Deep Generative model with Hierarchical Latent Factors for Time Series Anomaly Detection

Cristian Challu CMU
Peihong Jiang Amazon Research
Ying Nian Wu UCLA
Laurent Callot Amazon Research

Abstract

Multivariate time series anomaly detection has become an active area of research in recent years, with Deep Learning models outperforming previous approaches on benchmark datasets. Among reconstruction-based models, most previous work has focused on Variational Autoencoders and Generative Adversarial Networks. This work presents DGHL, a new family of generative models for time series anomaly detection, trained by maximizing the observed likelihood by posterior sampling and alternating back-propagation. A top-down Convolution Network maps a novel hierarchical latent space to time series windows, exploiting temporal dynamics to encode information efficiently. Despite relying on posterior sampling, it is computationally more efficient than current approaches, with up to 10x shorter training times than RNN based models. Our method outperformed current state-of-the-art models on four popular benchmark datasets. Finally, DGHL is robust to variable features between entities and accurate even with large proportions of missing values, settings with increasing relevance with the advent of IoT. We demonstrate the superior robustness of DGHL with novel occlusion experiments in this literature. Our code is available at `https://github.com/cchallu/dghl`.

1 INTRODUCTION

Recent advancements in Deep Learning such as Recurrent Neural Networks (RNN), Temporal Convolution Networks (TCN) and Graph Networks (GN) have been successfully incorporated by recent models to outperform previous approaches such as out-of-limits, clustering-based, distance-based, and dimensionality reduction on a wide range of tasks. (Chalapathy and Chawla, 2019) (Ruff et al., 2021) present comprehensive reviews of current state-of-the-art methods for time-series anomaly detection.

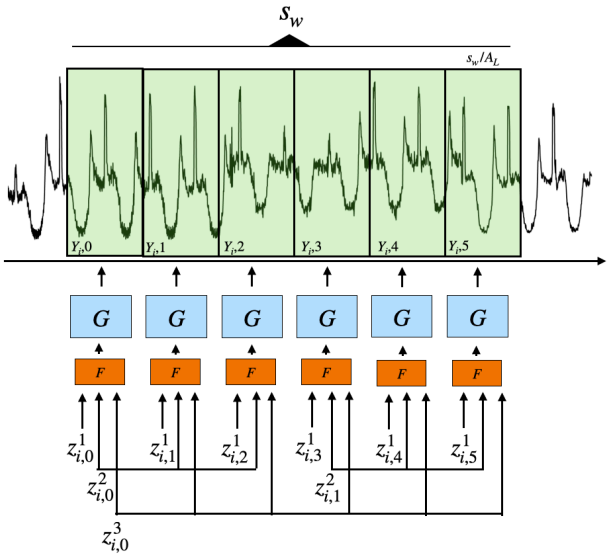

Figure 1: Example of hierarchical latent factor space for $\boldsymbol{a} = [1, 3, 6]$. On each level l, the latent vectors of a_l consecutive sub-windows are tied. For instance, the latent vector on the highest layer, L, is shared by all sub-windows of $\boldsymbol{Y}$.

In this work, we propose DGHL, a novel Deep Generative model based on a top-down Convolution Network (ConvNet), which maps multivariate time-series windows to a novel hierarchical latent space. The model is trained by maximizing the observed likelihood directly with the Alternating Back-Propagation algorithm (Han et al., 2017), so it does not rely on auxiliary networks such as encoders or discriminators

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

as VAEs and GANs do. DGHL, therefore, comprehends a separate family of generative models, previously unexplored for time-series anomaly detection. We perform experiments on several popular datasets and show the proposed model outperforms the recent state-of-the-art while reducing training times against previous reconstruction-based and generative models.

With the advent of IoT, we believe that settings with corrupted or missing data have increasing relevance. For example, faulty sensors can cause missing values, privacy issues on consumer electronics devices, or heterogeneous hardware can lead to variable features. We present the first extensive analysis on the robustness of current state-of-the-art models on datasets with missing inputs and variable features with novel occlusion experiments. DGHL achieved superior performance on this setting, maintaining state-of-the-art performance with up to 90% of missing data, without modification to the architecture or training procedure. We perform additional qualitative experiments of our model to assess desirable properties of lower-dimensional representations such as continuity and extrapolation capabilities. Finally, we show how DGHL can be used as a forecasting model, demonstrating its versatility on various time-series tasks.

The main **contributions** of our paper are:

- **Short-run MCMC**. First time-series anomaly detection generative model based on short-run MCMC for estimating posterior of latent variables and inferring latent vectors. In particular, first application of Alternating Back-Propagation algorithm for learning generative model for time-series data.
- **Hierarchical latent factors**. We present a novel hierarchical latent space representation to generate windows of arbitrary length. We demonstrate with ablation studies how DGHL achieves state-of-the-art performance by leveraging this representation on four benchmark datasets.
- **Robustness to missing data**. We present the first experiments on robustness to missing inputs of state-of-the-art anomaly detection models, and demonstrate DGHL achieves superior performance in this setting.
- **Open-source implementation**. We publish an open implementation of our model and full experiments for reproducibility of the results.

The rest of the paper is structured as follows. Section 2 reviews current state-of-the-art models, Section 3 introduces DGHL and describes the Alternating Back-Propagation algorithm, Section 4 contains the empirical findings. In section 5 we present a discussion of the main findings. Finally in Section 6 we wrap up and conclude.

2 RELATED WORK

2.1 Reconstruction-based models

Reconstruction-based models learn representations for the time-series by reconstructing the input based on latent variables. The reconstruction error or the likelihood are commonly used as anomaly scores. Among these models, variational auto-encoders (VAE) are the most popular. The LSTM-VAE, proposed in (Park et al., 2018), uses LSTM both as encoders and decoders and models the reconstruction error with support vector regression (SVR) to have a dynamic threshold based on the latent space vector. Omni-Anomaly (Su et al., 2019) improves on the LSTM-VAE by adding normalizing planar flows to increase the expressivity and including a dynamic model for the latent space.

