overconfident and uncertain outputs, struggle with temporal dependencies, lack robustness to noise and data variability and fail to handle aleatoric and epistemic uncertainties. To effectively address these issues Bayesian deep learning offer key advantages, including uncertainty quantification, improved generalization and temporal dependencies in the data without over estimation. These models also have ability to deal with long as well as short sequences and size of TSC tasks. Bayesian methods connect stochastic learning algorithms to posterior approximation in complex models. Dropout, a widely used regularization technique, was shown by Gal and Ghahramani (2016a, b) to be interpretable as a variational Bayes method. This insight led to a simple sampling technique for approximating deep neural network parameters from the posterior distribution. Additionally, prediction uncertainty in deep neural networks can be evaluated using Monte Carlo (MC) methods, specifically through MC dropout, which applies dropout during both training and inference to approximate Bayesian inference.

To address uncertainty, probabilistic methods can be integrated with deep neural networks, combining the strengths of both approaches. Kendall and Gal (2017) introduced Bayesian neural networks to quantify epistemic (model) uncertainty by representing neural network weights as probability

distributions. While epistemic uncertainty can be reduced by increasing the training data, aleatoric uncertainty, arising from inherent noise in the input data, remains irreducible regardless of data availability. This dual approach allows for more comprehensive uncertainty modeling in deep learning systems. Variational inference (VI) is an effective method for approximating posterior distributions in Bayesian models by minimizing the divergence between true and approximate posteriors (Zhang et al. 2018). While VI provides a tractable approach to parameter estimation, representing weights as probability distributions significantly increases the model's parameter space, leading to computational complexity especially for weight matrices. To address this, Variational Autoencoders (VAEs) can enhance the efficiency of Bayesian deep learning. VAEs are non-linear and probabilistic latent space (as Gaussian distribution) to estimate uncertainty unlike principal component analysis and singular value decomposition, which conduct deterministic linear transformations. VAE reduce dimensionality by encoding input data into a lower-dimensional probabilistic structure (Kingma and Welling 2013; Doersch 2016; Pawlowski et al. 2018), thus mitigating the computational burden while preserving the ability to handle both model and stochastic uncertainties for robust TSC. In this research work, we have trained roughly a million parameters across one dimensional TSC datasets. Though a lot of parameters go overfit (Zhang et al. 2017), almost small train set in the UCR/UEA archive. The experimental findings showed that proposed Bayesian deep Bi-LSTM model are able to significantly outperform the sequential deep learning and traditional methods. The main contribution are as:

- The proposed model Bayesian deep Bi-LSTM that integrates Variational Bayesian dropout to approximate posterior distributions.
- Experiments were performed for quantifying aleatoric and epistemic uncertainties in time series classification using a Bayesian framework.
- Integrating a Variational Autoencoder to manage high dimensional model parameters and improve computational efficiency.
- The newly proposed models are evaluated on univariate time series datasets from the UCR repository, compare it with traditional machine learning and sequential deep learning based methods.

The remainder of this article is structured as: Sect. 2 presents the material and methods, detailing the proposed Bayesian deep Bi-LSTM models. Section 3 describes the experimental setup, while Sect. 4 discusses results and comparison with baseline models. Finally, Sect. 5 concludes the work and outlines direction for future research.

2 Material and methods

This section describe the fundamental concepts of TSC, sequential deep learning models and proposed Bayesian deep Bi-LSTM framework, which integrates Variational Bayesian dropout and Variational Autoencoders for improved uncertainty quantification.

2.1 Time series classification

TSC involves assigning labels to time series data based on patterns within sequential observations.

A TSC data $D = [(T_1, Y_1), (T_2, Y_2), \dots, (T_d, Y_d)]$ contains d pairs (T_m, Y_m) consisting of a time series length T_m , representing univariate time series $Y_m = [y_1, y_2, \dots, y_t]$ and its corresponding one-hot encoded label vector X_m , containing K predefined classes, which is to assign a binary or multi-class label X_j to each time series segment based on the observed pattern $j \in [1, K]$ is equal to 1 if class of X_j is j and 0 otherwise.

2.2 Sequential deep learning for time series classification

Sequential deep learning models, particularly RNNs and LSTM networks, have been widely used for TSC due to their ability to capture sequential dependencies. LSTM mitigate the vanishing gradient problem by incorporating memory cells and gating mechanisms (Hochreiter and Schmidhuber 1997). LSTM uses four interacting layers inside of itself. Long-term memory can be efficiently retained in LSTM by use of this multi-layered architecture. While LSTMs mitigate this issue, they are not immune to it and may still face inefficiencies, requiring increased bandwidth for training large datasets. LSTM-FCN for TSC, augment the fast classification performance of temporal convolutional layers with the precise classification of LSTM RNNs (Karim et al. 2017). Unlike standard LSTMs, bidirectional nature of Bi-LSTM allows information to flow in both forward and backward directions, enabling the model to capture dependencies from both past and future contexts in the sequence (Jahangir et al. 2020; Toubeau et al. 2018). Attention-based models enhance TSC performance by dynamically focusing on relevant parts of the input sequence (Zhang et al. 2022).

Despite their effectiveness, these models often make overconfident predictions and fail to quantify uncertainty, limiting their reliability in real-world applications.

2.3 Proposed model: Bayesian deep Bi-LSTM

To address these challenges, we propose a Bayesian deep Bi-LSTM model incorporating Variational Bayesian dropout

for uncertainty quantification and Variational Autoencoder to manage high dimensional model parameters. The complete proposed model is presented in Fig. 1.

2.3.1 Deep Bi-LSTM network

The proposed model consists of deep Bi-LSTM layers, which extract temporal features by processing input sequences bidirectionally. Bidirectional LSTM (Bi-LSTM) is enhanced by adding two LSTM layers—one processing data forward, one backward. The network can capture contextual information from past and future time steps to develop stronger long-term dependencies with this bidirectional technique. By combining both forward and backward LSTM layers, Bi-LSTM networks effectively disentangle the recurrent relationships in sequential data, improving accuracy and enabling a deeper understanding of temporal dependencies. LSTM models process data bi-directionally in the hidden layers, gathering essential information from both directions. This helps retrieve many contextual details. Bi-LSTM models send the above data to the same output layer. However, the Bi-LSTM model output at time “ t ” is affected by both the preceding and following segments (Ogawa and Hori 2017). After training a tremendous amount of data with large and complex sequences, this study uses a deep Bi-LSTM network to understand temporal signals and recognize sequential information. Deep architectures use numerous Bi-LSTM layers to learn input data hierarchical representations at different abstraction levels. The Bi-LSTM equations in the forwards path correspond as:

$$\vec{\mathbf{f}}_t = \sigma \left(\vec{\mathbf{W}}_{y,f} \mathbf{y}_t + \vec{\mathbf{W}}_{h,f} \vec{\mathbf{h}}_{t-1} + \vec{\mathbf{b}}_f \right) \quad (1)$$

$$\vec{\mathbf{i}}_t = \sigma \left(\vec{\mathbf{W}}_{y,i} \mathbf{y}_t + \vec{\mathbf{W}}_{h,i} \vec{\mathbf{h}}_{t-1} + \vec{\mathbf{b}}_i \right) \quad (2)$$

$$\vec{\mathbf{o}}_t = \sigma \left(\vec{\mathbf{W}}_{y,o} \mathbf{y}_t + \vec{\mathbf{W}}_{h,o} \vec{\mathbf{h}}_{t-1} + \vec{\mathbf{b}}_o \right) \quad (3)$$

$$\vec{\mathbf{g}}_t = \vartheta \left(\vec{\mathbf{W}}_{y,g} \mathbf{y}_t + \vec{\mathbf{W}}_{h,g} \vec{\mathbf{h}}_{t-1} + \vec{\mathbf{b}}_g \right) \quad (4)$$

$$\vec{\mathbf{c}}_t = \vec{\mathbf{f}}_t * \vec{\mathbf{c}}_{t-1} + \vec{\mathbf{i}}_t * \vec{\mathbf{g}}_t \quad (5)$$

$$\vec{\mathbf{h}}_t = \vec{\mathbf{o}}_t * \tanh \left(\vec{\mathbf{c}}_t \right) \quad (6)$$

where the rightward arrow accent indicates the forward path.

The gates are computed using sigmoid and tanh activation functions, ensuring that the outputs are in the range $[0, 1]$ (forget and input gates) and $[-1, 1]$ (for the cell state).

