PrimeNet: Pre-training for Irregular Multivariate Time Series

Ranak Roy Chowdhury[1], **Jiacheng Li**[1], **Xiyuan Zhang**[1],
Dezhi Hong[2], **Rajesh K. Gupta**[1], **Jingbo Shang**[1]

[1] University of California, San Diego
[2] Amazon
{rrchowdh, j9li, gupta, jshang}@eng.ucsd.edu, {xiyuanzh}@ucsd.edu, hondezhi@amazon.com

Abstract

Real-world applications often involve *irregular* time series, for which the time intervals between successive observations are non-uniform. Irregularity across multiple features in a multi-variate time series further results in a different subset of features at any given time (i.e., *asynchronicity*). Existing pre-training schemes for time-series, however, often assume regularity of time series and make no special treatment of irregularity. We argue that such irregularity offers insight about domain property of the data—for example, frequency of hospital visits may signal patient health condition—that can guide representation learning. In this work, we propose PrimeNet to learn a self-supervised representation for irregular multivariate time-series. Specifically, we design a time-sensitive contrastive learning and data reconstruction task to pre-train a model. Irregular time-series exhibits considerable variations in sampling density over time. Hence, our triplet generation strategy follows the density of the original data points, preserving its native irregularity. Moreover, the sampling density variation over time makes data reconstruction difficult for different regions. Therefore, we design a data masking technique that always masks a constant time duration to accommodate reconstruction for regions of different sampling density. We learn with these tasks using unlabeled data to build a pre-trained model and fine-tune on a downstream task with limited labeled data, in contrast with existing fully supervised approach for irregular time-series, requiring large amounts of labeled data. Experiment results show that PrimeNet significantly outperforms state-of-the-art methods on naturally irregular and asynchronous data from Healthcare and IoT applications for several downstream tasks, including classification, interpolation, and regression.

Introduction

Many real-world applications generate data with non-uniform time-interval between successive observations. For example, sensors are triggered at irregular intervals driven by events in real-life. Further, not all sensors are triggered at the same time. Thus, irregularity (in time) and asynchronicity (across sensors) are natural characteristics of many time series that provide rich insights into real-world events.

Naturally occurring irregularity in many datasets reflect intrinsic domain property about the underlying system, which can be leveraged to learn the task better. For example, to predict whether a patient is sick or healthy based on their schedule of doctor visits and medical test results, the frequency of visits may be a useful signal, in addition to the physiological variables, i.e. sick patients visit doctors more frequently than healthy ones. However, no regular time series models, whether fully- (Chowdhury et al. 2022), semi- (Zerveas et al. 2021) or un- (Tonekaboni, Eytan, and Goldenberg 2021) supervised, encode time information. They assume constant time intervals between all consecutive observations in a sequence (*regularity*) with all features observed at any given time (*synchronicity*) (Marlin et al. 2012). Hence, simply adapting pre-trained regular time-series methods for irregular time-series is sub-optimal.

Recently, ODE- (Rubanova, Chen, and Duvenaud 2019), attention- (Shukla and Marlin 2021), and set- (Horn et al. 2020) based models that directly learn time information and encode irregularity have outperformed regular time series models on irregular data. However, most of these irregular methods are fully- (Rubanova, Chen, and Duvenaud 2019) or semi- (Shukla and Marlin 2021) supervised, requiring large amounts of labeled data. Data labeling in IoT is time-consuming and both cost and labor-intensive as it involves physical sensor deployment and human annotators with domain expertise. Moreover, accessing labeled data in healthcare may raise security and privacy concerns.

To this end, we propose PrimeNet, the first pre-trained model for irregular multivariate time series. Specifically, we design two time-sensitive tasks based on contrastive learning and data reconstruction to build a self-supervised representation from *completely unlabeled* irregular time series data.

Time-slicing (Franceschi, Dieuleveut, and Jaggi 2019), which chunks a time series into slices containing equal number of readings, is commonly used to augment regular time-series data for contrastive learning. However, it cannot generate representative sub-sequences for an irregular time series as irregular time series exhibit significant variations in sampling density over time and the sampling density of a given time slice may not mirror that of the entire time series (Figure 3). A common approach would be to randomly sample observations from an irregular time series to construct a sub-sequence. However, an irregular time series may also exhibit significant imbalanced occurrences of dense and sparse observations. Hence, random sampling will lead to

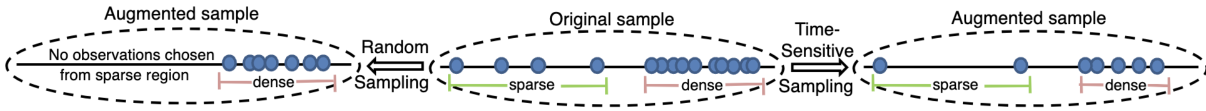

Figure 1: The solid black line represents the time axis and an blue circle represents an observation in time. Random sampling from a time series may lead to no observations being sampled from sparse regions in the augmented sample. Time-sensitive stratified sampling draws observations from both dense and sparse regions, better reflecting the sampling density variation of the original sample.

(a) Constant length masking for regular time-series.

(b) Constant length masking for irregular time-series.

(c) Constant time duration masking for irregular time-series.

Figure 2: Grey box shows the time duration of successive observations we mask under each masking technique.

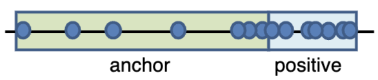

Figure 3: For an irregular time-series, time-slicing the sequence generates an anchor and a positive, neither of which reflects the sampling density variation of the original data.

high sampling bias, drawing observations from some regions but not others, resulting in an unrepresentative sub-sequence (Figure 1). Instead, we propose a time-sensitive stratified sampling technique that draws observations from dense and sparse regions of the time series and combine them to construct a sub-sequence. The generated sub-sequences better reflect the sampling density variations and are, therefore, more representative of the original data.