Generative Adversarial Networks (GANs) were also adapted for anomaly detection as alternatives to VAE, with models such as AnoGAN (Schlegl et al., 2017), MAD-Li (Li et al., 2018), and MAD-GAN (Li et al., 2019). For instance, in MAD-GAN, a GAN is used to generate short windows of time-series with LSTM Generator and Discriminator networks. The anomaly score considers both the reconstruction error of the reconstructed window by the Generator network and the score of the Discriminator network.

MTAD-GAT (Zhao et al., 2020) proposed to combine both forecasting and reconstruction approaches. It includes a Multi Layer Perceptron (MLP) for forecasting and a VAE for reconstructing the time-series, with the anomaly score including both forecasting and reconstruction loss terms. It also includes two Graph Attention Networks (GAT) to model temporal dynamics and correlations explicitly. (Deng and Hooi, 2021) later proposed GDN, which models the relationships between time-series with a Graph Neural Network and uses a GAT for forecasting.

Most recent models propose to detect anomalies directly on the latent representation and embeddings. THOC (Shen et al., 2020) proposed to use one-class classifiers based on multiple hyperspheres on the representations on all intermediate layers of a dilated RNN. NCAD (Carmona et al., 2021) uses a TCN to map context windows and suspect windows into a neural representation and detect anomalies in the suspect window on the latent space with a contextual hypersphere loss.

2.2 Generative models with Alternating Back-Propagation

Virtually all current models, including our proposed approach, rely on mapping the original time-series input into embeddings or a lower-dimensional latent space. DGHL, however, is trained with the Alternating Back-Propagation (ABP) algorithm, presented in (Han et al., 2017). ABP maximizes the observed likelihood directly; it does therefore not rely on variational inference approximations or auxiliar networks such as discriminators. Instead, our approach uses MCMC sampling methods to sample from the true posterior to approximate the likelihood gradient.

Several generative models which rely on MCMC sampling, and in particular Langevin Dynamics, have shown state-of-the-art performance on computer vision (Pang et al., 2020) and NLP (Pang et al., 2021) tasks. To our knowledge, this algorithm has not been used for time-series forecasting and time-series anomaly detection. We present the ABP algorithm in more detail in subsection 3.2.

3 DGHL

3.1 Hierarchical Latent Factors

Let $\boldsymbol{Y} \in \mathbb{R}^{m \times s_w}$ be a window of size s_w of a multivariate time-series with m features. The window $\boldsymbol{Y}$ is further divided in sub-windows of equal length $\boldsymbol{Y_j} \in \mathbb{R}^{m \times \frac{s_w}{a_L}}, j = 0, ..., a_L$. The structure of the hierarchy is specified by $\boldsymbol{a} = [a_1, ..., a_L]$, where L is the number of levels, and a_l determines the number of consecutive sub-windows with shared latent vector on level l, with $a_l \mid a_L$. Our model for each sub-window $\boldsymbol{Y_j}$ of $\boldsymbol{Y}$ is given by,

$$\begin{aligned} \boldsymbol{s_j} &= F_{\boldsymbol{\alpha}}(\boldsymbol{z}^1_{\lfloor \frac{j}{a_1} \rfloor}, ..., \boldsymbol{z}^L_{\lfloor \frac{j}{a_L} \rfloor}) \\ \boldsymbol{Y_j} &= G_{\boldsymbol{\beta}}(\boldsymbol{s_j}) + \boldsymbol{e_j} \end{aligned} \tag{1}$$

where F_α is the *State model*, G_β is the *Generator model*, $\boldsymbol{\theta} = [\boldsymbol{\alpha}, \boldsymbol{\beta}]$ are the parameters, $\boldsymbol{s_j} \in \mathbb{R}^d$ is the state vector, $\boldsymbol{e_j} \sim N(0, \boldsymbol{I}_D)$, and

$$\boldsymbol{Z} = \{\boldsymbol{z}^l_{\lfloor \frac{j}{A_l} \rfloor} \in \mathbb{R}^{d_l}\}_{l,j} \tag{2}$$

is the hierarchical latent factor space for window $\boldsymbol{Y}$. For the *State model* we used a concatenation layer. For the *Generator model* we used a top-down Convolution Network (ConvNet), shown in Figure 2, which maps an input state vector to a multivariate time-series window. The *State model* for each sub-window has L latent vectors inputs. On each level l, the latent vectors of a_l consecutive sub-windows are tied. For instance, the latent vector on the highest layer, L, is shared by all sub-windows of $\boldsymbol{Y}$. Figure 1 shows an example of a hierarchical latent space with $\boldsymbol{a} = [1, 3, 6]$.

The key principle of the hierarchical latent space is to leverage dynamics on the time-series, such as seasonalities, to encode the information on the latent space more efficiently, with lower-dimensional vectors. The hierarchical latent space allows generating realistic time-series of arbitrary length while preserving their long-term dynamics. The hierarchical structure can be incorporated as hyper-parameters to be tuned or pre-defined based on domain knowledge. For instance, hierarchies can correspond to the multiple known seasonalities on the time-series.

The hierarchical latent space $\boldsymbol{Z}$ is jointly inferred using Langevin Dynamics. The relative size of the lowest level state vector and the upper levels controls the flexibility of the model. Larger lower hierarchy level vectors make the model more flexible, making it robust to normal changes or randomness in long-term dependencies of the time-series and therefore reducing false positives by reducing the reconstruction error. Larger tied vectors will make the model more strict, better for detecting contextual anomalies. The model presented in (Han et al., 2017) can be seen as a single level hierarchical latent space model, with $\boldsymbol{a} = [1]$, in the current framework.

Previous work such as OmniAnomaly incorporates transition models to learn dynamics in the latent space. We believe our proposed hierarchical latent factors structure has several advantages over transition models. First, the computational cost and training time is lower for the proposed model since it does not rely on sequential computation and therefore on back-propagation through time for training parameters. Second, transition models implicitly assume the dynamics are constant over time, a non-realistic assumption in many settings. Our solution allows the model to share information across windows to model long-term dynamics without relying on a parametric model which assumes constant dynamics.