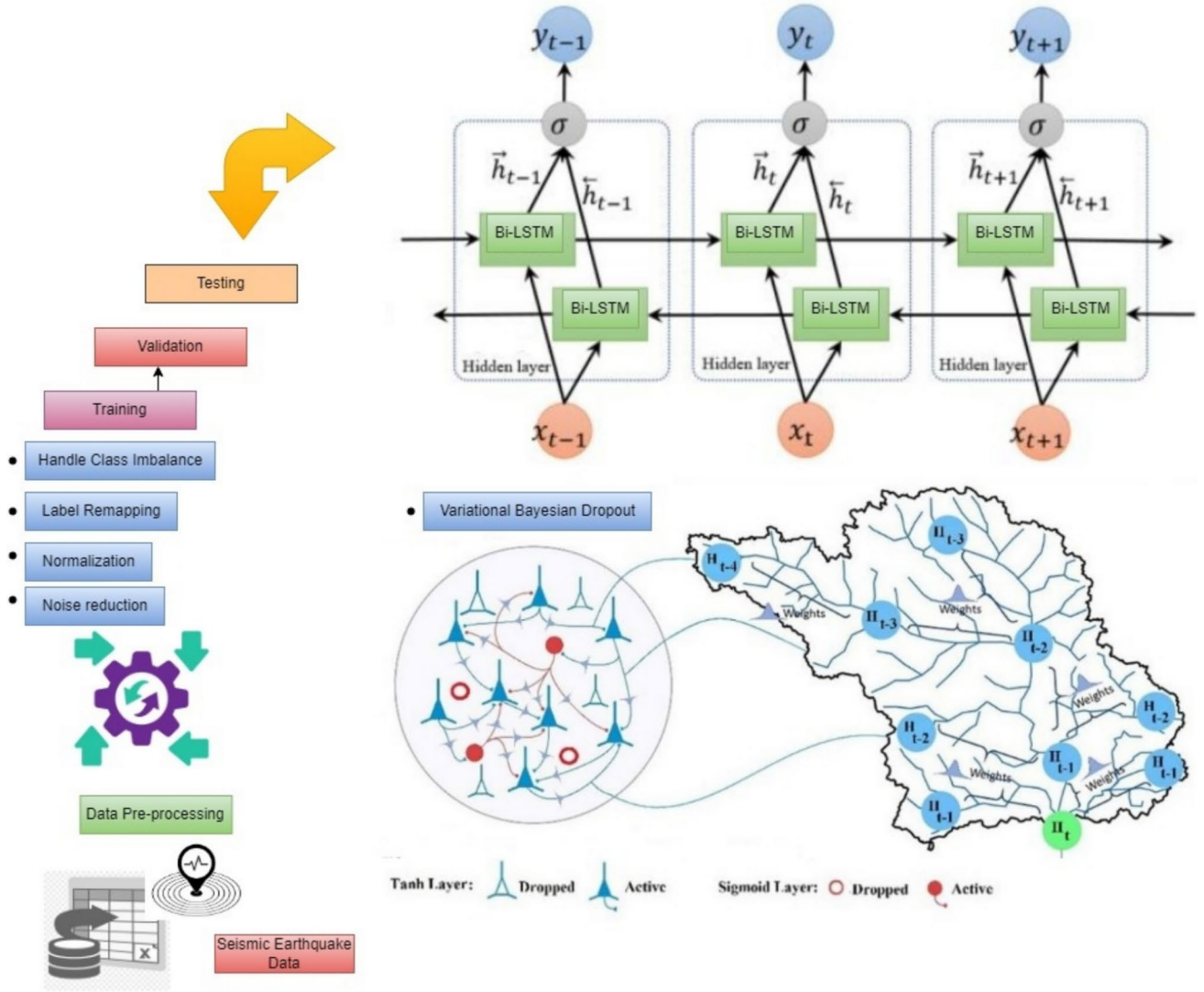


Fig. 1 Complete flow diagram of the proposed model

The backward LSTM generates the hidden state $\overleftarrow{\mathbf{h}}_t$ at time step t based on future hidden $\overleftarrow{\mathbf{h}}_{t+1}$ and the input vector \mathbf{y}_t , while the forward LSTM $\overrightarrow{\mathbf{h}}_t$ uses past hidden $\overrightarrow{\mathbf{h}}_{t-1}$ and the input vector \mathbf{y}_t . Concatenating the hidden vectors of both directions creates the Bi-LSTM model's final hidden state. The final Bi-LSTM model output at step t is:

$$\mathbf{h}_t = \left[\overrightarrow{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t \right] \quad (7)$$

The backward direction hidden state from Eq. (7) is replaced by $\overleftarrow{\mathbf{h}}_t$. TSC data is analyzed using a deep Bi-LSTM model to find prolonged temporal patterns. Both forward and backward passes created a deep Bi-LSTM network with two-layer LSTMs.

2.3.2 Variational Bayesian dropout with a Gaussian prior

Bayesian probability theory provides a robust framework for quantifying model uncertainty (Gal and Ghahramani 2016a), particularly by distinguishing between epistemic and aleatoric uncertainties. By integrating Bayesian theory with deep Bi-LSTM networks, both types of uncertainty can be captured, improving prediction performance to state-of-the-art levels. Variational Bayesian methods are fundamental for approximating intractable integrals, encountered in Bayesian inference, particularly for models involving observed variables, unknown parameters, and latent variables. These methods treat latent variables as unobserved, facilitating reliable uncertainty quantification. Variational Bayesian approaches serve two primary purposes:

- (a) Analyzing the posterior probability of unobserved variables for statistical inference.
- (b) To find a lower bound for the marginal likelihood of observed data given the model, with marginalization over unobserved variables.

This is used to select models because a model with a higher marginal likelihood fits the data better and is more likely to generate the data. We estimate the posterior distribution over the model weights using variational Bayesian approximation as exact Bayesian inference is computationally intractable in deep learning models. Variational inference optimizes a simpler, tractable distribution that approximates the true posterior via the minimization of the Kullback–Leibler (KL) divergence as follows:

$$KL[q(\mathbf{W}) \parallel g(\mathbf{W}|\mathbf{Y}, \mathbf{X})] = E_{q(\mathbf{W})}[\log q(\mathbf{W}) - \log g(\mathbf{W}|\mathbf{Y}, \mathbf{X})] \quad (8)$$

Due to Bayesian inference, given time series data set into a classification problem the input is $\mathbf{Y} = \mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N$ and the label is $\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$, the posterior over weights using Bayes' theorem $g(\mathbf{W}|\mathbf{Y}, \mathbf{X}) = \frac{g(\mathbf{Y}, \mathbf{X}|\mathbf{W})g(\mathbf{W})}{g(\mathbf{Y}, \mathbf{X})}$, Substitute this into KL divergence equation:

$$KL[q(\mathbf{W}) \parallel g(\mathbf{W}|\mathbf{Y}, \mathbf{X})] = E_{q(\mathbf{W})}[\log q(\mathbf{W}) - \log g(\mathbf{Y}, \mathbf{X}|\mathbf{W}) - \log g(\mathbf{W}) + \log g(\mathbf{Y}, \mathbf{X})] \quad (9)$$

Since $g(\mathbf{Y}, \mathbf{X})$ is independent of \mathbf{W} , it can be ignored in the optimization. Rearranging:

$$KL[q(\mathbf{W}) \parallel g(\mathbf{W}|\mathbf{Y}, \mathbf{X})] = E_{q(\mathbf{W})}[\log q(\mathbf{W}) - \log g(\mathbf{W}) - \log g(\mathbf{Y}, \mathbf{X}|\mathbf{W})] \quad (10)$$

KL divergence minimization maximizes the evidence lower bound (ELBO):

$$\ell_{VI} = E_{q(\mathbf{W})}[\log g(\mathbf{Y}, \mathbf{X}|\mathbf{W})] - KL[q(\mathbf{W}) \parallel g(\mathbf{W})] \quad (11)$$

Here, $g(\mathbf{W})$ represents the prior distribution, which assumes to be Gaussian, while a variational distribution $q(\mathbf{W})$ denotes the variational approximation of the posterior, $g(\mathbf{Y}, \mathbf{X}|\mathbf{W})$ is true posterior distribution learned during training over weights $\mathbf{W} = \left(\overset{\leftarrow}{\mathbf{W}}_{ih}^l, \overset{\leftarrow}{\mathbf{W}}_{hh}^l, \overset{\leftarrow}{\mathbf{W}}_{ih}^l, \overset{\leftarrow}{\mathbf{W}}_{hh}^l, l = 1, \dots, L \right)$.

Thus the minimization objective is

$$\hat{\mathbf{W}} = \arg \min_{\mathbf{W}} \left[\frac{1}{N} \sum_{i=1}^N \log g(\mathbf{y}_i|\mathbf{x}_i, \mathbf{W}) + KL(q(\mathbf{W}) \parallel g(\mathbf{W})) \right]. \quad (12)$$

Variational Bayesian inference (MC-dropout) is an efficient and practical method to approximate Bayesian inference in complex models (Gal and Ghahramani 2016b).

MC-dropout, when applied at both training and inference stages, introduces randomness by randomly dropping units according to a Bernoulli distribution, thus approximating Bayesian inference. This technique allows the network to generate multiple stochastic forward passes, which can be aggregated to obtain predictive distributions. We propose that dropout (and its variants) in deep Bi-LSTM networks can be interpreted as a variational Bayesian approximation of a Gaussian Prior (GP). A GP is defined by its mean and variance functions, which together establish a distribution over potential model weights. Thus Gaussian process posterior forecasts are derived as weighted averages of observed data based on variance and mean functions. Bayesian neural networks differ from standard networks by introducing probabilistic weights rather than deterministic ones. The **Bayesian LSTM layer** is an extension of the traditional LSTM layer used in neural networks, modified to include uncertainty estimation by placing a probability distribution over the model's weights. This concept is essential in applications where capturing uncertainty is as important as providing accurate predictions, improved generalization and robust-decision making for time series classification. Bayesian LSTM layers represent each weight as a probability distribution usually modeled as a Gaussian distribution.

$$\mathbf{W} \sim N(\mu, \sigma^2)$$

where $\mathbf{W} = \left(\overset{\leftarrow}{\mathbf{W}}_{ih}^l, \overset{\leftarrow}{\mathbf{W}}_{hh}^l, \overset{\leftarrow}{\mathbf{W}}_{ih}^l, \overset{\leftarrow}{\mathbf{W}}_{hh}^l, l = 1, \dots, L \right)$ is the weight matrix, μ is the mean, and σ^2 is the variance. This distribution allows the model to capture uncertainty in the weight values.

In the training phase, the Bayesian deep Bidirectional LSTM model is fed with input sequences through the two layers of Bayesian LSTM. Each layer includes bidirectional Bayesian LSTM units with specified variance scaling, followed by dropout layers to prevent overfitting, creating hidden states for each time step. Bayesian LSTM layer with hidden units, each unit has a probabilistic weight Gaussian distribution. The prior distribution over weights follows a GP.

$\mathbf{W} \sim N(\mu, \sigma^2 - \alpha \cdot \hat{\sigma}^2)$ Where μ is the mean weight and $\sigma^2 - \alpha \cdot \hat{\sigma}^2$ is the variance, scaled by a factor $\alpha = 0.1$ controls the uncertainty of the weights by scaling the variance, effectively making the distribution more precise (narrower). Smaller values of α reduce the uncertainty in weights. Bidirectional LSTMs process the input sequence forward and backward as:

$$\mathbf{h}_t = \overset{\leftarrow}{\mathbf{h}}_t \oplus \overset{\leftarrow}{\mathbf{h}}_t$$

where \oplus denotes concatenation and h_t represents the hidden state at time step t , produced by both the forward and backward LSTMs. Return sequences ensures that each time step outputs a hidden state, so the output of this layer is a sequence of hidden states $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_T]$, where T is the sequence length and each \mathbf{h}_t is a hidden state. Concatenating hidden states from the previous forward pass and the backward pass creates the final hidden states for each time step. Back propagation across time updates network weights to lessen predicted-true output difference.

Variational dropout is applied after each Bayesian LSTM layer, introducing a regularization effect by randomly deactivating neurons at each training step. In variational dropout, the dropout masks \mathbf{M}_i can be considered as samples from a Bernoulli distribution with probability p .

$$\mathbf{M}_i \sim \text{Bernoulli}(p)$$

For each neuron i , the output \tilde{y}_i after dropout can be represented as $\tilde{y}_i = \mathbf{M}_i \cdot \mathbf{y}_i$,

Where \mathbf{y}_i is the neuron's original output, and $\mathbf{M}_i = 0$ with probability $1 - p$, simulating weight uncertainty. Outlining this difference usually uses a loss function like Binary cross-entropy (BCE). BCE is a binary classification loss function. It quantifies the difference between true labels and model-predicted probabilities. For a single prediction, if: $y \in [0, 1]$ represents the true label, where 1 indicates the positive class and 0 the negative class. $\hat{y} \in [0, 1]$ represents the predicted probability of the positive class, i.e., the model's output. The binary cross-entropy loss is given by:

$$\text{BCE} = -(y \cdot \log(\hat{y}) + (1 - y) \cdot \log(1 - \hat{y})) \quad (13)$$

The BCE loss for a batch of size N (where each instance is indexed by i) is the average of the losses for each individual prediction:

$$\text{BCE}_{\text{batch}} = -\frac{1}{N} \sum_{i=1}^N (\mathbf{y}_i \cdot \log(\hat{\mathbf{y}}_i) + (1 - \mathbf{y}_i) \cdot \log(1 - \hat{\mathbf{y}}_i)) \quad (14)$$

In practice, deep Bi-LSTM network training adjusts learning rates, dropout rates, and LSTM layer numbers. An optimization approach like Adam (Adaptive Moment Estimation) is used to update network weights and model parameters to minimize loss. Adam mixes Momentum and RMSProp features and adjusts the learning rate for each parameter.