Masking and reconstructing observations learn a good representation for regular time-series (Zerveas et al. 2021). For a regular time series, masking the same number of successive observations for every segment (fixed *length*) will always mask over the same time duration of data because the sampling density is constant (Figure 2(a)). However, for an irregular time series with inconsistent sampling density, a fixed length mask segment will mask over a different duration for different segments depending on their respective sampling densities (Figure 2(b)). Hence, the difficulty to reconstruct data varies through an irregular time series, resulting in poor representations. Therefore, we propose a fixed *time* masking technique that always masks over the same duration of data instead (Figure 2(c)). This effectively adjusts the number of masked observations for each segment depending on its sampling density, thereby stabilizing the reconstruction task across regions of different sampling densities, and improving the learned representation.

We conduct experiments on naturally occurring irregular and asynchronous time-series datasets from Healthcare (Silva et al. 2012; Johnson et al. 2016), Activity (Kaluža et al. 2010), and Energy (Tan et al. 2021). We conduct analysis on very few-shot settings and also on full training data settings for several downstream tasks, including classification, regression, and interpolation. We compare PrimeNet to several 1) self-supervised methods for regular time series to show how modeling irregularity helps and 2) fully supervised methods for irregular time series to show the performance boost from using unlabeled data. Experiment results show that PrimeNet significantly outperforms all baselines on all datasets for all downstream tasks, under both few-shot and full training data settings.

To summarize, our main contributions include:

- We present PrimeNet to learn a self-supervised representation from irregular and asynchronous time series.
- We propose time-sensitive contrastive learning and data reconstruction to learn from data irregularity patterns.
- We evaluate PrimeNet on numerous real-world datasets across several downstream tasks to validate and quantify its efficacy compared with benchmark methods.

Reproducibility Code is publicly available at https://github.com/ranakroychowdhury/PrimeNet

Related Work

Self-Supervised Regular Time Series Methods

Recent research on learning unsupervised representations from *regular* and *synchronous* time-series performs well under limited labeled data settings, using triplet loss (Franceschi, Dieuleveut, and Jaggi 2019), hierarchical contrastive loss (Yue et al. 2021; Chowdhury, Adnan, and Gupta 2020), Fourier transform (Li et al. 2021; Zhang et al. 2023), task-aware reconstruction (Chowdhury et al. 2022). They can be adapted to irregular time-series by discretizing continuous-time samples into uniformly-spaced fixed-size bins (Shukla and Marlin 2019; Zhang et al. 2022). However, irregular time series are marked by regions of high and low sampling densities. A wide bin would aggregate data in dense regions, resulting in a loss of fine-grained details. A short bin would generate a high fraction of missing data in sparse regions exploding the sequence length, making imputation increasingly difficult. Consequently, imputation-based methods are hardly used to learn from such irregular time series. Moreover, by treating irregular time series data like a regular one, they abstract the vital irregular time information away from the model, inhibiting performance.

We exploit the irregular time-interval property to design our self-supervsion tasks to pre-train PrimeNet, thereby learning suitable representations from unlabeled *irregular* and *asynchronous* multivariate time-series data.

Irregular Time Series Methods

Gated Recurrent Unit (GRU)-based methods (Che et al. 2018) involve modifying the LSTM forget gate (Pham et al. 2017) or introducing new time gate (Neil, Pfeiffer, and Liu 2016). (Chen et al. 2018) combined a neural network with a latent ordinary differential equation (ODE) model. (Rubanova, Chen, and Duvenaud 2019; Lechner and Hasani 2020) used neural ODEs to model hidden state dynamics. Others use attention mechanism (Zhang 2019; Tan et al. 2020) similar to (Vaswani et al. 2017) by replacing positional encoding with a fixed time encoding. (Shukla and Marlin 2021; Kazemi et al. 2019) learns time representation.

These models work under fully- or semi-supervised setting and require large amounts of labeled data to learn a task. PrimeNet leverages unlabeled data to operate completely under self-supervised setting. Once pre-trained, PrimeNet can learn any downstream task with limited labeled data.

Notation

An individual data case is a N-dimensional, irregularly and asynchronously sampled multivariate time series, $\mathrm{D} = (\mathrm{T}, \mathrm{X}, \mathrm{M})$, where $\mathrm{T} \in \mathbb{R}^S$, $\mathrm{X} \in \mathbb{R}^{S\times N}$, and $\mathrm{M} \in \mathbb{R}^{S\times N}$. T denotes the union of timestamps at which all the N features have been sampled and S is the number of such timestamps. Let t and n represent a particular time and feature, respectively, in D. X constitutes a sequence of S feature vectors. $\mathrm{X}_t \in \mathbb{R}^N$ represents the feature values sampled at time t. This formulation also covers the uni-variate case when $N = 1$. However, since the time-series is asynchronously sampled, not all the N features may be sampled at t. Hence, we use masking variable M to denote the set of observed features. Specifically, $\mathrm{M}_t \in \mathbb{R}^N$ denote the set of observed and unobserved features at t. If feature n is sampled at t, then $\mathrm{M}_{tn} = 1$, otherwise $\mathrm{M}_{tn} = \mathrm{X}_{tn} = 0$. Using M to deal with unobserved dimensions allows us to transform the irregular length time-series into uniform length and parallelize computations, thereby enabling efficient GPU implementation.

Our time-sensitive self-supervision tasks are model agnostic and can be plugged into any irregular time series architectures, like ODE-, attention-, or set-based models, that explicitly encode time information. We follow a similar architecture to mTAND (Shukla and Marlin 2021) because it is the current state-of-the-art for irregular time series. Our model is pre-trained on unlabeled data using time-sensitive contrastive and data reconstruction loss and then fine-tuned on available labeled data for a given downstream task.

Model Architecture

We present PrimeNet's core architecture - Time Embedding, Time-Feature Attention (TFA) and Feature-Feature Attention (FFA), as shown in Figure 4.