3.2 Training with Alternating Back-Propagation

The parameters $\boldsymbol{\theta}$ of DGHL are learned with the Alternating Back-Propagation algorithm. First, the training multivariate time-series $\boldsymbol{Y} \in \mathbb{R}^{m \times T}$ with m features and T timestamps, is divided in consecutive windows of size s_w and step size s in a rolling-window fashion. Let $\{\mathbf{Y}^{(i)}, i = 1, ..., n\}$ be the training set of time-series windows. Alternating Back-Propagation algorithm learns parameters $\boldsymbol{\theta}$ by maximizing the ob-

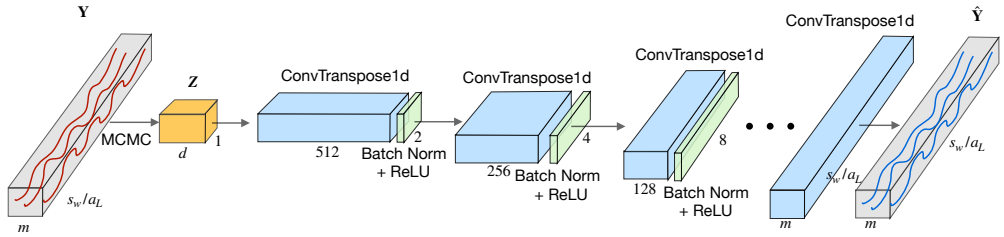

Figure 2: *Generator* architecture, which maps a latent vector $\boldsymbol{Z}$ to a time-series window. Each layer of the ConvNet increases the temporal dimension and reduces the filters by a factor of 2. A batch normalization layer and ReLU activations are included between each convolutional layer.

served log-likelihood directly, given by,

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log p_{\boldsymbol{\theta}}(\mathbf{Y}^{(i)}) = \sum_{i=1}^{n} \log \int p_{\boldsymbol{\theta}}(\mathbf{Y}^{(i)}, \mathbf{Z}^{(i)}) d\mathbf{Z}^{(i)} \tag{3}$$

where $\mathbf{Z}^{(i)}$ is the latent vector for window i specified in equation 2. The observed likelihood $L(\boldsymbol{\theta})$ is analytically intractable. However, the gradients $L'(\boldsymbol{\theta})$ for a particular observation can be simplified to,

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathbf{Y}^{(i)}) &= \frac{1}{p_{\boldsymbol{\theta}}(\mathbf{Y}^{(i)})} \frac{\partial}{\partial \boldsymbol{\theta}} \int p_{\boldsymbol{\theta}}(\mathbf{Y}^{(i)}, \mathbf{Z}^{(i)}) d\mathbf{z} \\ &= \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{Z}^{(i)}|\mathbf{Y}^{(i)})} \left[\frac{\partial}{\partial \boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathbf{Y}^{(i)}, \mathbf{Z}^{(i)}) \right] \end{aligned} \tag{4}$$

where $p_{\boldsymbol{\theta}}(\mathbf{Z}^{(i)}|\mathbf{Y}^{(i)}) = p_{\boldsymbol{\theta}}(\mathbf{Y}^{(i)}, \mathbf{Z}^{(i)}) / p_{\boldsymbol{\theta}}(\mathbf{Y}^{(i)})$ is the posterior. The expectation in the previous equation can be approximated with the Monte Carlo average by taking samples using MCMC. In particular, Alternating Back-Propagation takes a single sample of the posterior using Langevin Dynamics (Neal et al., 2011), a Hamiltonian Monte Carlo algorithm, which iterates,

$$\begin{aligned} \mathbf{Z}_{t+1}^{(i)} &= \mathbf{Z}_{t}^{(i)} + \frac{s}{\sigma_z} \frac{\partial}{\partial \mathbf{Z}^{(i)}} \log p_{\boldsymbol{\theta}}(\mathbf{Z}_{t}^{(i)}|\mathbf{Y}^{(i)}) + \sqrt{2s}\epsilon_t \\ &= \mathbf{Z}_{t}^{(i)} + \sqrt{2s}\epsilon_t + \\ &\frac{s}{\sigma_z} \left[(\mathbf{Y}^{(i)} - f(\mathbf{Z}_{t}^{(i)}, \boldsymbol{\theta})) \frac{\partial}{\partial \mathbf{Z}^{(i)}} f(\mathbf{Z}_{t}^{(i)}, \boldsymbol{\theta}) - \mathbf{Y}_{t}^{(i)} \right] \end{aligned} \tag{5}$$

where $\boldsymbol{\epsilon}_t \sim N(0, \boldsymbol{I}_D)$, t is the time step of the dynamics, s is the step size, and σ_z controls the relative size of the injected noise. This iteration is an explain-away process where latent factors are chosen such that the current residual on the reconstruction, $\mathbf{Y}^{(i)} - f(\mathbf{Z}_{t}^{(i)}, \boldsymbol{\theta})$, is minimized. With large values of σ_z, the posterior will be close to the prior, while small σ_z allows for a richer posterior. The iterative process is truncated to a predefined number of iterations, and the rejection step is not considered. As explained in (Neal et al., 2011), for an observation $\mathbf{Y}^{(i)}$, the resulting vector is a sample from an approximated posterior, $p_{\boldsymbol{\theta}}(\mathbf{Z}^{(i)}|\mathbf{Y}^{(i)})$. The Monte Carlo approximation of the gradient then becomes,

$$\begin{aligned} L'(\boldsymbol{\theta}) &\approx \frac{\partial}{\partial \boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathbf{Z}^{(i)}, \mathbf{Y}^{(i)}) \\ &= \frac{1}{\sigma^2} (\mathbf{Y}^{(i)} - f(\mathbf{Z}^{(i)}, \boldsymbol{\theta})) \frac{\partial}{\partial \boldsymbol{\theta}} f(\mathbf{Z}^{(i)}, \boldsymbol{\theta}) \end{aligned} \tag{6}$$

Algorithm 1 presents the alternating back-propagation algorithm with mini-batches. Analogous to the EM algorithm (Dempster et al., 1977), it iterates two distinct steps: (1) *inferential back-propagation* and (2) *learning back-propagation*. During (1), the latent vectors $\{\mathbf{Z}^{(i)}\}$ are inferred for a sample $\{\mathbf{Y}^{(i)}, i = 1, ..., b_s\}$. In step (2), $\{\mathbf{Z}^{(i)}\}$ are used as input of the Generator model f and parameters $\boldsymbol{\theta}$ are updated with SGD. We use Adam optimizer for learning parameters with default parameters (Kingma and Ba, 2014).