Mathematical as: ϑ represent the model parameters (weights and biases). $g_t = \nabla_{\vartheta} J(\vartheta)$ represent the gradient of the loss function $J(\vartheta)$ with respect to parameters ϑ at time step t . Each parameter has its own learning rate, which adapts as training progresses. Dropout regularization prevents deep architecture overfitting. To fit training data, alter

epoch number and batch size based on dataset size since the proposed model trained on different datasets. The deep Bi-LSTM network predicts sequences at each testing time step. The model's output are compared for accuracy, precision, recall, and F1 score. The model experiments with complex data patterns and representations using deep architecture. The deep Bi-LSTM network generates output by evaluating time-related sequences using cells for forward and backward propagation.

2.3.3 Aleatoric and epistemic uncertainty

Data uncertainty and model uncertainty are types of predictive uncertainty. Aleatoric uncertainty is caused by data generating unpredictability and cannot be mitigated by gathering more data. It is due to inherent randomness and the limitations of accurately predicting outcomes. Epistemic uncertainty stems from the model's inability to fully capture the underlying data patterns. Both uncertainties contribute to the overall uncertainty in predictions, but they originate from different sources. Homoscedastic and heteroscedastic aleatoric uncertainty exist (Kendall and Gal 2017). No matter the input, homoscedastic uncertainty is constant. However, heteroscedastic uncertainty changes with model inputs. However, epistemic uncertainty is caused by a lack of data, which may be explained with further observations. A Bayesian neural network (BNN) with a Gaussian prior distribution across its weights captures epistemic uncertainty in models. BNN marginalizes deep Bi-LSTM weight parameters by averaging all potential weights $\mathbf{W} \sim N(0, \sigma^2)$ and replacing them with distributions. Due to Bayesian inference the posterior over weights is $g(\mathbf{W}|\mathbf{Y}, \mathbf{X})$. The model likelihood would be $g(\mathbf{x}|p^{\mathbf{W}}(\mathbf{y}))$ which $p^{\mathbf{W}}(\mathbf{y})$ denotes the random output of the BNN. Assuming the likelihood as a Gaussian with $[\mu : p^{\mathbf{W}}(\mathbf{y})]$ and $[\sigma : \text{observation noise}]$ as $g(\mathbf{x}|p^{\mathbf{W}}(\mathbf{y})) = N(p^{\mathbf{W}}(\mathbf{y}), \sigma^2)$. The marginal probability $g(\mathbf{X}|\mathbf{Y})$ cannot be calculated analytically, but it can be estimated, making posterior inference difficult. Numerous research have provided approximations (Graves 2011; Blundell et al. 2015; Hernandez-Lobato et al. 2016; Gal 2016). These approximate inference methods fit the posterior $g(\mathbf{W}|\mathbf{Y}, \mathbf{X})$ to a variational distribution $q(\mathbf{W})$. This proposed approximating all BNN weights instead of averaging, which is intractable. Dropout randomly loses nodes during training to prevent co-tuning (Gal and Ghahramani 2016a). Dropout would create random predictions from the approximate posterior during testing. This method finds a variational distribution $q(\mathbf{W})$ that minimizes KL divergence to the true model posterior $g(\mathbf{W}|\mathbf{Y}, \mathbf{X})$ in Eq. (8–11). According to Jordan et al. (1999), the likelihood term

for each data point \mathbf{x}_i from a predictive model $p^{\hat{\mathbf{W}}_i}(\mathbf{y}_i)$ with Gaussian noise:

$\mathbf{x}_i = p^{\hat{\mathbf{W}}_i}(\mathbf{y}_i) + \epsilon$, $\epsilon \sim N(0, \sigma^2)$. The likelihood of \mathbf{x}_i given $p^{\hat{\mathbf{W}}_i}(\mathbf{y}_i)$ is:

$$g(\mathbf{x}_i | p^{\hat{\mathbf{W}}_i}(\mathbf{y}_i)) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|\mathbf{x}_i - p^{\hat{\mathbf{W}}_i}(\mathbf{y}_i)\|^2}{2\sigma^2}\right), \quad (15)$$

Gaussian likelihood simplifies negative log-likelihood (Kendall and Gal 2017):

$$-\log g(\mathbf{x}_i | p^{\hat{\mathbf{W}}_i}(\mathbf{y}_i)) = \frac{1}{2\sigma^2} \|\mathbf{x}_i - p^{\hat{\mathbf{W}}_i}(\mathbf{y}_i)\|^2 + \frac{1}{2} \log(2\pi\sigma^2), \quad (16)$$

Averaging over N samples $-\frac{1}{N} \sum_{i=1}^N \log g(\mathbf{x}_i | p^{\hat{\mathbf{W}}_i}(\mathbf{y}_i))$ measures the model's fit to the observed data. The Bayesian regularization $\frac{1-p}{2N} \|\vartheta\|^2$ term arises from the variational Bayesian approximation of a Gaussian prior on the weights. In Bayesian inference, weights are assigned prior distribution as Gaussian $\mathbf{W} \sim N(0, \frac{1}{\lambda} I)$. The loss function $\lambda(\vartheta, p)$ combines:

$$\lambda(\vartheta, p) = -\frac{1}{N} \sum_{i=1}^N \log g(\mathbf{x}_i | p^{\hat{\mathbf{W}}_i}(\mathbf{y}_i)) + \frac{1-p}{2N} \|\vartheta\|^2 \quad (17)$$

where, N and p denote the number of data points and dropout probability respectively, i is the sampled masked model weights and ϑ are the core distribution parameters to optimize. The regularization term ensures that dropout masks approximate Bayesian inference by applying stochastic weight sampling, effectively treating dropout as a form of variational Bayesian inference. The total predictive uncertainty is given by:

$$Var(\mathbf{x}) = \underbrace{E_{q(\mathbf{w})}[\sigma^2(\mathbf{y})]}_{Aleatoric} + \underbrace{V_{q(\mathbf{W})}[E[\mathbf{x}|\mathbf{W}]]}_{Epistemic} \quad (18)$$

$$Var(\mathbf{x}) \approx \sigma^2 + \frac{1}{T} \sum_{t=1}^T p^{\hat{\mathbf{W}}_t}(\mathbf{y})^T p^{\hat{\mathbf{W}}_t}(\mathbf{y}_t) - E(\mathbf{x})^T E(\mathbf{x}) \quad (19)$$

where $E(\mathbf{x}) \approx \frac{1}{T} \sum_{t=1}^T p^{\hat{\mathbf{W}}_t}(\mathbf{y})$. The predictive uncertainty term σ^2 represents data noise, whereas the second term describes model parameter uncertainty. Tune the observation noise parameter σ to incorporate aleatoric uncertainty. Heteroscedastic aleatoric uncertainty considers that observation noise may vary with different input data, unlike homoscedastic uncertainty, which assumes constant noise

for all inputs. By adopting a heteroscedastic model uncertainty assessment, we account for scenarios where some inputs produce noisier outputs than others, especially in cases with fewer positive examples. This approach allows for a data-dependent uncertainty estimation, making the uncertainty functionally related to the input data, thereby providing more accurate and tailored predictions.

$$\lambda_{Deep\ Bi-LSTM}(\vartheta) = \frac{1}{N} \sum_{i=1}^N \frac{1}{2\sigma(\mathbf{y}_i)^2} \|\mathbf{x}_i - p^{\hat{\mathbf{W}}_i}(\mathbf{y}_i)\|^2 + \frac{1}{2} \log \sigma(\mathbf{y}_i)^2 \quad (20)$$

Heteroscedastic Deep Bi-LSTM (Eq. (4)) into a Bayesian deep Bi-LSTM by distributing its weights to accommodate epistemic and aleatoric uncertainties. Compute model output with predictive mean and variance $\left[(\hat{\mathbf{x}}_i^f + \hat{\mathbf{x}}_i^b), (\hat{\sigma}_i^f + \hat{\sigma}_i^b)^2 = p^{\hat{\mathbf{W}}}(\mathbf{y}) \right]$ using model weights from estimated posterior $\hat{\mathbf{W}}_i \sim q(\mathbf{W})$. Consequently, the minimization objective causes:

$$\lambda_{Bay\ Deep\ Bi-LSTM}(\vartheta) = \frac{1}{N} \sum_{i=1}^N \frac{1}{2\hat{\sigma}_i^2} \|\mathbf{x}_i - (\hat{\mathbf{x}}_i^f + \hat{\mathbf{x}}_i^b)\|^2 + \frac{1}{2} \log(\hat{\sigma}_i^f + \hat{\sigma}_i^b)^2 \quad (21)$$

The loss avoids division by zero, therefore train the network to predict the log variance $\mathbf{s} = \log \hat{\sigma}_i^2$, for σ_y^2 numerical stability. The regression task implicitly learns in this method. Regressing unconstrained scalar values with $\exp(-\mathbf{s}_t)$ an exponential mapping yielded a positive domain and valid variance.

$$\lambda_{Bay\ Deep\ Bi-LSTM}(\vartheta) = \frac{1}{2N} \sum_{i=1}^N \left[\exp(-\mathbf{s}_i) \left(\mathbf{x}_i - \frac{\hat{\mathbf{x}}_i^f + \hat{\mathbf{x}}_i^b}{2} \right)^2 + \mathbf{s}_i \right] \quad (22)$$

The term \mathbf{s}_i regulates the $\exp(-\mathbf{s}_i)$ to prevent unreserved decrease during training. The combined model's predictive uncertainty can be approximated as follows:

$$Var(\mathbf{x}) \approx \underbrace{\frac{1}{T} \sum_{t=1}^T (\hat{\mathbf{x}}_t^f + \hat{\mathbf{x}}_t^b)^2 - \left(\frac{1}{T} \sum_{t=1}^T (\hat{\mathbf{x}}_t^f + \hat{\mathbf{x}}_t^b) \right)^2}_{Epistemic(Model)} + \underbrace{\frac{1}{T} \sum_{t=1}^T (\hat{\sigma}_t^f + \hat{\sigma}_t^b)^2}_{Aleatoric(data)} \quad (23)$$

where $\left[\left(\hat{\mathbf{x}}_i^f + \hat{\mathbf{x}}_i^b \right), \left(\hat{\sigma}_i^f + \hat{\sigma}_i^b \right)^2 \right]_{t=1}^T$ denotes a set of T

samples to both forward and backward layers during inference outputs $\left(\hat{\mathbf{x}}_i^f + \hat{\mathbf{x}}_i^b \right), \left(\hat{\sigma}_i^f + \hat{\sigma}_i^b \right)^2 = p^{\hat{\mathbf{W}}_t}(\mathbf{y})$ for randomly masked Bayesian weights $\hat{\mathbf{W}}_t \sim q(\mathbf{W})$ to capture epistemic uncertainty via variational dropout and aleatoric uncertainty average predictive variance across the samples.

