Changepoint Detection using Self Supervised Variational AutoEncoders

Sourav Chatterjee [1]

Abstract

Changepoint Detection methods aim to find locations where a time series shows abrupt changes in properties, such as level and trend, which persist with time. Traditional parametric approaches assume specific generative models for each segment of the time series, but often, the complexities of real time series data are hard to capture with such models. To address these issues, in this paper, we propose VAE-CP, which uses a variational autoencoder with self supervised loss functions to learn informative latent representations of time series segments. We use traditional hypothesis test based and Bayesian changepoint methods in this latent space of normally distributed latent variables, thus combining the strength of self-supervised representation learning, with parametric changepoint modeling. This proposed approach outperforms traditional and previous deep learning based changepoint detection methods in synthetic and real datasets containing trend changes.

1. Introduction

The problem of detecting abrupt changes in a sequence of observations, known as the changepoint detection problem, has application in a number of domains, such as climate modeling(Manogaran & Lopez, 2018), human activity recognition(Cleland et al., 2014), speech recognition(Panda & Nayak, 2016), finance(Lavielle & Teyssiere, 2007) etc. Such algorithms find particular applications in monitoring internet infrastructure, such as regressions in large codebases(Valdez-Vivas et al., 2018)and network traffic(Kurt et al., 2018). Changepoint Detection is also useful in the context of time series forecasting(Taylor & Letham, 2018).

A number of parametric changepoint detection methods are used in practice, which compare past and future time series intervals using a dissimilarity metric. Burg and Williams(van den Burg & Williams, 2020) presents an evaluation of traditional changepoint detection methods on a wide variety of real life time series. Many of these changepoint detection algorithms assume that the data in a segment of the time series, is generated from a parametric model, and attempt to detect changes in the model parameters, as evidence of changepoint.

For instance, Bayesian Online Changepoint Detection(BOCPD)(Adams & MacKay, 2007) algorithm assumes a Bayesian predictive model for a segment. Many implementations(Pagotto, 2019) assume that the data in a segment comes from a normal distribution, a condition that is violated by many real life time series(van den Burg & Williams, 2020).Other predictive models for BOCPD have also been explored , such as Gaussian processes(Saatçi et al., 2010; Han et al., 2019), Bayesian LSTM(Detommaso et al., 2019). Similarly, Prophet(Taylor & Letham, 2018) models each segment using a Bayesian linear regression, and CUSUM(Page, 1954) uses various hypothesis tests(Flynn & Yoo, 2019; Li et al., 2010; Healy, 1987) to detect dissimarities between segments. However, these methods are unable to capture the complexities of time series data, such as (multiple)seasonalities, autocorrelations etc. They also need to be specifically tuned for each time series, using expert knowledge, to guess the generative model for a particular time series.

Deep learning based approaches have been very successful in learning informative latent representations of sequential data, in natural language processing(Devlin et al., 2018) using self-supervision. Predicting neighboring time series segments, have also resulted in similarly powerful embeddings for time series clustering tasks(Franceschi et al., 2019). Similar approaches have recently been applied to the changepoint detection problem. KL-CPD(Chang et al., 2019) uses deep kernel based hypothesis tests. The approaches closest to our approach are TIRE(De Ryck et al., 2020) which uses autoencoders and $TS-CP^2$(Deldari et al., 2020), which uses contrastive learning, to learn latent representations, and find dissimilarities. However, the success of these algorithms are limited to large, multivariate time series, in which their efficacies are demonstrated. While these algorithms attempt to learn good latent representations, they fail to combine them with the strength of parametric changepoint detection algorithms.

[1]Facebook, Menlo Park, CA, USA. Correspondence to: Sourav Chatterjee <souravc83@fb.com>.

ICML 2021 Time Series Workshop, Virtual Workshop, 2021. Copyright 2021 by the author(s).

In the paper, we propose a self-supervised learning approach, VAE-CP that combines the power of deep learning architectures to learn informative latent representations, with the application of parametric changepoint detection methods, which have been very successful in settings, where their model assumptions are strictly satisfied. In particular, we consider the Variational AutoEncoder(Kingma & Welling, 2013), where the latent variables are drawn from a normal distribution. Hypothesis tests based algorithms, such as CUSUM(Page, 1954) and bayesian algorithms such as BOCPD(Adams & MacKay, 2007) are well known algorithms when the data in a segment comes from a normal distribution, and hence we can easily run changepoint detection algorithms in the latent space. We specifically concentrate on identifying trend changes, but the methods can be expanded to other changes in future.