Time Embedding (Shukla and Marlin 2021) embeds continuous time points into a vector space by leveraging H

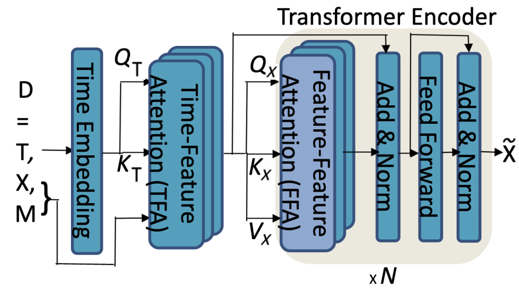

Figure 4: Model Architecture

embedding functions $\phi h(\mathrm{T})$, each outputting a representation of size d_r. Dimension i of embedding h is defined as:

$$\phi h(\mathrm{T})[i] = \begin{cases} \omega_{0h}.\mathrm{T} + \alpha_{0h}, \text{ if } i = 0 \\ sin(\omega_{ih}.\mathrm{T} + \alpha_{ih}), \text{ if } 0 < i < d_r \end{cases} \tag{1}$$

where ω_{ih}'s and α_{ih}'s are learnable parameters. Linear term encodes the non-periodic patterns. Periodic terms captures periodicity with ω_{ih} and α_{ih} as frequency and phase.

Time-Feature Attention, TFA (Shukla and Marlin 2021) captures the interaction between feature values with their corresponding sampling times. The output from the Time Embedding layer forms the query Q_T and key K_T vectors.

$$\begin{aligned} \text{TFA}(\mathrm{Q}_T, \mathrm{K}_T, \mathrm{M}, \mathrm{X}) = (\mathrm{M} \odot \mathrm{A}_T)\mathrm{X}, \\ \mathrm{A}_T = \mathrm{softmax}(\mathrm{Q}_T \mathrm{K}_T / d_r) \end{aligned} \tag{2}$$

We do an element-wise multiplication of M with A_T to nullify the effect of unobserved features in X at a given sampling time in T. We compute a weighted summation of features, where weights are self-attention scores of features' corresponding sampling time embeddings, Q_T and K_T.

Feature-Feature Attention, FFA (Vaswani et al. 2017) models the self-attention within features. The output from TFA becomes the query, key and value vectors for FFA.

$$\mathrm{Q}_X = \mathrm{K}_X = \mathrm{V}_X = \text{TFA}(\mathrm{Q}_T, \mathrm{K}_T, \mathrm{M}, \mathrm{X}) \tag{3}$$

Then, FFA will encode the outputs from TFA as follows:

$$\begin{aligned} \text{FFA}(\mathrm{Q}_X, \mathrm{K}_X, \mathrm{V}_X, \mathrm{M}) = (\mathrm{M} \odot \mathrm{A})\mathrm{V}_X, \\ \mathrm{A} = \mathrm{softmax}(\mathrm{Q}_X \mathrm{K}_X / d_r) \end{aligned} \tag{4}$$

The output of FFA is fed through residual and feed forward layers like in a typical Transformer Encoder. The data is fed through N such Encoders to generate the output $\tilde{\mathrm{X}}$.

Time-Sensitive Pre-training

We present two time-sensitive self-supervised pre-training objectives, namely, Time Contrastive Learning (TimeCL) and Time Reconstruction (TimeReco). We discuss how we augment the data, followed by loss computation details.

Time Contrastive Learning (TimeCL)

Augmenting data for contrastive learning through time-slicing or random sampling fails to capture the irregularity in data, motivating our stratified-sampling based approach.

Algorithm 1: TimeCL Data Augmentation

Input: D
Hyper-parameters: (μ_l, μ_u), (λ_l, λ_u)
Output: D_A, D_P

1: $\mathrm{Z} \leftarrow \{\frac{(\mathrm{T}_i - \mathrm{T}_{i-1}) + (\mathrm{T}_{i+1} - \mathrm{T}_i)}{2} | \forall i \in \mathbb{Z}^+, 1 < i < S\}, \mathrm{Z} \in \mathbb{R}^S$
2: Sort (Z, T) in ascending order of Z
3: $\mathrm{T}_{dense} \leftarrow \mathrm{T}[1 : S/2], \mathrm{T}_{dense} \in \mathbb{R}^{S/2}$
4: $\mathrm{T}_{sparse} \leftarrow \mathrm{T}[S/2 : S], \mathrm{T}_{sparse} \in \mathbb{R}^{S/2}$
5: $\mu \sim \mathrm{U}(\mu_l, \mu_u), 0 < \mu_l < \mu_u < 1$
6: $\lambda \sim \mathrm{U}(\lambda_l, \lambda_u), 0 < \lambda_l < \lambda_u < 1$
7: $\mathrm{T}_A \sim$ Sample $\lambda \mu S$ and $(1 - \lambda)\mu S$ timestamps from T_{dense} and T_{sparse}, respectively, $\mathrm{T}_A \in \mathbb{R}^{\mu S}$
8: $\mathrm{T}_P \leftarrow \mathrm{T} - \mathrm{T}_A, \mathrm{T}_P \in \mathbb{R}^{(1-\mu)S}$
9: $\mathrm{D}_A, \mathrm{D}_P \leftarrow (\mathrm{T}_A, \mathrm{X}_{\mathrm{T}_A}, \mathrm{M}_{\mathrm{T}_A}), (\mathrm{T}_P, \mathrm{X}_{\mathrm{T}_P}, \mathrm{M}_{\mathrm{T}_P})$
10: **return** D_A, D_P

Algorithm 2: TimeReco Data Augmentation

Input: D
Hyper-parameters: J, α
Output: D_U, D_V

1: $\mathrm{X}_U, \mathrm{M}_U \leftarrow \mathrm{X}, \mathrm{M}$
2: $\mathrm{X}_V, \mathrm{M}_V \leftarrow$ initialized with $0's$
3: **for** n in N **do**
4: $\quad \mathrm{T}_n = \mathrm{T}[\mathrm{M}_n], t_n \in \mathbb{R}^{L_n}$
5: $\quad q_n \leftarrow \alpha(\mathrm{T}_n[-1] - \mathrm{T}_n[0])$
6: $\quad$ **for** j in J **do**
7: $\quad\quad t \sim$ Randomly sample a timestamp from T_n
8: $\quad\quad \mathrm{X}_{V_n}[t : t + q_n], \mathrm{M}_{V_n}[t : t + q_n] = \mathrm{X}_{U_n}[t : t + q_n], \mathrm{M}_{U_n}[t : t + q_n]$
9: $\quad\quad \mathrm{X}_{U_n}[t : t + q_n], \mathrm{M}_{U_n}[t : t + q_n] = 0, 0$
10: $\quad$ **end for**
11: **end for**
12: $D_U, D_V \leftarrow (\mathrm{T}, \mathrm{X}_U, \mathrm{M}_U), (\mathrm{T}, \mathrm{X}_V, \mathrm{M}_V)$
13: **return** D_U, D_V

Data Augmentation Contrastive learning (Chen et al. 2020; Hogan, Li, and Shang 2022) augments observations from the same input to form an anchor and positive (similar to anchor), while a sample from other in-batch inputs form the negative (different from anchor). This pulls the latent space of the anchor and positive closer while pushing away the negatives, learning good data representation (Jaiswal et al. 2020; Li, Shang, and McAuley 2022).