2.3.4 Activation function

The activation function used in input first dense layer are ReLU (Rectified linear unit) $f(Y) = \max(0, Y)$. It does not activate all neurons at the same time, if the input is negative it will convert it to zero but another called Softplus activation function is a smooth approximation of ReLU, specifically used for variance prediction to ensure that the predicted variance is always non-negative and never zero. The Softplus function is defined as:

$$\text{softplus}(Y) = \frac{1}{\beta} \cdot \ln(1 + e^{\beta \cdot Y}), \text{ for } \beta = 1 \quad (24)$$

It is practically insignificant, but its subtle and analytically important advantage is that it never becomes zero for $Y \rightarrow -\infty$, whereas ReLU becomes zero for $Y \rightarrow -\infty$. Sigmoid $(Y) = \frac{1}{1+e^{-Y}}$ to predict the mean, ensuring it lies in the range $(0,1)$.

The output dense layer for mean prediction $\mu(Y) = \sigma(\mathbf{W}Y + b)$ and variance prediction

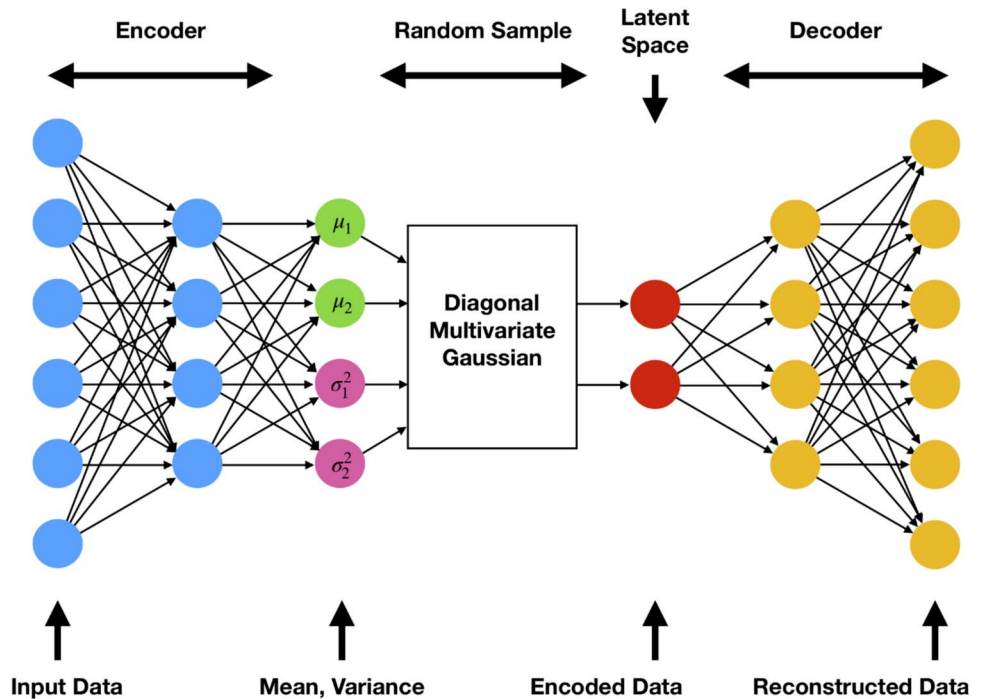
$\sigma^2(Y) = \text{softplus}(\mathbf{W}Y + b) = \ln(1 + e^{\mathbf{W}Y + b})$ are the \mathbf{W} weights and b bias of the variance layer.

2.3.5 Variational autoencoder

A Variational Autoencoder (VAE) is unsupervised deep generative which improve computational efficiency and reduce model complexity. VAE is also used to reduce the input layer's large dimensions when multiple time lags are considered and improve forecasting methods (Kaur et al. 2021). VAEs, widely employed in deep learning literature for dimensionality reduction (Bachhav et al. 2019), can improve Bayesian deep learning computing efficiency. VAEs use probabilistic encoders to reduce data dimensions (Kingma and Welling 2013). VAEs have successfully detected faults and anomalies in time-series energy data in power system applications using deep learning models (Wang et al. 2020). VAE uses Bayesian probability to teach neural networks complex data distribution from probabilistic latent variable spaces along with high dimensionality in the model parameters. VAE model structure is presented in Fig. 2. Encoder and decoder are in VAE. VAE encoders and decoders receive labels \mathbf{X} as conditions. Encoder input is a concatenation of original data \mathbf{Y} plus label information \mathbf{X} , while output is unaltered. The encoder maps input data \mathbf{Y} to latent space to get latent variables \mathbf{H} . Using latent variables \mathbf{H} and labels \mathbf{X} , the decoder produces output $\hat{\mathbf{Y}}$.

Instead of joint input variable distribution, VAE marginal likelihood can be calculated as:

Fig. 2 The model structure of Variational Autoencoder



$$g_{\vartheta}(\mathbf{Y}, \mathbf{X}) = \int_{\mathbf{H}} g_{\vartheta}(\mathbf{H}) g_{\vartheta}(\mathbf{Y}, \mathbf{X} | \mathbf{H}) d\mathbf{H} \quad (25)$$

The latent variable prior $g_{\vartheta}(\mathbf{H})$ follows the Gaussian distribution with parameter ϑ . The marginal likelihood is unsolvable because \mathbf{H} and ϑ are unknown. From input data (\mathbf{Y}, \mathbf{X}) , $g_{\vartheta}(\mathbf{H} | \mathbf{Y}, \mathbf{X})$ the latent variable $g_{\vartheta}(\mathbf{H})$ can be substituted. True posterior $g_{\vartheta}(\mathbf{H} | \mathbf{Y}, \mathbf{X})$ as:

$$g_{\vartheta}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) = \frac{g_{\vartheta}(\mathbf{H}) g_{\vartheta}(\mathbf{Y}, \mathbf{X} | \mathbf{H})}{g_{\vartheta}(\mathbf{Y})} \quad (26)$$

Variational inference is introduced here. The $q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X})$ Gaussian distribution with parameter φ (mean and variance) approximates real posterior distribution here (Kingma and Welling 2013).

Additionally, the marginal log-likelihood is:

$$\log g_{\vartheta}(\mathbf{Y} | \mathbf{X}) = \int_{\mathbf{H}} q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) d\mathbf{H} \cdot \log q_{\varphi}(\mathbf{Y} | \mathbf{X}) \quad (27)$$

$$\begin{aligned} &= \int_{\mathbf{H}} q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \log g_{\vartheta}(\mathbf{X} | \mathbf{Y}) d\mathbf{H} \\ &= \int_{\mathbf{H}} q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \log g_{\vartheta}(\mathbf{Y}, \mathbf{X}, \mathbf{H}) / g_{\vartheta}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) g_{\vartheta}(\mathbf{Y}) d\mathbf{H} \\ &= \int_{\mathbf{H}} q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \log \left(\frac{g_{\vartheta}(\mathbf{Y}, \mathbf{X}, \mathbf{H})}{q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) g_{\vartheta}(\mathbf{Y})} \cdot \frac{q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X})}{g_{\vartheta}(\mathbf{H} | \mathbf{Y}, \mathbf{X})} \right) d\mathbf{H} \\ &= \int_{\mathbf{H}} q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \log \left(\frac{g_{\vartheta}(\mathbf{Y}, \mathbf{X}, \mathbf{H})}{q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) g_{\vartheta}(\mathbf{Y})} \right) d\mathbf{H} \\ &\quad + \int_{\mathbf{H}} q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \log \left(\frac{q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X})}{g_{\vartheta}(\mathbf{H} | \mathbf{Y}, \mathbf{X})} \right) d\mathbf{H} \\ &= \text{ELBO} + D_{KL}(q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}), g_{\vartheta}(\mathbf{H} | \mathbf{Y}, \mathbf{X})) \quad (28) \end{aligned}$$

where $D_{KL}(q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}), g_{\vartheta}(\mathbf{H} | \mathbf{Y}, \mathbf{X}))$ is the Kullback–Leibler (KL) divergence of approximate posterior $q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X})$ and $g_{\vartheta}(\mathbf{H} | \mathbf{Y}, \mathbf{X})$ which is non-negative.

$$D_{KL}(q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) || g_{\vartheta}(\mathbf{H} | \mathbf{Y}, \mathbf{X})) \geq 0 \quad (29)$$

and zero if, and only if, $q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X})$ equals to true posterior distribution. $q_{\varphi}(\mathbf{Y} | \mathbf{X})$ is used to approximate $g_{\vartheta}(\mathbf{Y} | \mathbf{X})$. Variational inference's objective function was ELBO. Maximizing ELBO indirectly maximizes log likelihood. ELBO (evidence lower bound) is defined on log marginal probability can be further written as:

$$\text{ELBO} = \int_{\mathbf{H}} q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \log \left(\frac{g_{\vartheta}(\mathbf{Y}, \mathbf{X}, \mathbf{H})}{q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \cdot g_{\vartheta}(\mathbf{Y})} \right) d\mathbf{H} \quad (30)$$

$$\begin{aligned} &= \int_{\mathbf{H}} q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \log \left(\frac{g_{\vartheta}(\mathbf{X} | \mathbf{Y}, \mathbf{H}) g_{\varphi}(\mathbf{H} | \mathbf{Y}) \cdot g_{\vartheta}(\mathbf{Y})}{q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \cdot g_{\vartheta}(\mathbf{Y})} \right) d\mathbf{H} \\ &= \int_{\mathbf{H}} q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \log \left(\frac{g_{\vartheta}(\mathbf{H} | \mathbf{Y})}{q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X})} \right) d\mathbf{H} \\ &\quad + \int_{\mathbf{H}} q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}) \log g_{\vartheta}(\mathbf{X} | \mathbf{Y}, \mathbf{H}) d\mathbf{H} \\ &\quad - D_{KL}(q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}), g_{\vartheta}(\mathbf{H} | \mathbf{Y})) \\ &\quad + E_{q_{\varphi}}[\log g_{\vartheta}(\mathbf{X} | \mathbf{Y}, \mathbf{H})] \quad (31) \end{aligned}$$

Due to non-negativity of the KL divergence, the ELBO is a lower bound on the log-likelihood of the data. The marginal probability function can be maximized as the evidence lower bound. Thus, maximizing ELBO is:

$$\arg \max_{q_{\varphi}, \varphi(\mathbf{H})} \text{ELBO} = \max_{\vartheta, \varphi} [E_{q_{\varphi}}[\log g_{\vartheta}(\mathbf{X} | \mathbf{Y}, \mathbf{H})] - D_{KL}(q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}), g_{\vartheta}(\mathbf{H} | \mathbf{Y}))] \quad (32)$$

In the Bayesian deep Bi-LSTM approach, the estimated posterior is a distribution $q_{\vartheta, \varphi}(\mathbf{H})$, not a weight sample. Weight parameters chosen from the ideal distribution have high dimensionality. VAE is integrated to encode posterior approximation's huge sample space into lower-dimensional space to solve this problem.