2. Method

2.1. Problem Formulation

Given an time-series $X = (x_1, x_2, x_3 .....x_T)$, $x_i \in \mathbb{R}$, we are interested in finding changepoints $\tau_1, \tau_2 .....\tau_k$, such that $\tau_1 < \tau_2 < ...\tau_k$. We assume that a segment of the time series $x_{\tau_i}, x_{\tau_i+1}, x_{\tau_i+2} ......x_{\tau_{i+1}}$ has some common characteristic, such as the same seasonality, trend, and level. However, unlike conventional time series models, we will refrain from assuming a known distribution for the segment.

We will work with windows of size w. At the current time t, we consider the window $X^w_t = \{x_{t-w}, x_{t-w+1}, ...x_{t-1}\}$. We aim to find changepoints based on dissimilarities between consecutive windows in the latent space.

2.2. Self Supervised VAE

One of the main contributions of the paper is, to learn a latent representation using a Variational AutoEncoder, and use self-supervision as a signal. While previous studies on changepoint detection have used various encoding architectures(De Ryck et al., 2020; Deldari et al., 2020), we use VAEs since VAEs are a stochastic generative model, which produces calibrated probabilities. The latent variables come from a normal distribution, and this is the setting, for which parametric changepoint algorithms are designed for. Hence, our contribution, is to create embeddings, which can be easily fed to powerful parametric changepoint detection algorithms.

We wish to learn an encoder, from an window X^w_t, to its corresponding latent value z_t, in a continuous space, such that the encoding is informative of the segment of the time series X^w_t belongs to. We assume that $z_t \sim N(\mu_t, diag(\sigma^2_t))$, where $[\mu_t, \sigma_t] \sim \phi_{encoder}(X^w_t)$.

The decoder produces a time series window of identical

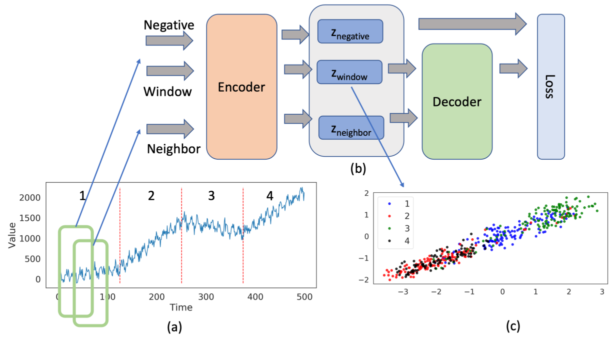

Figure 1. (a) The original time series, with 3 changepoints, at 125, 250 and 375. Original/negative/neighboring windows are passed through a VAE. (b) The VAE tries to minimize a reconstruction loss, a regularization loss and a triplet loss. (c) shows the distributions of latent variables, qualitatively demonstrating the separation of the different segments. Colors represent the four segments, same as the numbers marked on the original time series.

length, $\hat{X}^w_t = \psi_{decoder}(z_t)$. To encourage a representation of z_t that contains information about the segment it belongs to, we train latent representation to be similar to a nearby "neighbor" window. For each window X^w_t, we randomly sample another window $X^w_{t'}$, such that the windows are nearby, i.e. $|t - t'| \leq L$, where L is an user defined limit, which determines the smoothing. This helps us obtain latent representations that are independent of local effects within the segment, Although VAEs are trained to minimize reconstruction error traditionally, similar self-supervised approaches have been explored(Zhu et al., 2020) for sequential data.

We optimize the ELBO loss of the VAE,

$$\mathcal{L}_{VAE} = \mathbb{E}_{z_t \sim q}[-p(X^w_{t'}|z_t) + \gamma D_{KL}(q(z_t)||p(z))] \quad (1)$$

where γ is a factor that determines the strength of regularization.

In addition to $\mathcal{L}_{VAE}$, we also optimize a triplet loss $\mathcal{L}_{triplet}$, used commonly in self-supervised learning(Wang & Gupta, 2015). We add a random trend to the the values in X^w_t, to construct a negative sample, and obtain its corresponding latent variable z^{neg}_t. To encourage windows with dissimilar trends to be separated in the latent space, we define

$$\mathcal{L}_{triplet} = max(D(z_t, z^{neighbor}_t) - D(z_t, z^{neg}_t) + m, 0) \quad (2)$$

where m is a margin, and D is an Euclidean distance. Negative sampling techniques have been used in the past in self-supervised learning for time series, in changepoint detection(Deldari et al., 2020) and clustering(Franceschi et al., 2019).

The architecture of the encoder $\phi_{encoder}$ consists of a fully connected layer, followed by RELU activation, to generate a hidden state. We then use two separate fully connected layers to generate μ_t and $log(\sigma_t)$. The decoder $\psi_{decoder}$ similarly consists of an MLP with a single hidden layer and RELU activation.