To better replicate the irregularity pattern of data, we maintain an approximate sampling density distribution of the data in its augmented sub-sequences, as outlined in Algorithm 1. For every sampling time T_i, we compute its mean time interval, Z_i, by averaging the time lapse between its immediate predecessor, T_{i-1}, and successor, T_{i+1}, to estimate the local density of the sampled features at T_i (Line 1). We reorder Z and T in ascending order of Z and group T into two bins, T_{dense} (lowest 50% Z-values) and T_{sparse} (highest 50% Z-values) (Lines 2 - 4). We sample a proportion of the timestamps from T_{dense} and the remaining from T_{sparse} to form the sampled timestamps of one augmented sub-sequence, T_A (Line 6). The remaining $\mathrm{T} - \mathrm{T}_A$ timestamps form the sampled timestamps of the other sub-sequence, T_P (Line 8). This stratified sampling technique ensures that we draw observations from regions of different sampling density of D, regardless of how scarce the observations from certain regions are. Hence, the augmented sub-sequence can better approximate the irregularity in D. To preserve asynchronicity of D, we extract only the subset of features that was sampled together at a given time in D to form the feature set for that time in the augmented sub-sequence. Thus, $\mathrm{D}_A = (\mathrm{T}_A, \mathrm{X}_{\mathrm{T}_A}, \mathrm{M}_{\mathrm{T}_A})$ and $\mathrm{D}_P = (\mathrm{T}_P, \mathrm{X}_{\mathrm{T}_P}, \mathrm{M}_{\mathrm{T}_P})$ form good quality anchor and positive sub-sequences, respectively, that are representative of D, thereby improving contrastive learning.

Contrastive Loss, $\mathcal{L}_{CL}$ For a given X_A and X_P, the X_N is formed from all other instances in the same mini-batch. We add a special [CLS] symbol in front of every input and pass it through an embedding layer. The final hidden state corresponding to [CLS] is used as the aggregate sequence representation for contrastive learning.

We use Normalized Temperature-scaled Cross Entropy (NT-Xent) Loss (Chen et al. 2020) as our Contrastive Loss function. This has shown improvement over other contrastive losses, like CPC (Oord, Li, and Vinyals 2018) and MoCo (He et al. 2020), in several other domains (Chen et al. 2020). It is a modification of the multi-class B-pair loss, where B is the batch size, with addition of the temperature parameter, τ, to scale the cosine similarities as follows:

$$\mathcal{L}_{CL} = -\log \frac{\exp(\tilde{\mathrm{X}}_i \tilde{\mathrm{X}}_j / \tau)}{\sum_{k=1}^{2B} \exp(\tilde{\mathrm{X}}_i \tilde{\mathrm{X}}_k / \tau)} \tag{5}$$

Time Reconstruction (TimeReco)

Masking constant length of data across time is not suitable to learn reconstruction from non-uniform time-interval data, prompting a constant time masking technique.

Data Augmentation Reconstruction for irregular time-series presents two key challenges. First, some regions are more densely sampled than others. Therefore, masking a constant length of data will mask over a shorter time-interval for a denser region compared to a sparser region, as shown in Figure 2(b). Hence, reconstructing data at the masked segment may be trivial for the dense region with abundant unmasked data in close temporal vicinity to use as contextual information but difficult for a sparse region. Second, for multivariate time-series, each feature may have different sampling frequency and may be sampled over different duration. Hence, each feature has different time gaps between successive observations, i.e. asynchronicity, rendering a constant length masking strategy ineffective.

To learn a better reconstruction for irregular time-series, we propose Algorithm 2. We specify the number of masking segments J, and the fraction of time interval α, to mask for each segment. To address the asynchronicity problem, we compute the timespan q_n to mask separately for each feature n (Line 5) because the total duration for which each feature

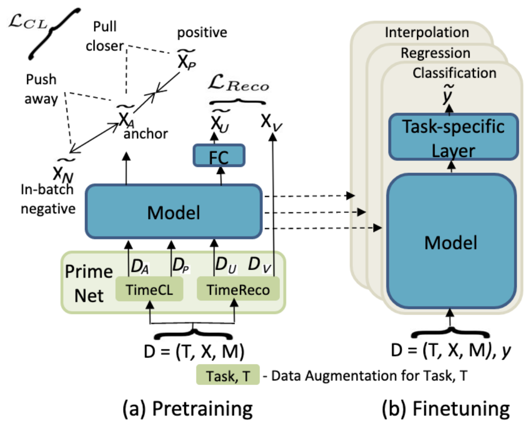

Figure 5: PrimeNet pre-training and fine-tuning. Green modules represent the data augmentation algorithms. Blue modules represent the learnable components.

lasts may vary. Features lasting shorter should have shorter masking segments as compared to those lasting longer. To deal with irregularity within a time-series, we fix the timespan q_n to mask for feature n, instead of fixing the number of observations to mask. This adapts the length of masking segment based on the sampling density of the time-series in the masking region, as shown in Figure 2(c). For a given q_n, a dense region will mask more observations than a sparse region. Hence for denser regions, there will not be sufficient unmasked observations in close temporal proximity of the masked segment to make the reconstruction trivial. Similarly for sparser regions, the number of masked observations will be low, so there will be sufficient observations in the temporal vicinity of the masked region to keep the task tractable. Lines 6 - 10 outlines this procedure.