The estimated posterior $q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X})$ is assumed to be multivariate Gaussian with mean $\mu_{\mathbf{YX}}$ and variance $\Sigma_{\mathbf{YX}}$. The KL divergence:

$$\begin{aligned} D_{KL}(q_{\varphi}(\mathbf{H} | \mathbf{Y}, \mathbf{X}), g_{\vartheta}(\mathbf{H} | \mathbf{Y})) &= \int_{\mathbf{H}} N(\mu_{\mathbf{YX}}, \Sigma_{\mathbf{YX}}) \log N(\mu_{\mathbf{YX}}, \Sigma_{\mathbf{YX}}) / N(0, \mathbf{I}) d\mathbf{H} \\ &= -\frac{1}{2} \sum_{t=1}^T \left(1 + \log(\Sigma_{\mathbf{YX}}^T)^2 - (\mu_{\mathbf{YX}}^T)^2 - (\Sigma_{\mathbf{YX}}^T)^2 \right) \quad (33) \end{aligned}$$

With $\mu_{\mathbf{YX}}^T$ and $\Sigma_{\mathbf{YX}}^T$ as the sample mean and variance at t time, $T = 1, 2, \dots, t$. VAE Bayesian Deep BiLSTM loss function second term represents likelihood of reconstructed data at decoder, while first term is KL divergence between approximated and real posterior distribution of latent variable. VAE aggregated loss function:

Table 1 Parameter details for time series classification dataset

| Dataset | Train size | Test size | Length | Classes | Type |
|-----------------------|------------|-----------|--------|---------|-----------|
| D1: Earthquakes | 139 | 322 | 512 | 2 | Sensor |
| D2: Synthetic Control | 300 | 300 | 60 | 6 | Simulated |

D1: Earthquakes- A sensor dataset used for seismic event classification, distinguishing between major and non-major seismic readings. The data comes from the Northern California Earthquake Data Center, spanning from December 1, 1967, to 2003, with each reading representing average seismic activity recorded hourly. A major seismic event is defined as any reading exceeding 5 on the Richter scale. Positive cases are defined as major events not preceded by another major event within at least 512 h, while negative cases are those with readings below 4 that are preceded by at least 20 non-zero readings within the same 512-h window. The dataset contains 86,066 hourly readings, resulting in 368 negative cases and 93 positive cases, with no temporal overlap between positive and negative instances (Bag-nall et al. 2018)

D2: Synthetic Control- A simulated control charts synthetically generated by the process in Alcock and Manolopoulos (1999) of six different balanced classes’ chart patterns: normal, cyclic, increasing trend, decreasing trend, upward shift and downward shift

$$\begin{aligned}
\lambda(\theta, \varphi; \mathbf{Y}, \mathbf{X}) &= -D_{KL}(q_{\varphi}(\mathbf{H}|\mathbf{Y}, \mathbf{X}), g_{\theta}(\mathbf{H}|\mathbf{Y})) \\
&\quad + E_{q_{\varphi}}[\log g_{\theta}(\mathbf{Y}|\mathbf{X}, \mathbf{H})] \\
&= \frac{1}{2} \sum_{t=1}^t \left(1 + \log(\Sigma_{\mathbf{YX}}^T)^2 - (\mu_{\mathbf{YX}}^T)^2 - (\Sigma_{\mathbf{YX}}^T)^2 \right) \\
&\quad + \frac{1}{T} \sum_{t=1}^t \log g_{\theta}(\mathbf{Y}|\mathbf{X}, \mathbf{H}^T)
\end{aligned} \quad (34)$$

where t is the sampling number and $\mathbf{H}^T = \mu_{\mathbf{YX}}^T + \Sigma_{\mathbf{YX}}^T \mathbf{x}$, $\varepsilon, \varepsilon \sim N(0, 1)$ is obtained by reparameterization, train and test the VAE-Bayesian deep BiLSTM model utilizing the loss function and evaluation metrics.

3 Experiments

The proposed model is evaluated using two univariate time series datasets from UCR archive or using the aeon toolkit to load the data directly from website for experiments. The performance is evaluated by comparing it with sequential deep learning and traditional methods respectively. The experiments are conducted on Google Colab utilizing Tensorflow and keras for implementation. The Bayesian inference components are integrated using Tensorflow probability.

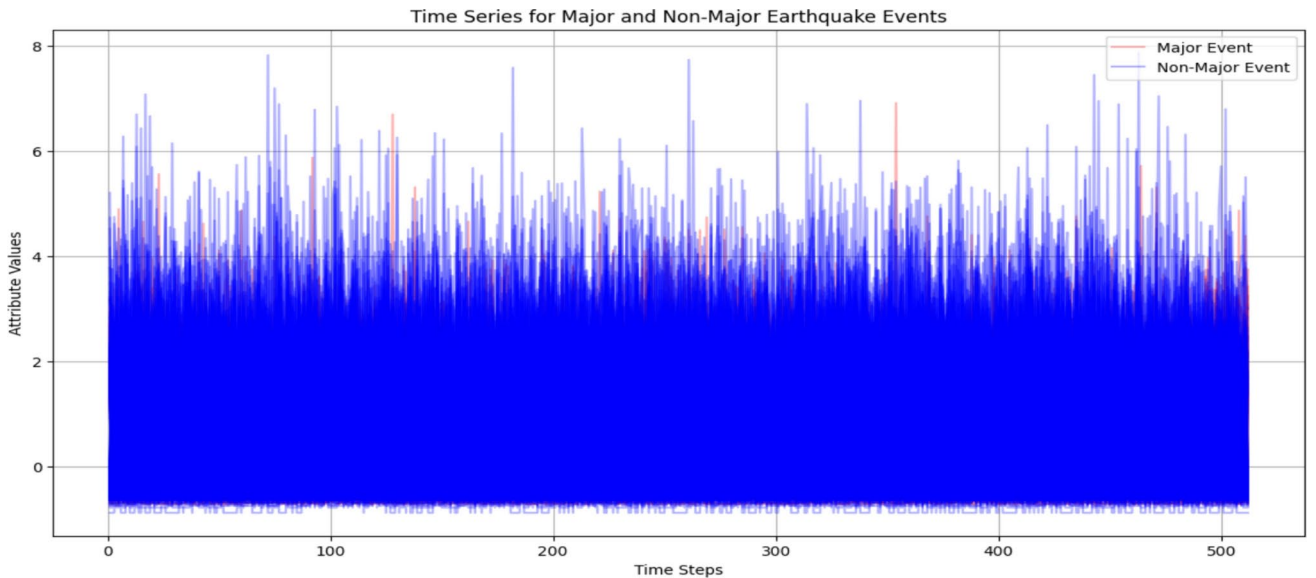
3.1 Classification of univariate time series datasets

The UCR archive (Yanping et al. 2015) publicly available time series datasets. These datasets are differentiated according to the number of categories, dataset type, number of samples and length. The datasets are categorized as sensor and simulated. Their characteristics are summarized in Table 1.

Here Fig. 3 show the earthquake classification using time series data, with high-magnitude major events in red and non-major events in blue. Figure 4 shows the synthetic control chart patterns as normal, cyclic, increasing trend, decreasing trend, upward shift and downward shift.

3.2 Data pre-processing

The data undergo preprocessing must be clean and normalized to the same scale before training to ensure consistency and improve model performance. The min-max scalar is used to normalize data as:

**Fig. 3** Time series plot for major and non-major seismic earthquake events

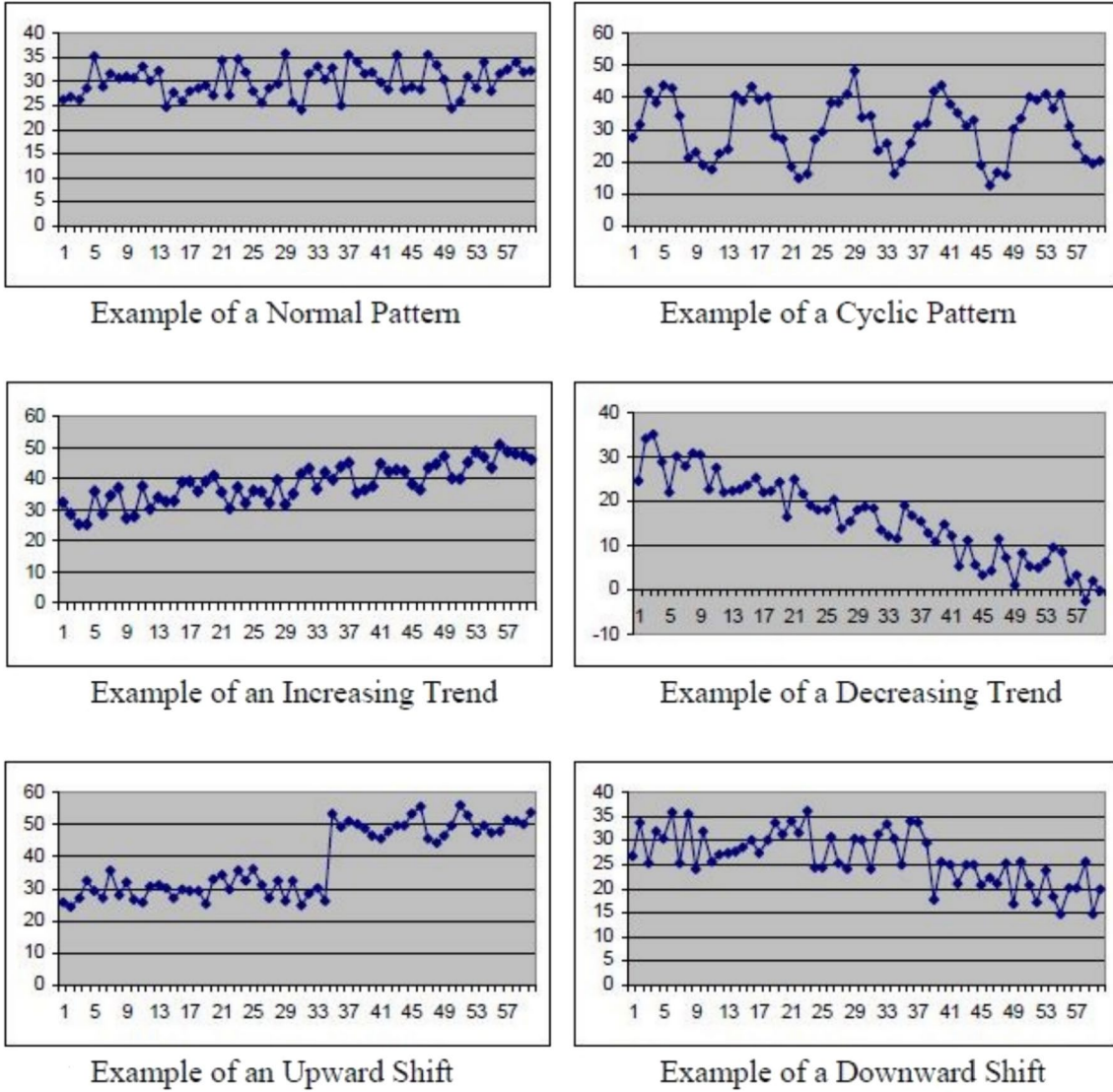


Fig. 4 Synthetic control chart pattern

$$Y' = \frac{Y - Y_{\min}}{Y_{\max} - Y_{\min}} \quad (35)$$

This transformation ensures that all features are within range $[0, 1]$, improving convergence during training. The proposed Bayesian deep Bi-LSTM (Variational Bayesian dropout and VAE), sequential deep learning and traditional methods have been tested on two datasets in Sect. 4. The optimal hyperparameters and number of deep Bi-LSTM, Bi-LSTM cells for data was found via Grid search 16, 32, 64, 128, dropout rates $[0.25-0.5]$ for Bayesian uncertainty estimation. During training and testing phase the number of epochs 100–200 using batch size of 16, 32. To evaluate the model robustly, we performed tenfold stratified cross-validation on the training set. Performance metrics (accuracy, precision, recall, F1-score and AUC-ROC) were averaged

across all folds to provide a robust estimate of model performance. For imbalanced datasets, stratification assures that each fold has the same class distribution as the original dataset. SMOTE balances class distribution in imbalanced datasets. Before SMOTE the original class distribution of D1 is $(\{0: 368, 1: 93\})$ and after SMOTE the class distribution becomes $(\{1: 368, 0: 368\})$ ensure a balanced dataset. All models are trained using the Adam optimizer (Kingma and Ba 2015) with initial $1e^{-3}$ and final learning rates $1e^{-4}$. The classification model was optimized using categorical cross-entropy loss function. We stopped early after patience 10 epochs to avoid overfitting. Predictive uncertainty was estimated using prediction mean and standard deviation.