One disadvantage of MCMC methods is the computational cost. Langevin Dynamics, however, relies on the gradients of the Generator function, which can be efficiently computed with modern automatic differentiation packages such as Tensorflow and PyTorch. Moreover, back-propagation on ConvNet is easily parallelizable in GPU. Several recent works, have shown models trained with alternating back-propagation (Han et al., 2017), (Xie et al., 2019) and short-run MCMC (Nijkamp et al., 2021) achieved state-of-the-art performance in computer vision and NLP tasks while remaining computationally efficient and comparable in training time to methods relying solely on SGD. We discuss training and inference times on section 4.

Algorithm 1: Mini-batch Alternating back-propagation

input : training examples $\{\mathbf{Y}^{(i)}, i = 1, ..., n\}$, Langevin steps l, learning iterations T
output: learned parameters $\boldsymbol{\theta}$, inferred latent vectors $\{\mathbf{Z}^{(i)}, i = 1, ..., n\}$

Let $t \leftarrow 0$, initialize $\boldsymbol{\theta}$
Initialize $\mathbf{Z}^{(i)}$, for $i = 1, ..., n$
while $t < T$ **do**
  Take a random mini-batch $\{\mathbf{Y}^{(j)}, j = 1, ...b\}$.
  Inferential back-propagation: For each j, run l steps of Langevin dynamics to sample $\mathbf{Z}^{(j)} \sim p(\mathbf{Z}|\mathbf{Y}^{(j)}, \boldsymbol{\theta})$ following equation 5.
  Learning back-propagation: Compute gradients following equation 6 and update $\boldsymbol{\theta}$.
  $t \leftarrow t + 1$
end

As described in (Welling and Teh, 2011), Bayesian posterior sampling provides inbuilt protection against overfitting. The MCMC sampling allows DGHL to model complex multivariate time-series, while reducing the risk of overfitting. This is particularly helpful on problems with small training sets.

3.3 Online Anomaly Detection

By learning how to generate time-series windows based on the training data $\boldsymbol{Y}$, DGHL implicitly learns *normal* (non-anomalous) temporal dynamics and correlations between the multiple time-series. In this subsection, we explain the proposed approach to reconstruct windows on unseen test data $\boldsymbol{Y}^{test}$ to detect anomalies.

In Online Anomaly Detection we consider the test set $\boldsymbol{Y}^{test} \in \mathbb{R}^{m \times T_{test}}$ to be a stream of m time-series. The goal is to detect anomalies (the evaluation is equivalent to a supervised setting with two classes) as soon as possible. As with the training set, $\boldsymbol{Y}^{test}$ is first divided in consecutive windows with the same parameters s_w and s. We propose to reconstruct and compute anomalies scores one window at a time.

Let $\boldsymbol{Y_{t^*}}$ be the current window of interest. The latent space $\boldsymbol{Z_{t^*}}$ is jointly inferred to reconstruct the target window, namely $\hat{\boldsymbol{Y}}_{\boldsymbol{t^*}}$. The anomaly score for a particular timestamp t in the window is computed as the Mean Square Error (MSE) considering all m time-series, given by

$$s_t = \frac{1}{m} \sum_{i=1}^{m} (y_{i,t} - \hat{y}_{i,t})^2 \tag{7}$$

The size of the window s_w and step size s control how early anomalies can be detected. With a smaller s, anomaly scores for newer values in the stream are computed sooner. When $s < s_w$, consecutive windows have overlapping timestamps. In this case, scores are updated by considering the average reconstruction. In datasets with multiple entities (for instance, machines in SMD), we scale the scores by the accumulated standard deviation of scores before window t^*.

One main difference with the inference step during training is the removal of the Gaussian noise, ϵ_t, of the Langevin Dynamics update equation. The inferred factors then correspond to the maximum a posteriori mode, which in turn minimizes the reconstruction error conditional on the learned models F and G. This novel strategy makes DGHL unique among reconstruction-based models: it avoids overfitting during training by sampling from the posterior with Langevin Dynamics and minimizes the reconstruction error to reduce false positives by MAP estimation.

Many previous models rely on complex and unusual specific scores, but DGHL uses the simple MSE. The anomaly scores of our approach are interpretable, since they can be disaggregated by the m features. Users can rank the contribution to the anomaly score of each feature to gather insights of the anomaly.

3.4 Online Anomaly Detection with missing data

The first step of the ABP algorithm is to infer latent vectors with Langevin Dynamics. This is an explain-away process where latent factors are chosen such that the current residual on the reconstruction, $\mathbf{Y} - f(\mathbf{Z}_t, \boldsymbol{\theta})$, is minimized. The model can intrinsically deal with missing data by inferring $\mathbf{Z}$ computing the residuals only on the observed signal $\mathbf{Y}_{obs}$. The inferred vectors then correspond to samples from the posterior distribution conditional on the available signal, $p_{\boldsymbol{\theta}}(\mathbf{Z}|\mathbf{Y}_{obs})$. Since no explicit learnable parameters map inputs to the latent space, the model is more robust to missing values and outliers (masked as missing data). Generative models trained with ABP algorithm outperformed VAEs and GANs on experiments with missing information on computer vision and NLP tasks (Han et al., 2017).

Figure 3 shows an example of occluded data, for a subset of features of one machine of the SMD dataset. Occluded segments are marked in gray. First, DGHL is able to precisely reconstruct the observed data (white region), even when most features are missing. This is most relevant for the anomaly detection task, since only the observed features are used to compute the

anomaly score. Second, the model is able to recover missing data with great precision, which can be helpful in complete pipelines with downstream applications.