Reconstruction Loss, $\mathcal{L}_{Reco}$ We feed the masked out features X_U to PrimeNet and extract the generated features $\tilde{\mathrm{X}}_U$. The Reconstruction Error between model output $\tilde{\mathrm{X}}_U$ and target X_V, is computed using Mean Squared Error (MSE),

$$\mathcal{L}_{Reco} = \left\| \mathrm{M}_V \odot (\tilde{\mathrm{X}}_U - \mathrm{X}_V) \right\|_2^2 \tag{6}$$

Hence the total loss $\mathcal{L}$ becomes:

$$\mathcal{L} = \eta \mathcal{L}_{CL} + (1-\eta)\mathcal{L}_{Reco}, \tag{7}$$

where η is a hyperparameter, $0 < \eta < 1$. It balances the two losses because different datasets may benefit differently from these two tasks.

Figure 5(a) shows the pre-training workflow of PrimeNet. For a given data point D, we feed it through Algorithm 1 to generate the anchor D_A, and positive D_P. Additionally, we feed D through Algorithm 2 to mask its features and prepare the masked input D_U and target D_V.

For finetuning on supervised downstream tasks, like classification, regression and interpolation, we append task-specific layers on top of pre-trained PrimeNet, as shown in Figure 5(b). The task-specific layers typically consists of some fully connected layers with non-linear activation.

Experiments

We evaluate PrimeNet on some real-world irregular and asynchronous time-series data from Healthcare and IoT domain for classification, regression, and interpolation tasks.

Datasets

- **PhysioNet Challenge 2012** (Silva et al. 2012) and **MIMIC-III** (Johnson et al. 2016) are multivariate time series datasets consisting of 37 and 12 physiological variables, respectively, extracted from intensive care unit (ICU) records. Each record contains 48 hours of measurements after admission to ICU. We predict in-hospital mortality (binary classification) from this data.
- **Activity** (Kaluža et al. 2010) dataset has 3-D positions of the waist, chest and ankles from 5 individuals performing activities including walking, sitting, lying, standing, etc.
- **Appliances Energy** (Tan et al. 2021) dataset contains 138 time series with 24 dimensions, including temperature, humidity, pressure, wind speed, visibility, and dew point. The data is averaged for 10 minutes and spans 4.5 months.

PhysioNet, MIMIC-III, and Activity are naturally irregular, i.e., data was sampled at irregular times during collection. Appliances Energy is a regularly sampled dataset where we synthetically induce irregularity by dropping out random data. To better understand their irregularity pattern, we provide some summary statistics: (mean, standard deviation) of the missing ratio of each feature's time series across the dataset. If a dataset was sampled for 100 timestamps, then a 0.75 mean missing ratio means that on average, each feature was present for 25 and was missing for the remaining 75 timestamps across this dataset. Missing ratio statistics: PhysioNet $(0.86, 0.24)$, MIMIC-III $(0.65, 0.36)$, Activity $(0.75, 0.64)$, Appliances Energy $(0.87, 0.47)$.

Baselines

Self-Supervised Regular Time-Series Methods

1. **TS2Vec** (Yue et al. 2021) performs hierarchical contrastive learning over augmented context views.
2. **TNC** (Tonekaboni, Eytan, and Goldenberg 2021) defines temporal neighborhoods from local smoothness of data.
3. **TST** (Zerveas et al. 2021) pre-trains Transformer by masking fixed length segments and reconstructing them.

Irregular Time-Series Methods Recurrence-, ODE-, and attention-based fully- and semi-supervised methods.

1. **GRU-Mean** (Che et al. 2018) combines hidden state decay with input decay.
2. **P-LSTM** (Neil, Pfeiffer, and Liu 2016) adds a learnable oscillator to modulate LSTM to create dependencies on elapsed-time, and uses vanishing factor in gradients.
3. **RNN-VAE** VAE model with RNN encoder and decoder.
4. **ODE-RNN** (Rubanova, Chen, and Duvenaud 2019) uses neural ODEs to model hidden state dynamics and RNN to update hidden states with new observations.
5. **L-ODE** (Rubanova, Chen, and Duvenaud 2019) Latent ODE with ODE-RNN encoder and neural ODE decoder.
6. **mTAND** (Shukla and Marlin 2021) Multi-time attention module followed by a VAE-based encoder-decoder.

Baselines	PhysioNet (Silva et al. 2012)				MIMIC-III (Johnson et al. 2016)			
	1-shot	2-shot	4-shot	Full	1-shot	2-shot	4-shot	Full
Self-supervised regular time-series methods								
TNC	$\underline{0.615 \pm 0.014}$	$\underline{0.641 \pm 0.023}$	$\underline{0.632 \pm 0.011}$	0.755	0.558 ± 0.036	0.568 ± 0.053	0.555 ± 0.031	0.749
TS2Vec	0.554 ± 0.04	0.556 ± 0.046	0.584 ± 0.023	0.737	0.559 ± 0.039	0.560 ± 0.028	0.565 ± 0.029	0.824
TST	0.535 ± 0.021	0.564 ± 0.030	0.552 ± 0.042	0.815	0.542 ± 0.081	$\underline{0.596 \pm 0.079}$	0.591 ± 0.053	0.822
Irregular time-series methods								
mTAND	0.612 ± 0.016	0.587 ± 0.009	0.598 ± 0.009	$\underline{0.837}$	0.534 ± 0.003	0.538 ± 0.012	0.528 ± 0.003	$\underline{0.829}$
GRU-Mean	0.542 ± 0.054	0.523 ± 0.042	0.575 ± 0.038	0.806	$\underline{0.573 \pm 0.010}$	0.568 ± 0.005	0.576 ± 0.007	0.786
P-LSTM	0.579 ± 0.058	0.551 ± 0.048	0.569 ± 0.049	0.782	0.546 ± 0.086	0.573 ± 0.033	$\underline{0.595 \pm 0.083}$	0.745
RNN-VAE	0.444 ± 0.008	0.530 ± 0.043	0.459 ± 0.042	0.542	0.516 ± 0.001	0.516 ± 0.002	0.516 ± 0.003	0.512
ODE-RNN	0.515 ± 0.095	0.573 ± 0.097	0.495 ± 0.119	0.694	0.552 ± 0.005	0.562 ± 0.009	0.564 ± 0.009	0.709
L-ODE	0.592 ± 0.048	0.597 ± 0.042	0.598 ± 0.036	0.701	0.481 ± 0.005	0.485 ± 0.004	0.484 ± 0.003	0.590
PrimeNet	$\mathbf{0.641 \pm 0.071}$	$\mathbf{0.663 \pm 0.047}$	$\mathbf{0.681 \pm 0.026}$	$\mathbf{0.842}$	$\mathbf{0.595 \pm 0.063}$	$\mathbf{0.601 \pm 0.06}$	$\mathbf{0.638 \pm 0.038}$	$\mathbf{0.838}$