3.3 Class imbalanced SMOTE method

SMOTE addresses training data class imbalance. SMOTE creates minority class synthetic samples to balance class distribution. An imbalanced dataset can lead to a biased model that performs poorly on underrepresented classes, making this strategy useful in machine learning. Interpolating minority class samples creates synthetic samples with SMOTE. Non-duplicate synthetic samples are generated by this interpolation along the minority class samples' k -nearest neighbors. The nearest neighbors for synthetic points are determined via k -neighbors. SMOTE manages synthetic sample variety by modifying k -neighbors, improving class decision bounds (Douzas et al. 2018).

Given a minority class sample \mathbf{Y}_i , SMOTE randomly selects one of its k -nearest neighbors, $\mathbf{Y}_{z_i}, \forall i \in (1, 2, \dots, n)$ and n is the number of samples in the minority class. A synthetic sample \mathbf{Y}_{new} is then generated as:

$$\mathbf{Y}_{new} = \mathbf{Y}_i + \xi (\mathbf{Y}_{z_i} - \mathbf{Y}_i) \quad (36)$$

Here \mathbf{Y}_i is an original minority class sample, \mathbf{Y}_{z_i} is a randomly selected neighbor of \mathbf{Y}_i and ξ is a random number between 0 and 1.

4 Results and discussion

The proposed Bayesian deep Bi-LSTM model with variational Bayesian dropout and VAE are applied on both datasets. To compare results effectiveness of four sequential deep learning and traditional machine learning models on UCR dataset, respectively. Model evaluation ensures the reliability of predictive models by assessing performance on unseen data. Cross-validation, a resampling method, trains and tests a model on different data subsets to reduce overfitting (model performs well on training data but poorly on new, unseen data) and underfitting. The dataset has k folds, with $k-1$ folds used for training and k folds used for validation. The results are averaged to provide a robust performance estimate as in Table 2. Cross-validation maximizes data usage and provides a more accurate measure of out-of-sample error, making it essential for reliable model evaluation, especially with limited data. Stratified k -fold cross-validation retains class proportions inside each fold. This is crucial for binary or multiclass classification problems with class

imbalance. Cross-validation shows that the model performs well (high mean score, low standard deviation), suggesting it would perform well on unseen data. Deep Bi-LSTM with variational Bayesian dropout is the better-performing model overall, especially on D2, where it achieves near-perfect metrics. The VAE performs well on D1 but is outperformed on D2. The choice of model may depend on the dataset. The specific comparison between baseline models as LSTM, LSTM-FCN, Bi-LSTM, Bi-LSTM Attention, ED, DTW, TSF and COTE and proposed models is given in Table 3. The metrics reported are Accuracy, Precision, Recall, and F1-score.

It can be seen that for dataset 1 (D1), traditional machine learning models, such as ED (68.35%) and DTW (69.06%), show moderate accuracy, while ensemble-based models like TSF (79.56%) and COTE (79.66%) perform slightly better. Sequential deep learning models outperform traditional methods, with LSTM (83.33%), LSTM-FCN (87.04%), and Bi-LSTM (85.19%) showing significant improvement. However, Bi-LSTM Attention (75.00%) performs worse than standard Bi-LSTM. The proposed Bayesian deep learning models achieve the highest accuracy, particularly Bayesian Bi-LSTM (87.04%), and Bayesian deep Bi-LSTM with variational Bayesian dropout with SMOTE (88.89%), outperforming all other models. The VAE with SMOTE (87.50%) also shows strong performance. For dataset 2 (D2), traditional machine learning models perform well, with DTW (96.33%) achieving the highest accuracy among them. Sequential deep learning models further improve performance, with Bi-LSTM (93.33%) and LSTM-FCN (92.50%) achieving strong results. The proposed Bayesian deep learning models outperform all others, with Bayesian deep Bi-LSTM (98.33%) achieving the highest accuracy, followed by the VAE with SMOTE (97.50%). The Bayesian Bi-LSTM (90.83%) and variational Bayesian dropout model (98.33%) also show superior performance compared to traditional and sequential deep learning models.

Bayesian uncertainty quantification further enhances performance by reducing overconfidence in predictions. The VAE component improves robustness by reducing high-dimensional noise. Overall, Bayesian deep learning models demonstrate the best performance for both datasets, particularly when combined with variational Bayesian dropout with SMOTE, which significantly boosts classification accuracy. The results highlight the effectiveness of Bayesian deep learning models in handling time series classification

Table 2 tenfold cross validation performance analysis of Bayesian Deep Bi-LSTM

| Tenfold cross validation models | | | Accuracy | Precision | Recall | F1-score | AUC |
|---------------------------------|------------------------------|----|---------------|---------------|---------------|---------------|---------------|
| Bayesian deep Bi-LSTM | Variational Bayesian Dropout | D1 | 0.8050±0.0926 | 0.8227±0.0950 | 0.8050±0.0926 | 0.8016±0.0947 | 0.8860±0.0788 |
| | | D2 | 0.9600±0.0318 | 0.9656±0.0271 | 0.9600±0.0318 | 0.9595±0.0324 | 0.9967±0.0047 |
| | Variational autoencoders | D1 | 0.8250±0.0848 | 0.8601±0.0896 | 0.8677±0.0782 | 0.8062±0.0621 | 0.8486±0.0858 |
| | | D2 | 0.8750±0.0336 | 0.8873±0.0301 | 0.8750±0.0332 | 0.8740±0.0341 | 0.9843±0.0076 |

Table 3 Performance analysis of Bayesian deep Bi-LSTM and comparison with baseline models

| Models | | Datasets | Accuracy | Precision | Recall | F1-score | | |
|---------------------------------------------|-----------------------------------------------|----------------------------------------------|--------------|-----------|--------|----------|--------|--------|
| Traditional machine learning | Euclidean distance (ED) | D1 | 0.6835 | 0.6274 | 0.6835 | 0.6479 | | |
| | | D2 | 0.8917 | 0.9049 | 0.8917 | 0.8739 | | |
| | Dynamic time warping (DTW) | D1 | 0.6906 | 0.6106 | 0.6906 | 0.6385 | | |
| | | D2 | 0.9633 | 0.9663 | 0.9633 | 0.9632 | | |
| | Time series forest (TSF) | D1 | 0.7956 | 0.7331 | 0.7956 | 0.7051 | | |
| | | D2 | 0.9250 | 0.9247 | 0.9250 | 0.9242 | | |
| | Collective of transformation ensembles (COTE) | D1 | 0.7966 | 0.7361 | 0.7966 | 0.7061 | | |
| | | D2 | 0.9001 | 0.9006 | 0.9000 | 0.8976 | | |
| | Sequential deep learning | Long-short-term-memory (LSTM) | D1 | 0.8333 | 0.8338 | 0.8333 | 0.8333 | |
| | | | D2 | 0.8417 | 0.8900 | 0.8417 | 0.8254 | |
| Fully convolutional network+LSTM (LSTM-FCN) | | D1 | 0.8704 | 0.8835 | 0.8704 | 0.8692 | | |
| | | D2 | 0.9250 | 0.9290 | 0.9250 | 0.9246 | | |
| Bidirectional-LSTM (Bi-LSTM) | | D1 | 0.8519 | 0.8538 | 0.8519 | 0.8516 | | |
| | | D2 | 0.9333 | 0.9427 | 0.9333 | 0.9313 | | |
| Bi-LSTM attention | | D1 | 0.7500 | 0.7600 | 0.7500 | 0.7333 | | |
| | | D2 | 0.8500 | 0.8249 | 0.8500 | 0.8335 | | |
| Bayesian deep learning (Proposed) | Bayesian Bi-LSTM | D1 | 0.8704 | 0.8750 | 0.8704 | 0.8700 | | |
| | | D2 | 0.9083 | 0.9166 | 0.9083 | 0.9085 | | |
| | Bayesian deep Bi-LSTM | Variational Bayesian dropout (without SMOTE) | D1 | 0.8000 | 0.8107 | 0.8000 | 0.8033 | |
| | | | (with SMOTE) | D2 | 0.8889 | 0.9091 | 0.8889 | 0.8875 |
| | Variational Autoencoder (without SMOTE) | D1 | 0.9833 | 0.9837 | 0.9833 | 0.9831 | | |
| | | (with SMOTE) | D1 | 0.8250 | 0.8846 | 0.8519 | 0.8679 | |
| | | | (with SMOTE) | D2 | 0.8750 | 0.8929 | 0.9259 | 0.9091 |
| | | | D2 | 0.9750 | 0.9769 | 0.9756 | 0.9759 | |

Table 4 Epistemic and aleatoric uncertainties for the Bayesian deep Bi-LSTM

| Models | | | Epistemic | Aleatoric | Total Uncertainty |
|-----------------------|-------------|----|-----------|-----------|-------------------|
| Bayesian deep Bi-LSTM | Variational | D1 | 0.009198 | 0.078390 | 0.087588 |
| | Bayesian | D2 | 0.006863 | 0.102680 | 0.109543 |
| Dropout | Variational | D1 | 0.008660 | 0.159582 | 0.168243 |
| | Autoencoder | D2 | 0.003290 | 0.095758 | 0.099048 |

tasks, particularly when combined with data augmentation techniques like SMOTE.

The Table 4 shows aleatoric uncertainty is significantly higher than epistemic uncertainty, indicating that the primary source of uncertainty arises from data variability rather than the model’s confidence in its predictions. Variational Bayesian dropout model demonstrates lower total uncertainty in D1 and D2, suggesting that it provides more stable and reliable predictions compared to the VAE model. D2 has lower aleatoric uncertainty compared to D1, indicating that it is a cleaner and more structured dataset, whereas D1 has more inherent noise, leading to higher overall uncertainty. In

Fig. 5 the model shows promising results with good generalization and no significant signs of overfitting.