To train the algorithm, we sample minibatches corresponding to multiple $(X^w_t, X^w_{t'}, X^w_{t-neg})$. Drawing multiple random samples corresponding to the same X^w_t is helpful in practice. Once the algorithm is trained, we generate the the corresponding latent representations $z_i \forall i \in [w, T]$. To qualitatively evaluate the quality of the embeddings, in Figure 1, we visualize the latent space for an example synthetic time series with three changepoints. We color the four segments with different colors, and we can observe the partial separation of the latent variables, in this space. This separation is further clear in Figure 2, where we plot the time series of the latent variables.

2.3. Changepoint Detection

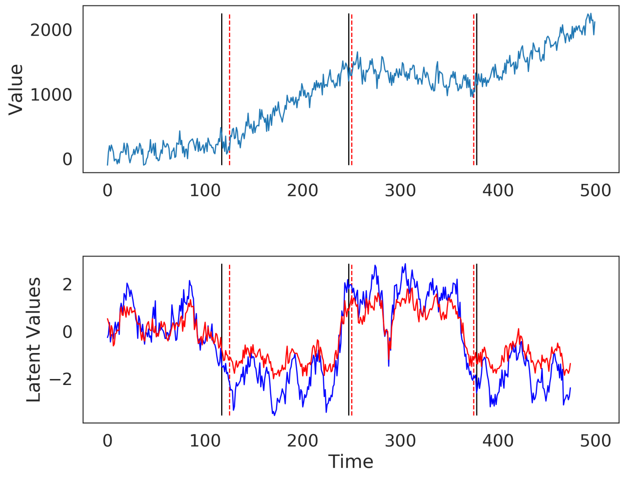

Figure 2. Original TimeSeries(above) with detected changepoints (black), and ground truth(red). Time series of each dimension of the 2D latent variable(below).

We use multivariate changepoint detection techniques in the low dimensional latent space $Z = \{z_w, z_{w+1}, z_{w+2}, ...z_T\}$. We rely on two conventional changepoint detection techniques, hypothesis tests and Bayesian Online Changepoint Detection(BOCPD). We have used the techniques with some modification, since in our case, as seen in Figure 2, we observe a more gradual change, rather than an abrupt change, that the algorithms are designed for. This is because there are w windows around a changepoint, which contain data points from the previous and current segments.

Once we obtain the changepoints from these two methods, our final result is obtained by taking an union of the changepoints from both the methods. If two changepoints are within a margin, we take their average as the result. Algorithm 1 sketches the full implementation of VAE-CP.

Hypothesis Test Following reasoning similar to the CUSUM algorithm(Page, 1954; Flynn & Yoo, 2019; Li et al., 2010; Healy, 1987), we sequentially perform hypothesis tests at every time point t, over windows of the latent variable. Defining w_H as the window size, for hypothesis test, we define the left and right windows as $Z^l_t = \{z_{t-w_H}, z_{t-w_H+1}, ...z_{t-1}\}$ and $Z^r_t = \{z_t, z_{t+1}, z_{t+2}, ...z_{t+w_H-1}\}$.

Since $z_t \sim N(\mu_t, diag(\sigma^2_t))$, we use the Hoteling's T2 test, and obtain the test statistic and p-values from the test. During post-processing, we identify regions in the time series, where the p-value is below a threshold for a duration larger than the original window size w.This region correspond to windows which contain data points from before and after changepoint. We then find the peak of the test statistic inside this region and assign this as the changepoint.

Multivariate BOCPD Bayesian Online Changepoint Detection(BOCPD)(Adams & MacKay, 2007) works by finding the posterior distribution of the run length r_t, given the data. r_t corresponds to the number of time points since the last changepoint. The basic idea of BOCPD is to define a recursive equation for the joint distribution

$$P(r_t, z_{1:t}) = \sum_{r_{t-1}} P(r_{t-1}, z_{1:t-1}) P(z_t|r_{t-1}, z_{1:t-1}) P(r_t|r_{t-1}) \tag{3}$$

See Adams and McKay(Adams & MacKay, 2007) for a more detailed description of BOCPD. Key to implementing BOCPD is the specification of a Bayesian "predictive model", which tells us the probability of the next point, when we consider only the data points since the last changepoint, i,e. $P(z_t|r_{t-1}, z_{1:t-1})$. We assume a constant $P(r_t|r_{t-1})$, and can hence solve the equations recursively, once a predictive model is specified. Since z_t comes from a Multivariate Normal distribution, we assume a multivariate normal predictive model. We use a Normal-Wishart prior on the parameters of the normal distribution.

$$z \sim N(\mu, \Lambda) \tag{4}$$

$$p(\mu, \Lambda) = N(\mu|\mu_0, (\kappa\Lambda)^{-1}) Wi_\nu(\Lambda|T) \tag{5}$$

To modify this for our gradual change use case, we subsample the time series(uniform subsampling), so that the change is more abrupt. The tradeoff is greater uncertainty in determining the location of the changepoint.