Table 1: Classification on PhysioNet and MIMIC-III measured by Area Under the ROC curve (AUC). Higher AUC is better.

Baselines	Activity at 10% (Kaluža et al. 2010) ($\times 10^{-2}$)				PhysioNet at 50% (Silva et al. 2012) ($\times 10^{-2}$)			
	1-shot	2-shot	4-shot	Full	1-shot	2-shot	4-shot	Full
Self-supervised regular time-series methods								
TNC	63.33 ± 1.91	60.48 ± 3.75	47.73 ± 2.27	20.55	14.28 ± 3.07	13.92 ± 2.98	11.63 ± 2.72	7.18
TS2Vec	69.83 ± 6.79	60.34 ± 2.15	45.97 ± 3.32	23.98	15.37 ± 3.71	$\underline{12.96 \pm 3.24}$	11.53 ± 2.79	7.69
TST	62.99 ± 1.57	52.80 ± 3.25	50.11 ± 2.03	24.81	$\underline{14.03 \pm 3.11}$	13.58 ± 2.91	10.87 ± 2.58	6.92
Irregular time-series methods								
mTAND	$\underline{62.17 \pm 8.91}$	$\underline{51.08 \pm 4.79}$	$\underline{40.35 \pm 3.91}$	$\underline{20.46}$	15.21 ± 3.16	13.73 ± 3.02	$\underline{10.83 \pm 2.74}$	$\underline{6.89}$
GRU-Mean	75.44 ± 1.13	74.41 ± 0.86	72.87 ± 0.59	30.49	18.34 ± 3.72	17.24 ± 3.61	14.82 ± 3.21	8.83
P-LSTM	186.81 ± 3.45	186.61 ± 3.03	183.61 ± 2.10	31.94	42.18 ± 7.84	40.10 ± 6.81	36.53 ± 6.28	14.27
RNN-VAE	194.18 ± 0.25	194.20 ± 0.46	194.05 ± 0.31	61.41	47.91 ± 8.26	41.92 ± 7.72	38.35 ± 6.89	19.37
ODE-RNN	112.84 ± 7.85	107.79 ± 4.68	103.55 ± 5.34	26.69	22.71 ± 4.82	19.26 ± 4.13	18.91 ± 3.67	11.91
L-ODE	97.67 ± 5.92	92.52 ± 2.53	85.96 ± 2.85	22.51	18.63 ± 3.79	16.32 ± 3.21	15.52 ± 2.62	9.26
PrimeNet	$\mathbf{60.84 \pm 21.62}$	$\mathbf{41.56 \pm 16.98}$	$\mathbf{25.45 \pm 3.28}$	$\mathbf{14.3}$	$\mathbf{11.28 \pm 2.91}$	$\mathbf{9.37 \pm 2.72}$	$\mathbf{7.85 \pm 1.98}$	$\mathbf{3.59}$

Table 2: Interpolation on Activity and PhysioNet datasets measured by RMSE. Lower RMSE is better.

Experimental Protocols

We infer a continuous missing segment of 10% and 50% values for interpolation, while conditioning on the remaining 90% and 50% of the observed points for Activity and PhysioNet, respectively. For interpolation, we use the entire output representation from PrimeNet, while for classification and regression we use the final hidden state of `[CLS]` symbol as the aggregate sequence representation. We compute Cross-Entropy Loss for classification and Root Mean Squared Error (RMSE) for regression and interpolation. Due to class imbalance in Physionet and MIMIC-III, we assess classification using Area Under the ROC curve (AUC score). We assess interpolation and regression using RMSE.

During pretraining, we measure contrastive learning classification (i.e. how many samples are predicted correctly among the $2B$ sub-samples) and use the validation accuracy for early stopping. During finetuning, we update the parameters of both the task-specific layers and PrimeNet

We conduct grid search on hyper-parameters, $\eta = (0.3, 0.4, 0.5, 0.6, 0.7)$, $\alpha = (0.15, 0.05, 0.03)$, $J = (1, 3, 5)$, $\mu_l, \lambda_l = (0.3, 0.4)$ and $\mu_u, \lambda_u = (0.7, 0.6)$ to report test results based on the best held-out validation performance. Best values for $\eta = 0.5, 0.6, 0.5, 0.5$ for PhysioNet, MIMIC-III, Activity, Appliances Energy, respectively.

Results

Table 1, 2, and 3 show the results. k-shot refers to k labeled training examples. For each few-shot setup, we repeat an experiment five times using a different training sample set each time, to report the mean and standard deviation of metrics. We mark the **best** and second best values.