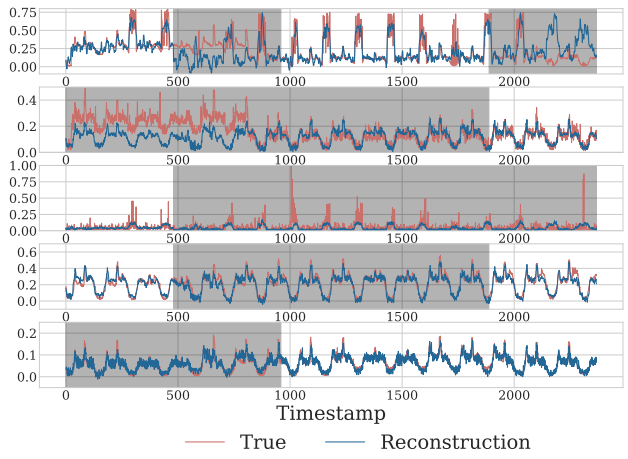

Figure 3: Occlusion experiment on machine-1-1 of the SMD. Red lines correspond to the actual values and blue lines present the reconstructed time-series with DGHL. Gray areas correspond to the occluded information during training.

4 EXPERIMENTS

4.1 Datasets

Server Machine Dataset (SMD) – Introduced in (Su et al., 2019), SMD is a multivariate time-series dataset with 38 features for 28 server machines, monitored during 5 weeks. The time-series include common activity metrics in servers such as CPU load, network and memory usage, among others. Both training and testing sets contain around 50k timestamps each, with 5 % of anomalous cases. We trained separate models for each machine as suggested by the authors but with the same hyperparameters.

Soil Moisture Active Passive satellite (SMAP) and Mars Science Laboratory rover (MSL)– Published by NASA in (Hundman et al., 2018), they contain real telemetric data of the SMAP satellite and MSL rover. SMAP includes 55 multivariate time-series datasets, each containing one anonymized channel and 24 variables encoding information sent to the satellite. MSL includes 27 datasets, each with one telemetry channel and 54 additional variables. Again, we train separate models for each telemetry channel, considering additional variables as exogenous, ie. only the anomaly score of the telemetry channel was used for detecting anomalies.

Secure Water Treatment (SWaT) – Is a public dataset with information of a water treatment testbed meant for cyber-security and anomaly detection research. It contains network traffic and data from 51 sensors for 11 days, 7 days of normal operation (train set) and 4 days with cyber attacks (test set).

4.2 Evaluation

We evaluate the performance of DGHL and benchmark models on the four datasets with the F_1-score, considering the anomaly detection problem as a binary classification task where the positive class corresponds to anomalies. Anomalies often occur continuously over a period of time creating anomalous segments. (Xu et al., 2018) proposed an adjustment approach, where the predicted output is re-labeled as an anomaly for the whole continuous anomalous segment if the model correctly identifies the anomaly in at least one timestamp. We use this adjustment technique for SMAP, MSL and SMD datasets to make results comparable with existing literature. Moreover, we followed the common practice of comparing the performance using the best F_1-score, by choosing the best threshold on the test set. For SMAP, MSL and SMD we use a single threshold through the entire dataset (not different thresholds for each machine or channel).

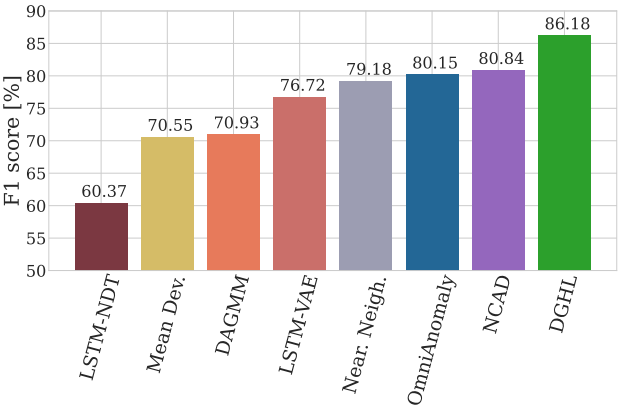

Figure 4: F_1 scores on SMD dataset using a single threshold across all machines.

To make our results comparable with previous work, we follow the train, validation and test split described in (Shen et al., 2020) for SMAP, MSL and SWaT. For SMD we use the train and test splits described in (Su et al., 2019). All architecture hyper-parameters of the *Generator* model, training hyper-parameters such as batch size and learning rate, and all hyper-parameters of the Langevin Dynamics were kept constant across the four datasets.

We compare DGHL to current state-of-the art models, such as THOC (Shen et al., 2020), NCAD (Carmona et al., 2021), and MTAD-GAT (Zhao et al., 2020); and previous widely used models such as AnoGAN (Schlegl et al., 2017), DeepSVDD (Ruff et al., 2018), DAGMM (Zong et al., 2018), OmniAnomaly, (Su et al., 2019), MAD-GAN (Li et al., 2019) and LSTM-VAE (Park et al., 2018). We also include simple one-line and non deep-learning approaches such as *Mean deviation* and *Nearest Neighbors*. *Mean deviation* uses the average deviation to the mean of each feature as anomaly score. The later uses the average distance the k nearest windows of the training set as anomaly score. Finally, we include two additional versions of DGHL removing the key contributions of our work. First, we remove the hierarchical factors ($\mathbf{a} = [1]$), and second, we replace the Langevin Dynamics algorithm for inferring latent factors with a convolutional encoder.

Table 1 shows the F_1 scores for DGHL, and the benchmark models for SMAP, MSL, and SWaT datasets. Our methods consistently achieves the Top-2 F_1 scores, with overall performance superior to state-of-the-art such as MTAD-GAT (Zhao et al., 2020), THOC and NCAD. Moreover, our approach achieved the highest performance between all reconstruction based and generative models on all datasets. Figure 2 shows the F_1 scores for SMD dataset. DGHL achieved a score of 86.18 ± 0.66, outperforming all benchmark models. Our model without hierarchical factors had a score of 80.84 ± 0.40, and with encoder had a score of 76.59 ± 0.78.