In Fig. 6, the model shows good performance on a potentially imbalanced dataset after applying SMOTE. There’s more variability across folds, suggesting some sensitivity to data splits in D1. Model demonstrates outstanding performance with near-perfect classification capabilities. The tenfold cross-validation indicates that this performance is robust and generalizes well across different data splits. The narrow confidence interval suggests a high degree of certainty in the model’s performance estimate. In simpler terms, this visualization tells us that the model is remarkably good at whatever classification task it was trained for. It’s highly accurate, consistent, and reliable in D2.

5 Conclusion

Time series classification is a pivotal task across diverse domains, necessitating models capable of capturing the temporal dependencies and complexities of sequential data. This study focused on UCR, the largest time series

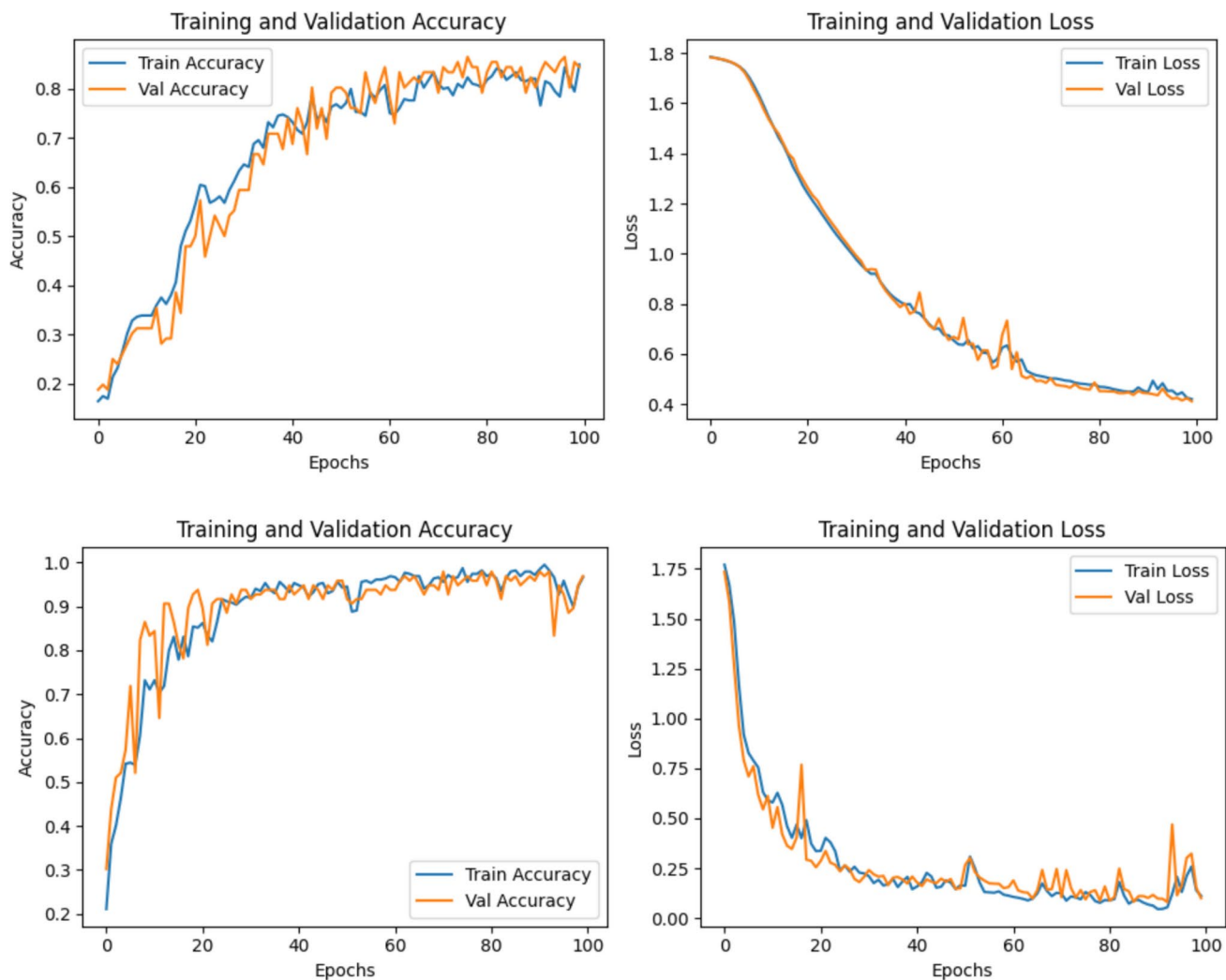


Fig. 5 Visualizing training and validation accuracy and loss curves during model training for D1 and D2

dataset repository, was utilized to demonstrate TSC task applicability.

Traditional models fail to model sequential dependencies and cannot quantify uncertainty. Deep learning models also lack uncertainty estimation. The proposed Bayesian deep Bi-LSTM models provides more robust, interpretable and generalizable performance for both datasets, particularly when combined with variational Bayesian dropout with SMOTE, which significantly boosts classification accuracy. The results highlight the effectiveness of Bayesian deep learning models in handling time series classification tasks, particularly when combined with data augmentation technique SMOTE. Variational Bayesian dropout model demonstrates lower total uncertainty in both datasets, suggesting that it provides more stable and reliable predictions compared to the VAE model. D2 has lower aleatoric uncertainty compared to D1, indicating that it is a cleaner and more structured dataset, whereas D1 has more inherent noise, leading

to higher overall uncertainty. While the proposed model achieves high accuracy, a few limitation remains: computational complexity, limited dataset and advance Bayesian techniques. Future work should focus on expanding dataset coverage, integrating Transformer based architectures and optimizing Bayesian inference techniques.

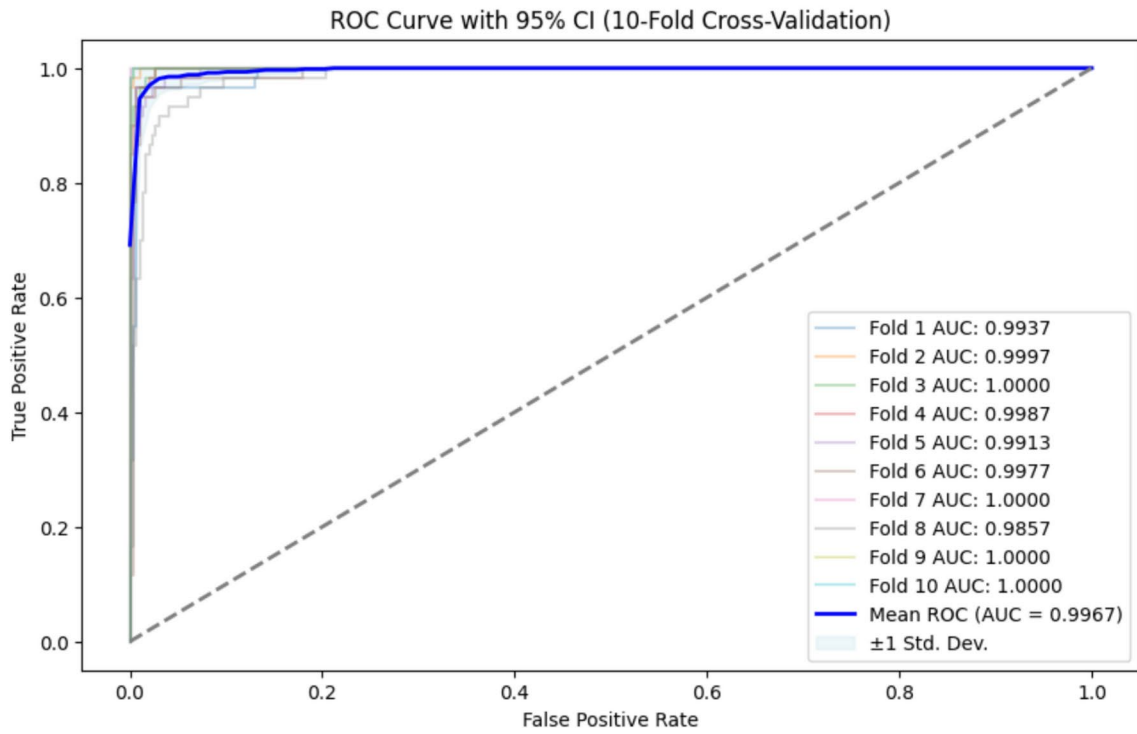
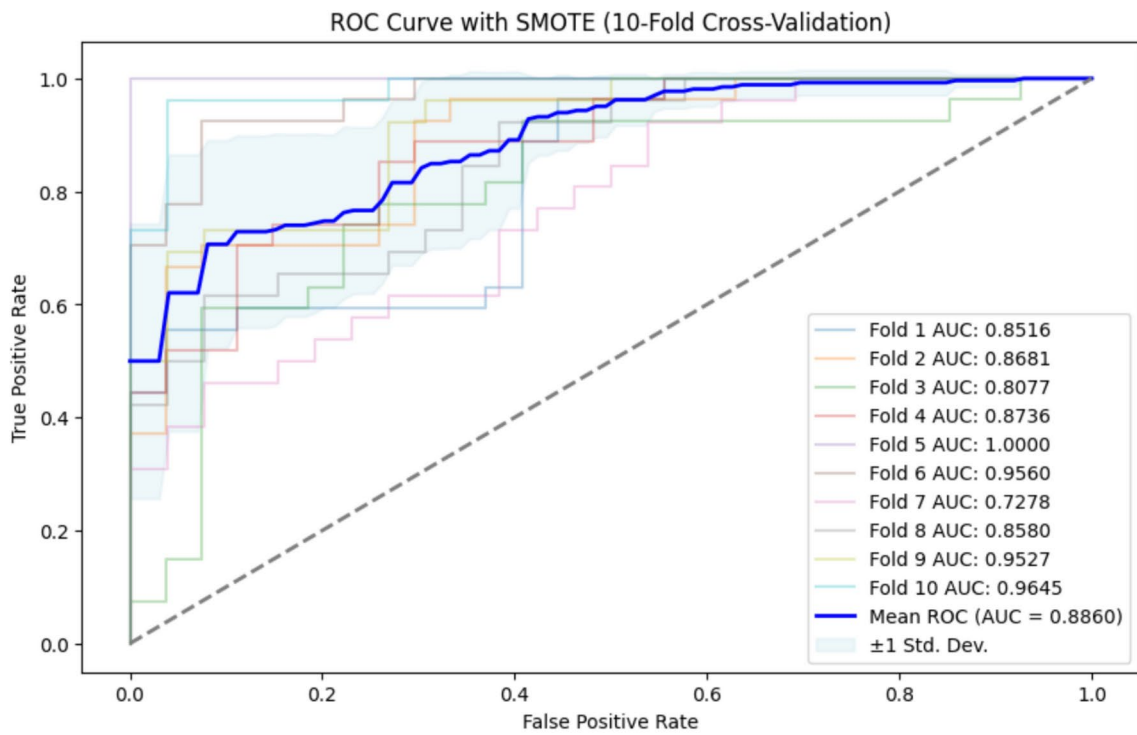


Fig. 6 ROC curve balances true positive and false positive rates. AUC values indicate model performance, with higher values showing class discrimination. The diagonal line reflects random classifier performance. This model predicts well with AUC for D1 and D2

Acknowledgements The authors are grateful to the Deanship of Scientific Research, King Saud University for funding through Vice Deanship of Scientific Research Chairs for empirical data collection and analysis.