Classification Table 1 shows AUC scores on mortality prediction task (binary classification) of PhysioNet and MIMIC-III datasets. For the fully supervised irregular time-

Baselines	1-shot	2-shot	4-shot	Full
Self-supervised regular time-series methods				
TNC	12.51 ± 0.56	11.68 ± 0.51	9.82 ± 0.73	3.78
TS2Vec	3.95 ± 0.09	$\underline{3.94 \pm 0.10}$	3.73 ± 0.30	3.40
TST	3.93 ± 0.44	4.11 ± 0.62	3.96 ± 0.56	3.43
Irregular time-series methods				
mTAND	4.25 ± 1.10	4.13 ± 0.81	$\underline{3.52 \pm 0.12}$	$\underline{3.39}$
GRU-Mean	$\underline{3.91 \pm 0.46}$	3.97 ± 0.65	3.56 ± 0.15	3.44
P-LSTM	8.53 ± 0.43	7.74 ± 0.43	7.07 ± 0.30	6.58
RNN-VAE	12.51 ± 0.80	12.31 ± 0.65	11.65 ± 0.57	6.79
ODE-RNN	12.32 ± 0.90	11.29 ± 0.55	9.18 ± 0.74	6.74
L-ODE	6.82 ± 1.05	5.42 ± 1.42	4.33 ± 0.52	3.51
PrimeNet	$\mathbf{3.66 \pm 0.54}$	$\mathbf{3.64 \pm 0.43}$	$\mathbf{3.44 \pm 0.03}$	$\mathbf{3.21}$

Table 3: Regression on Appliances Energy (Tan et al. 2021) dataset measured by RMSE. Lower RMSE is better.

Ablations	1-shot	2-shot	4-shot	Full
w/o pre	0.610 ± 0.02	0.596 ± 0.01	0.613 ± 0.01	0.829
(1)	0.624 ± 0.03	0.645 ± 0.05	0.651 ± 0.04	0.811
(2)	0.618 ± 0.06	0.638 ± 0.05	0.647 ± 0.04	0.817
(3) + (2)	0.620 ± 0.06	0.631 ± 0.07	$\underline{0.658 \pm 0.01}$	$\underline{0.834}$
(4) + (1)	$\underline{0.635 \pm 0.08}$	$\underline{0.649 \pm 0.07}$	0.655 ± 0.06	0.829
PrimeNet	$\mathbf{0.641 \pm 0.07}$	$\mathbf{0.663 \pm 0.05}$	$\mathbf{0.681 \pm 0.03}$	$\mathbf{0.842}$

Table 4: Classification performance of PrimeNet's ablation study on PhysioNet. (1) time-sensitive contrastive learning; (2) constant time data masking for reconstruction; (3) random-sampling-based contrastive learning; (4) constant length data masking for reconstruction. PrimeNet = (1)+(2)

series methods that can not leverage unlabeled data, the AUC score hovers around 0.5, suggesting that these methods do not learn any useful information from such limited labeled data, performing slightly better than a random classifier. Moreover, their performance do not improve by much when we increase the amount of labeled data.

For PhysioNet, the self-supervised regular time-series method TNC (Tonekaboni, Eytan, and Goldenberg 2021) performs better than irregular methods under few-shot. This may occur because the irregularity information may not be much relevant to the predictive task for PhysioNet and learning just the feature space is adequate to give good performance. Hence, methods like TNC gets significant performance gain by leveraging additional unlabeled data, despite not utilizing any time interval information. However, for MIMIC-III, the irregular time-series methods are better. For MIMIC-III, it may be crucial to learn from the irregular time gap information to do well on the predictive task. Therefore, despite using only a handful of labeled examples, irregular time-series methods can outperform the self-supervised regular time-series ones that leveraged a lot of unlabeled data.

Interpolation & Regression Table 2 shows the RMSE values of the interpolation task on Activity (Kaluža et al. 2010) and PhysioNet (Silva et al. 2012) datasets, while Table 3 shows the same for regression on Appliances Energy (Tan et al. 2020) dataset. The extremely poor performance of recurrence methods, like P-LSTM, RNN-VAE, and ODE-RNN on both tasks may be attributed to their recurrent nature that accumulates error through time, resulting in poor long-horizon imputations for time series that are sparsely observed. For both tasks, the performance of most methods improves as we increase the number of shots.

Across classification, interpolation, and regression, PrimeNet outperforms self-supervised regular time-series methods under all settings, owing to PrimeNet's ability to explicitly model for irregularity. Moreover, PrimeNet outperforms the fully and semi-supervised irregular time-series models under few-shot settings by transferring the knowledge learned from unlabeled data during pre-training to fine-tuning. Moreover, PrimeNet also outperforms the irregular time-series models under full-training data setting, when all models are trained with the entire labeled training set. This shows that the pretrain-finetune setup not only improves performance under few-shot scenarios but also when there is sufficient labeled data to train the downstream task. PrimeNet s robust to varying extents of irregularity since it excelled on datasets with widely different missing ratios.

Ablations

Table 4 shows the results of PrimeNet's ablation on PhysioNet classification. If we do not pre-train PrimeNet (first row) and directly fine-tune it on labeled data, its performance is similar to mTAND (Shukla and Marlin 2021) (0.829 vs 0.837) as both use similar architecture. All remaining ablations were pre-trained and therefore show improved performance. (1) and (2) are pre-trained using TimeCL and TimeReco, respectively, with minimal improvement. The next two ablations pre-train using both objectives but (3) + (2) uses random sampling to generate sub-sequences, while (4) + (1) masks a constant *length* of data instead of constant *time*. Results show that neither beats PrimeNet, substantiating time-sensitive sampling and constant time masking.

Conclusion

We propose a self-supervised representation learning approach to model irregular and asynchronous multivariate time-series. We use time-sensitive contrastive learning that preserves an approximate sampling density distribution of the data to learn from representative sub-sequences. We use time-sensitive data reconstruction to mask a fixed duration of data, instead of a fixed number of points, making reconstruction tractable across regions of varying sampling density. Our pre-trained model is then fine-tuned on downstream end tasks. Experiment results show that PrimeNet outperforms both fully- and semi-supervised irregular time-series and self-supervised regular time-series methods on classification, interpolation and regression tasks across several real-world datasets. In future, we plan to apply this to irregular time-series forecasting and unsupervised anomaly detection.

Acknowledgments

This work was supported in part by the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, under award 2018-JU-2779. This work was also sponsored in part by National Science Foundation Convergence Accelerator under award OIA-2040727 as well as generous gifts from Google, Adobe, and Teradata. Ranak is partially funded by a Graduate Prize Fellowship from Halıcıoğlu Data Science Institute. Any opinions, findings, and conclusions or recommendations expressed herein are those of the authors and should not be interpreted as necessarily representing the views, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for government purposes not withstanding any copyright annotation hereon.