DGHL significantly outperformed simple baselines in all datasets. The one-line solution ranked worst consistently. Nearest Neighbors, however, achieved a better performance than several complex models in all datasets with a fraction of the computational cost, demonstrating how simple models need to be considered to understand the benefits of recent models. DGHL outperforms other pure reconstruction-based models because inferring latent vectors for computing anomaly scores provides several advantages. First, it provides additional flexibility and generalization capabilities to prevent false positives, which is instrumental in noisy or non-constant temporal dynamics datasets. Second, it helps to reduce the lasting impact of anomalies on the reconstruction error over time, reducing false positives once anomalies end.

DGHL took an average of 2 minutes to train for each entity (e.g. one machine of SMD or one chanel of SMAP) consistently across datasets. For instance, the training time was around 60 minutes for SMD and MSL, and 100 minutes for SMAP. This is comparable to other state-of-the-art models self-reported training times such as NCAD and faster than RNN based models. For instance, OmniAnomaly took an average of 20 minutes to train each model for each machine on the SMD dataset. The inference time varies depending on the length of the test set. The average time to infer 3000 timestamps (average downsampled SMD test set), with $s_w = 32$, was lower than 5 seconds.

[0]MTAD-GAT authors do not provide public implementation of the model nor evaluation on SWaT. NCAD results on SWaT is not comparable since they use segment adjustment.

4.3 Online Anomaly Detection with missing data

All current benchmark datasets in the time-series anomaly detection literature assume *perfect* data. However, this is not usually the case in real scenarios, with issues like missing values, corrupted data, and variable features. This section presents the first experiments to assess the robustness of current state-of-the-art models to common data issues such as missing values. In particular, we adapt the popular occlusion experiments from computer vision literature for training models with incomplete data.

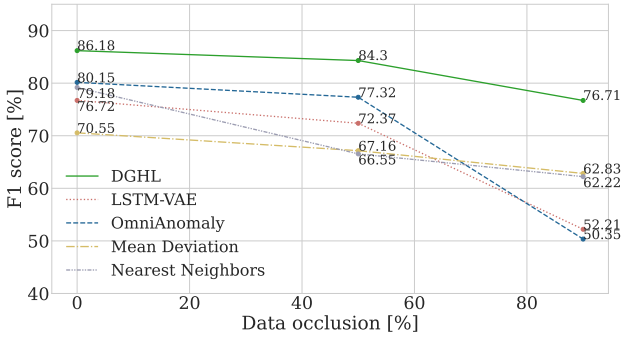

Figure 5: F_1 scores for DGHL and LSTM-VAE benchmark for occlusion experiments on SMD for three levels of occlusion probability p, 0, 0.5, 0.9 and $r = 5$.

We define the occlusion experiments with two parameters. First, the original time-series $\boldsymbol{Y} \in \mathbb{R}^{m \times T}$, is divided in r segments of equal length, $\boldsymbol{Y_i} \in \mathbb{R}^{m \times \frac{T}{r}}$. Second, each feature m in each segment is occluded for model training or inference with probability p.

We assess the robustness of models to incomplete training data with occluding experiments on the SMD dataset, for different levels of r and p, and using F_1 scores to evaluate performance. Figure 5 shows the F_1 score for DGHL , LSTM-VAE, and OmniAnomaly. DGHL achieves the highest scores consistently, with an increasing relative performance on higher data occlusion probability. Moreover, DGHL maintained high F_1 scores even with up to 90% of missing information,

Table 1: F_1 scores on benchmark datasets (the larger the better). The benchmark models performance was taken from (Shen et al., 2020), (Carmona et al., 2021) and (Zhao et al., 2020). First place is marked in bold and second place in bold and italic. DGHL corresponds to the full model described in previous section, without Hierarchical factors corresponds to the simpler model with fully independent latent vectors for each window.

Model	**SMAP**	**MSL**	**SWaT**
Mean deviation (one-line)	57.61	68.91	85.71
Nearest Neighbors	75.10	90.01	86.72
AnoGAN	74.59	86.39	86.64
DeepSVDD	71.71	88.12	82.82
DAGMM	82.04	86.08	85.37
LSTM-VAE	75.73	73.79	86.39
MAD-GAN	81.31	87.47	86.89
MSCRED	85.97	77.45	86.84
OmniAnomaly	85.35	90.14	86.67
MTAD-GAT	90.13	90.84	-
THOC	***95.18***	93.67	**88.09**
NCAD	94.45	**95.60**	-
DGHL	**96.38 ± 0.72**	***94.08 ± 0.35***	***87.47 ± 0.22***
DGHL, without Hierarchial factors	94.87 ± 0.71	91.26 ± 0.71	87.08 ± 0.12
DGHL, with encoder, no Hier. factors	78.41 ± 0.92	87.10 ± 0.54	86.39 ± 0.32

without any changes to the hyperparameters, architecture or training procedure.

4.4 Time-series generation

DGHL is trained to generate time-series windows from a latent space representation. We examine how DGHL learns the representation by interpolating and extrapolating between two latent vectors, $\boldsymbol{Z}_l$ and $\boldsymbol{Z}_u$, inferred from two windows of a real time-series from the SMD. The trained *Generator network* is then used to generate new windows across the interpolation subspace.

Figure 6 presents the generated windows for interpolated and extrapolated vectors for a subset of the original features. The generated time-series smoothly transition between clear patterns on both shape and scale. Moreover, DGHL is able to generate meaningful time-series on the extrapolation region. This experiment shows how our approach maps similar time-series windows into close points of the latent space, which is a desirable property of latent representations.

5 DISCUSSION

DGHL outperforms SoTA baselines and simple approaches by the current experimental standards of the literature. Although DGHL relies on MCMC for posterior sampling, it remains computationally efficient thanks to a lower number of training iterations needed. The ablations studies presented in Table 1 demonstrates the complementary gains of our two main contributions, namely, a novel hierarchical latent representation and training the *Generator* with the alternating back-propagation algorithm.