Author contributions The paper was plan, managed, data collection, analysis and first write-up was done by Iqra Sardar. Supervision of the work and required resources were provided by Farzana Noor and Muhammad Javed Iqbal. Final write-up, and proofreading done by Ahmed Alsanad, and Muhammad Azeem Akbar.

Funding Not applicable.

Data availability The codes are available under request from the author Iqra Sardar, iqra.phdst13@iu.edu.pk.

Declarations

Conflict of interest The authors declare no conflict of interest.

Ethical approval Not applicable as no human and/ or animal data is involved.

References

- Alcock RJ, Manolopoulos Y (1999) Time-series similarity queries employing a feature-based approach. In 7th Hellenic conference on informatics (pp. 27–29). <https://machinelearning101.pbworks.com/f/TimeSeriesData10.1.1.79.1572.pdf>
- Alwan LC, Roberts HV (1988) Time-series modeling for statistical process control. *J Bus Econ Statist* 6(1):87–95
- Ansari MA et al (2024) ANFIS-based approach to predict sediment removal efficiency of vortex settling basin. *LARHYSS J* 59:193–209
- Antonucci A, De Rosa R, Giusti A, Cuzzolin F (2015) Robust classification of multivariate time series by imprecise hidden Markov models. *Int J Approx Reason* 56:249–263
- Bachhav P, Todisco M, Evans N (2019) Latent representation learning for artificial bandwidth extension using a conditional variational auto-encoder. In: ICASSP IEEE International Conference Acoust Speech Signal Processing, pp. 7010–7014.
- Bagnall A, Lines J, Hills J, Bostrom A (2015) Time series classification with COTE: The collective of transform-based ensembles. *IEEE Trans Knowl Data Eng* 27:2522–2535
- Bagnall A, Lines J, Bostrom A, Large J, Keogh E (2017) The great time series classification bake-off: a review and experimental evaluation of recent algorithmic advances. *Data Min Knowl Discov* 31(3):606–660
- Bagnall A, Lines J, Vickers W, Keogh E (2018) The UEA & UCR time series classification repository. www.timeseriesclassification.com.
- Berndt DJ, Clifford J (1994) Using dynamic time warping to find patterns in time series. In: Proceeding 3rd international conference knowledge discovery and data mining (AAAIWS 94), Seattle, WA, USA, pp. 359–370
- Blundell C, Cornebise J, Kavukcuoglu K, Wierstra D (2015) Weight uncertainty in neural networks. In: International Conference in Machine Learning, pp. 1613–1622
- Chakraborty B (2007) Feature selection and classification techniques for multivariate time series. In: Proceeding Second International Conference Innovative Computing, Information and Control (ICICIC 2007), Kumamoto, Japan, pp. 5–7
- Chen Y, Keogh E, Hu B, Begum N, Bagnall A, Mueen A, Batista G (2015b) The UCR time series classification archive. www.cs.ucr.edu/~eamonn/time_series_data/, Accessed 28 Feb 2019
- Doersch C (2016) Tutorial on variational autoencoders. [arXiv:1606.05908](https://arxiv.org/abs/1606.05908)
- Douzas G, Bacao F, Last F (2018) Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. *J Artif Intell Res* 465:1–20
- Fisher T, Krauss C (2018) Deep learning with long short-term memory networks for financial market predictions. *Eur J Oper Res* 270:654–669
- Gal Y, Ghahramani Z (2016a) Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In: International Conference in Machine Learning, pp. 1050–1059
- Gal Y, Ghahramani Z (2016b) Bayesian convolutional neural networks with Bernoulli approximate variational inference. ICLR Workshop Track. [arXiv:1506.02158](https://arxiv.org/abs/1506.02158)
- Gal Y (2016) Uncertainty in Deep Learning (Ph.D. thesis). University of Cambridge
- Graves A (2011) Practical variational inference for neural networks. *Adv Neural Inf Process Syst*, pp. 2348–2356
- Hernandez-Lobato J, Li Y, Rowland M, Bui T, Hernández-Lobato D, Turner R (2016) Black-box alpha divergence minimization. *Int Conf Machine Learning (PMLR)*, pp. 1511–1520
- Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural Comput* 9(8):1735–1780
- Ismail Fawaz H et al (2019) Deep learning for time series classification: a review. *Data Min Knowl Discov* 33(4):917–963
- Jahangir H, Tayarani H, Gougheri SS, Golkar MA, Ahmadian A, Elkamel A (2020) Deep learning-based forecasting approach in smart grids with micro-clustering and bi-directional LSTM network. *IEEE Trans Ind Electron*. <https://doi.org/10.1109/TIE.2020.3009604>
- Jordan MI, Ghahramani Z, Jaakkola TS, Saul LK (1999) An introduction to variational methods for graphical models. *Mach Learn* 37:183–233
- Karim F et al (2019) Multivariate LSTM-FCNs for time series classification. *Neural Netw* 116:237–245
- Karim F, Majumdar S, Darabi H, Chen S (2017) LSTM fully convolutional networks for time series classification. *IEEE Access*, pp. 1–7
- Kaur D, Islam SN, Mahmud MA (2021) A variational autoencoder-based dimensionality reduction technique for generation forecasting in cyber-physical smart grids. *IEEE Int Conf Commun Workshops (ICCWorkshops)* 1–6
- Kendall A, Gal Y (2017) What uncertainties do we need in Bayesian deep learning for computer vision? *Adv Neural Inf Process Syst* 30:5574–5584
- Kim SB, Han KS, Rim HC, Myaeng SH (2006) Some effective techniques for naive Bayes text classification. *IEEE Trans Knowl Data Eng* 18:1457–1466
- Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In International conference on learning representations
- Kingma DP, Welling M (2013) Auto-encoding variational Bayes. [arXiv preprint arXiv:1312.6114](https://arxiv.org/abs/1312.6114).
- Kini BV, Sekhar CC (2013) Large margin mixture of AR models for time series classification. *Appl Soft Comput* 13:361–371
- Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. *Adv Neural Inf Process Syst* 25:1097–1105
- Kumar V, Azamathulla HM, Sharma KV, Mehta DJ, Maharaj KT (2023) The state of the art in deep learning applications, challenges, and future prospects: a comprehensive review of flood forecasting and management. *Sustainability* 15:10543. <https://doi.org/10.3390/su15131054>

-
- Kuyuk HS, Susumu O (2018) Real-time classification of earthquakes using deep learning. *Procedia Comput Sci* 140:298–305
- Lal TN, Schroder M, Hinterberger T, Weston J, Bogdan M, Birbaumer N, Scholkopf B (2004) Support vector channel selection in BCI. *IEEE Trans Biomed Eng* 51:1003–1010
- Lara OD, Labrador M (2013) A survey on human activity recognition using wearable sensors. *IEEE Commun Surv Tutor* 15:1192–1209
- LeCun Y, Bengio Y, Hinton G (2015) Deep learning. *Nature* 521:436–444
- Lines J, Bagnall A (2015) Time series classification with ensembles of elastic distance measures. *Data Min Knowl Discov* 29:565–592
- Lines J, Taylor S, Bagnall A (2018) Time series classification with HIVE-COTE: the hierarchical vote collective of transformation-based ensembles. *ACM Trans Knowl Discov Data* 12(5):1–35
- Neamtu R, Ahsan R, Rundensteiner EA, Sarkozy G, Keogh E, Dau HA, Nguyen C, Lovering C (2018) Generalized dynamic time warping: unleashing the warping power hidden in point-wise distances. *IEEE Int Conf Data Eng*, pp. 521–532
- Ogawa A, Hori T (2017) Error detection and accuracy estimation in automatic speech recognition using deep bidirectional recurrent neural networks. *Speech Commun* 89:70–83. <https://doi.org/10.1016/j.specom.2017.02.009>
- Pawlowski N, Dilokthanakul N, Shanahan M (2018) Explicit information placement on latent variables using auxiliary generative modeling task
- Puri D et al (2024) Analysis of data splitting on streamflow prediction using random forest. *AIMS Environ Sci* 11(4):593–609
- Singh D, Merdivan E, Psychoula I, Kropf J, Hanke S, Geist M, Holzinger A (2017) Human activity recognition using recurrent neural networks. In: *Machine learning and knowledge extraction. Lecture Notes in Computer Science LNCS 10410*. Springer/Nature, London, pp. 267–274
- Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. *IEEE Conf Comput Vis Pattern Recognit* 1–9.
- Toubeau JF, Bottieau J, Vallée F, De Grève Z (2018) Deep learning-based multivariate probabilistic forecasting for short-term scheduling in power markets. *IEEE Trans Power Syst* 34:1203–1215
- Wang X, Mueen A, Ding H, Trajcevski G, Scheuermann P, Keogh E (2013a) Experimental comparison of representation methods and distance measures for time series data. *Data Min Knowl Discov* 26:275–309
- Wang J, Liu P, She MFH, Nahavandi S, Kouzani A (2013b) Bag-of-words representation for biomedical time series classification. *Biomed Signal Process Control* 8:634–644
- Wang Z, Yan W, Oates T (2017) Time series classification from scratch with deep neural networks: a strong baseline. In: *International Joint Conference on Neural Networks*, pp. 1578–1585
- Wang X, Cui P, Du Y, Yang Y (2020) Variational autoencoder-based fault detection and location method for power distribution network. *Int Conf Cond Monit Diagn (CMD)*, pp. 282–285
- Xia L, Tang J, Li G et al (2024) Time series classification based on forward echo state convolution network. *Neural Process Lett* 56:173. <https://doi.org/10.1007/s11063-024-11449-8>
- Yanping C, Keogh E, Hu B, Begum N, Bagnall A, Mueen A, Batista G (2015) The UCR time series classification archive
- Ye L, Keogh E (2009) Time series shapelets: A new primitive for data mining. In: *Proceedings on ACM SIGKDD International Conference Knowledge Discovery and Data Mining*, Paris, France, pp. 947–956
- Yoon H, Yang K, Sahabi C (2005) Feature subset selection and feature ranking for multivariate time series. *IEEE Trans Knowl Data Eng* 17:1186–1198
- Zhang C, Bütepage J, Kjellström H, Mandt S (2018) Advances in variational inference. *IEEE Trans Patt Anal Mach Intell* 41:2008–2026
- Zhang Q et al (2022) A watershed water quality prediction model based on attention mechanism and Bi-LSTM. *Environ Sci Pollut Res* 29(50):75664–75680
- Zhang C, Bengio S, Hardt M, Recht B, Vinyals O (2017) Understanding deep learning requires rethinking generalization. In: *International Conference Learn Represent*