References

Che, Z.; Purushotham, S.; Cho, K.; Sontag, D.; and Liu, Y. 2018. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1): 1–12.

Chen, R. T.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. K. 2018. Neural ordinary differential equations. *Advances in neural information processing systems*, 31.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, 1597–1607. PMLR.

Chowdhury, R. R.; Adnan, M. A.; and Gupta, R. K. 2020. Real-time principal component analysis. *ACM Transactions on Data Science*, 1(2): 1–36.

Chowdhury, R. R.; Zhang, X.; Shang, J.; Gupta, R. K.; and Hong, D. 2022. TARNet: Task-Aware Reconstruction for Time-Series Transformer. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA*, 14–18.

Franceschi, J.-Y.; Dieuleveut, A.; and Jaggi, M. 2019. Unsupervised scalable representation learning for multivariate time series. *arXiv preprint arXiv:1901.10738*.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9729–9738.

Hogan, W.; Li, J.; and Shang, J. 2022. Fine-grained Contrastive Learning for Relation Extraction. *ArXiv*, abs/2205.12491.

Horn, M.; Moor, M.; Bock, C.; Rieck, B.; and Borgwardt, K. 2020. Set functions for time series. In *International Conference on Machine Learning*, 4353–4363. PMLR.

Jaiswal, A.; Babu, A. R.; Zadeh, M. Z.; Banerjee, D.; and Makedon, F. 2020. A survey on contrastive self-supervised learning. *Technologies*, 9(1): 2.

Johnson, A. E.; Pollard, T. J.; Shen, L.; Lehman, L.-w. H.; Feng, M.; Ghassemi, M.; Moody, B.; Szolovits, P.; Anthony Celi, L.; and Mark, R. G. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3(1): 1–9.

Kaluža, B.; Mirchevska, V.; Dovgan, E.; Luštrek, M.; and Gams, M. 2010. An agent-based approach to care in independent living. In *International joint conference on ambient intelligence*, 177–186. Springer.

Kazemi, S. M.; Goel, R.; Eghbali, S.; Ramanan, J.; Sahota, J.; Thakur, S.; Wu, S.; Smyth, C.; Poupart, P.; and Brubaker, M. 2019. Time2vec: Learning a vector representation of time. *arXiv preprint arXiv:1907.05321*.

Lechner, M.; and Hasani, R. 2020. Learning long-term dependencies in irregularly-sampled time series. *arXiv preprint arXiv:2006.04418*.

Li, J.; Shang, J.; and McAuley, J. 2022. UCTopic: Unsupervised Contrastive Learning for Phrase Representations and Topic Mining. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 6159–6169. Dublin, Ireland: Association for Computational Linguistics.

Li, S.; Chowdhury, R. R.; Shang, J.; Gupta, R. K.; and Hong, D. 2021. UniTS: Short-Time Fourier Inspired Neural Networks for Sensory Time Series Classification. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, 234–247.

Marlin, B. M.; Kale, D. C.; Khemani, R. G.; and Wetzel, R. C. 2012. Unsupervised pattern discovery in electronic health care data using probabilistic clustering models. In *Proceedings of the 2nd ACM SIGHIT international health informatics symposium*, 389–398.

Neil, D.; Pfeiffer, M.; and Liu, S.-C. 2016. Phased lstm: Accelerating recurrent network training for long or event-based sequences. *Advances in neural information processing systems*, 29.

Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Pham, T.; Tran, T.; Phung, D.; and Venkatesh, S. 2017. Predicting healthcare trajectories from medical records: A deep learning approach. *Journal of biomedical informatics*, 69: 218–229.

Rubanova, Y.; Chen, R. T.; and Duvenaud, D. K. 2019. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32.

Shukla, S. N.; and Marlin, B. M. 2019. Interpolation-prediction networks for irregularly sampled time series. *arXiv preprint arXiv:1909.07782*.

Shukla, S. N.; and Marlin, B. M. 2021. Multi-time attention networks for irregularly sampled time series. *arXiv preprint arXiv:2101.10318*.

Silva, I.; Moody, G.; Scott, D. J.; Celi, L. A.; and Mark, R. G. 2012. Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. In *2012 Computing in Cardiology*, 245–248. IEEE.

Tan, C. W.; Bergmeir, C.; Petitjean, F.; and Webb, G. I. 2021. Time Series Extrinsic Regression. *Data Mining and Knowledge Discovery*, 1–29.

Tan, Q.; Ye, M.; Yang, B.; Liu, S.; Ma, A. J.; Yip, T. C.-F.; Wong, G. L.-H.; and Yuen, P. 2020. Data-gru: Dual-attention time-aware gated recurrent unit for irregular multivariate time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 930–937.

Tonekaboni, S.; Eytan, D.; and Goldenberg, A. 2021. Unsupervised representation learning for time series with temporal neighborhood coding. *arXiv preprint arXiv:2106.00750*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Yue, Z.; Wang, Y.; Duan, J.; Yang, T.; Huang, C.; Tong, Y.; and Xu, B. 2021. TS2Vec: Towards Universal Representation of Time Series. *arXiv preprint arXiv:2106.10466*.

Zerveas, G.; Jayaraman, S.; Patel, D.; Bhamidipaty, A.; and Eickhoff, C. 2021. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2114–2124.

Zhang, X.; Chowdhury, R. R.; Hong, D.; Gupta, R. K.; and Shang, J. 2023. Modeling Label Semantics Improves Activity Recognition. *arXiv preprint arXiv:2301.03462*.

Zhang, X.; Chowdhury, R. R.; Shang, J.; Gupta, R.; and Hong, D. 2022. ESC-GAN: Extending Spatial Coverage of Physical Sensors. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 1347–1356.

Zhang, Y. 2019. ATTAIN: Attention-based Time-Aware LSTM Networks for Disease Progression Modeling. In *In Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI-2019), pp. 4369-4375, Macao, China.*