As mentioned in (Wu and Keogh, 2020), some of the benchmark datasets used in our experiments have mislabeled observations. While we agree this adds noise to the evaluation metrics, we believe the consistent improvement of our model (and current SoTA deep learning models) demonstrates their superior performance on this task over simpler approaches. We also observed the SMD dataset does not have a considerable amount of mislabel observations that could significantly alter the results. We decided to use these benchmark datasets for comparison purposes with existing methods.

Most of the anomalous segments in the benchmark datasets used in this paper can be easily identified by humans (e.g. large spikes), as noted in (Wu and Keogh, 2020). Accurately detecting such anomalies is relevant for many applications, in particular large-scale automatized systems, for which the current benchmarks provide representative estimates of the relative performance of models. Identifying contextual anomalies, which can be hard to detect even for humans, is also highly relevant but current benchmarks do not provide insights on the performance of models on such tasks. Creating relevant benchmark datasets with contextual anomalies is a pressing necessity.

As shown in equation 1, our approach assumes a con-

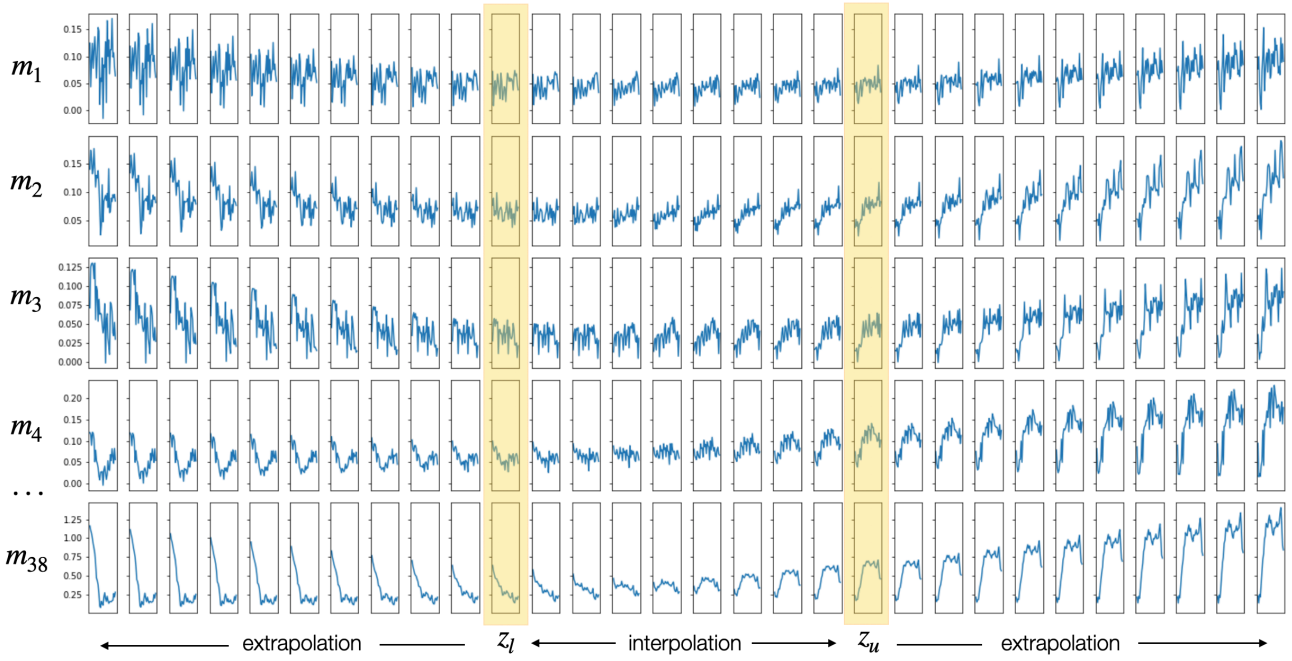

Figure 6: Generated time-series windows from interpolation and extrapolation of latent vectors using DGHL trained on machine-1-1 of the SMD.

stant variance. The normalization by the accumulated standard deviation of scores help to account for the variance of each future, allowing to have single thresholds across different entities. It would be interesting to extend DGHL to also model the variance of each feature. We show in the supplementary material how DGHL can be used for forecasting. An interesting research question is to evaluate how this approach performs on the forecasting task. The hierarchical latent factors allows the model to reconstruct long time-series, by learning long-term temporal relations, making our approach useful on multivariate long-horizon forecasting (Zhao et al., 2020; Challu et al., 2022). Moreover, as demonstrated with both NASA datasets, DGHL can incorporate exogenous variables, which are crucial in several applications such as electricity price forecasting (Olivares et al., 2021).

We do not believe the paper's contribution can be directly misused to have a negative societal impact. We recommend performing additional thorough experiments on healthcare before using the proposed approach in applications in this domain. DGHL required significantly less computational resources and training time than most current state-of-the-art models, but it still uses high-performance hardware and more resources than simpler models. These additional costs should be considered on applications compared to the improved performance benefits.

6 CONCLUSION

In this paper, we introduced DGHL, a state-of-the-art Deep Generative model for multivariate time-series anomaly detection. The proposed model maps time-series windows to a novel hierarchical latent space representation, which leverages the time-series dynamics to encode information more efficiently. A ConvNet is used as the *Generator*. DGHL does not rely on auxiliary networks, such as encoders or discriminators; instead, it is trained by maximizing the likelihood directly with the Alternating Back-Propagation algorithm. Our model has several advantages over existing methods: i. shorter training times, ii. demonstrated superior performance on several benchmark datasets, and iii. better robustness to missing values and variable features.

Aknowledgments

This work was completed during an internship at Amazon. We thank Kin Olivares and François-Xavier Aubet for insightful conversations.

References

Chris U Carmona, François-Xavier Aubet, Valentin Flunkert, and Jan Gasthaus. Neural contextual anomaly detection for time series. *arXiv preprint arXiv:2107.07702*, 2021.

Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*, 2019.

Cristian Challu, Kin G Olivares, Boris N Oreshkin, Federico Garza, Max Mergenthaler, and Artur Dubrawski. N-hits: Neural hierarchical interpolation for time series forecasting. *arXiv preprint arXiv:2201.12886*, 2022.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4027–4035, 2021.

Tian Han, Yang Lu, Song-Chun Zhu, and Ying Nian Wu. Alternating back-propagation for generator network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 387–395, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Dan Li, Dacheng Chen, Jonathan Goh, and See-kiong Ng. Anomaly detection with generative adversarial networks for multivariate time series. *arXiv preprint arXiv:1809.04758*, 2018.

Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. In *International Conference on Artificial Neural Networks*, pages 703–716. Springer, 2019.

Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2 (11):2, 2011.

E Nijkamp, B Pang, T Han, L Zhou, SC Zhu, and YN Wu. Learning multi-layer latent variable model with short run mcmc inference dynamics. In *European Conference on Computer Vision*, 2021.

Kin G Olivares, Cristian Challu, Grzegorz Marcjasz, Rafał Weron, and Artur Dubrawski. Neural basis expansion analysis with exogenous variables: Forecasting electricity prices with nbeatsx. *arXiv preprint arXiv:2104.05522*, 2021.

Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. Learning latent space energy-based prior model. *arXiv preprint arXiv:2006.08205*, 2020.

Bo Pang, Erik Nijkamp, Tian Han, and Ying Nian Wu. Generative text modeling through short run inference. *arXiv preprint arXiv:2106.02513*, 2021.

Daehyung Park, Yuuna Hoshi, and Charles C Kemp. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *IEEE Robotics and Automation Letters*, 3(3):1544–1551, 2018.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402. PMLR, 2018.

Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 2021.

Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging*, pages 146–157. Springer, 2017.

Lifeng Shen, Zhuocong Li, and James Kwok. Time-series anomaly detection using temporal hierarchical one-class network. *Advances in Neural Information Processing Systems*, 33:13016–13026, 2020.

Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2828–2837, 2019.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011.

Renjie Wu and Eamonn J Keogh. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *arXiv preprint arXiv:2009.13807*, 2020.

Jianwen Xie, Ruiqi Gao, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. Learning dynamic generator model by alternating back-propagation through time. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5498–5507, 2019.

Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 World Wide Web Conference*, pages 187–196, 2018.

Hang Zhao, Yujing Wang, Juanyong Duan, Congrui Huang, Defu Cao, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Multivariate time-series anomaly detection via graph attention network. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 841–850. IEEE, 2020.

Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*, 2018.

Supplementary Material: Deep Generative model with Hierarchical Latent Factors for Time Series Anomaly Detection

A Hyperparameters

The hyperparameters of our method are divided in the following groups:

- *Generator* architecture: number of convolutional filters, window size, hierarchical structure and step size.
- MCMC: latent space dimension and Langevin Dynamics update parameters.
- Optimization: learning rate, decay, batch size and number of iterations.

The hyperparameters where selected based on reconstruction error from a validation set of the SMD (constructed from the training set to maintain comparability). For the Langevin Dynamics hyperparameters we used the values proposed in previous work, which are known to work across many different datasets and applications.

We use the same hyperparameters for the four benchmark datasetes, SMAP, MSL, SWAT and SMD. This demonstrates the versatility of our method and how it can potentially be used in real datasets with minimal hyperparameter tuning, further reducing its computational cost. The following table shows the final hyperparameters used in the experiments.

Table 2: Hyperparameters

Hyperparameter	Value
Window size (s_w/a_L).	64
Hierarchical structure ($\mathbf{a}$).	$[1, 4]$
Step size (s).	256
ConvNet filters multiplier.	32
Max filters per layer.	256
Latent vector dimension ($[d_1, d_2]$).	$[20, 5]$
Langevin Dynamics steps during training.	25
Langevin Dynamics steps during inference.	500
Langevin Dynamics step size (s_z).	0.001
Langevin Dynamics sigma (σ_z).	0.025
Initial learning rate.	$1e^{-3}$
Training steps.	1000
Batch Size.	4
Learning rate decay (3 times).	0.8

B Time-series forecasting

DGHL can be used as a forecasting model without any changes to the training procedure or architecture. DGHL can forecast future values by masking them during the inference of latent vectors, analogous to how the model handles missing data. The observed timestamps (left to the forecasting starting timestamp as represented with the vertical line of Figure 7) are used to infer the current latent vector, which is used to generate the whole window, producing the forecasts.

Figure 7 shows an example of the forecasts produced for a subset of machine-1-1 of the SMD dataset. In this example, the model is trained to reconstruct and generate windows of size 128. The first 64 timestamps are available during the inference of the latent vector, and the last 64 corresponds to the forecasted region.

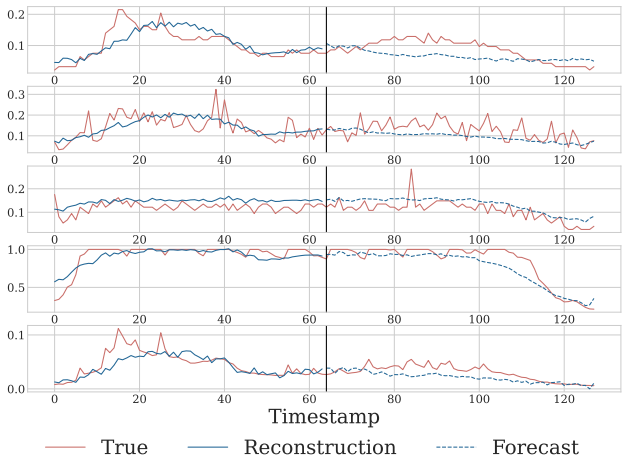

Figure 7: Example of forecasts produced by DGHL on a window of machine-1-1 test set of SMD. The model is trained to generate windows of size 128, the first 64 timestamps are available during inference of latent vectors, and the last 64 correspond to the forecasts.