Algorithm 1 VAE-CP

```
**Input:** data $X = (x_1, x_2....x_T)$, window size $w$
$X_1^w, X_2^w$... = createWindows(X)
**for** $i = 1$ **to** $N_{epoch}$ **do**
    Sample a minibatch $(X_t^w, X_{t'}^w, X_{t-neg}^w)$
    $\mathcal{L} = \mathcal{L}_{VAE}(X_t^w, X_{t'}^w) + \mathcal{L}_{triplet}(X_t^w, X_{t'}^w, X_{t-neg}^w)$
    Update $\phi_{encoder}$ and $\psi_{decoder}$
**end for**
**for** $t = w$ **to** $T$ **do**
    $z_t = \phi_{encoder}(X_t^w)$
**end for**
$CP_{Hoteling}$ = runHoteling(z)
$CP_{BOCPD}$ = runBOCPD(z)
$CP_{Final}$ = Union( $CP_{Hoteling}, CP_{BOCPD}$)
```

3. Experiments

Each segment of our synthetic data contains trend, seasonality and Gaussian noise, characteristics of many real world business time series(Taylor & Letham, 2018).

$$x_t = \beta_s t + c_s + A sin(\omega t) + N(0, \sigma^2) \tag{6}$$

where the slope β_s and the intercept c_s are specific to the segment. Slope changes are drawn from a normal distribution, and intervals between changepoints are drawn from a geometric distribution, assuming changepoints as independent events. See Appendix for a more detailed description of the synthetic data.

A huge challenge for us, is to find real datasets with reliable changepoint labeling. Burg and Williams(van den Burg & Williams, 2020) evaluate changepoint methods using manual labels from multiple annotators.We choose three datasets from their collection, which contain changepoints in trend, and have trend and complex seasonal patterns. Construction reports monthly data on the total private construction spending in USA, and JFK and LGA datasets contain monthly data on number of passengers departing the respective airports. We choose these datasets since they contain changepoints in trend, and have trend and complex seasonal patterns.

Our evaluation procedure exactly follows Burg and Williams(van den Burg & Williams, 2020), who modify the definition of precision and recall, to accomodate the margin of error around the true changepoint(Killick et al., 2012; van den Burg & Williams, 2020), and then calculate precision, recall and F-score. They also calculate F-scores averaging labels from multiple annotators.

4. Results

A summary of our results can be found in Table 1. We compare VAE-CP with three baseline methods, Prophet(Taylor & Letham, 2018), BOCPD(Adams & MacKay, 2007) as implemented by the OCP package(Pagotto, 2019) in R and TIRE(De Ryck et al., 2020). We choose Prophet for comparison, since we are focused on trend changes, and Prophet's generative model is identical to the synthetic data generation process (linear regression with trend changes, specified by changepoints). We choose BOCPD since it is one of the best models empirically, in an evaluation of real world changepoints(van den Burg & Williams, 2020). TIRE(De Ryck et al., 2020), using autoencoders to learn latent representations, is the closest deep learning approach to ours.

Method	Synthetic	Construction	JFK	LGA
BOCPD	0.233	0.547	0.394	**0.598**
Prophet	0.384	0.324	0.279	0.251
TIRE	0.194	0.669	0.437	0.34
VAE-CP	**0.499**	**0.707**	**0.531**	0.369

Table 1. Comparison of VAE-CP with baseline methods on synthetic and real datasets

We observe that VAE-CP consistently beats Prophet, BOCPD and TIRE in both the synthetic and real datasets, except LGA Passengers, where it is the second best performing algorithm. We also notice that, even though the synthetic data generating mechanism mimics the model of Prophet, it tends to have too many false positives near changepoints, and even in this case VAE-CP has a higher F-score. Although BOCPD is designed to detect level changes, it shows the best performance in LGA passengers data, consistent with its good empirical performance on real world data(van den Burg & Williams, 2020).

5. Conclusion

In this work, we proposed a novel time series changepoint detection algorithm, VAE-CP, which combines parametric changepoint detection and self supervised representation learning approaches. We showed that VAE-CP is able to out-perform baseline parametric and deep learning methods on synthetic and real datasets for trend changes.

In future, we would like to study the effect of parameters, such as window size on VAE-CP performance, as well as an ablation study of the various algorithmic choices we have made. We would like to explore VAE-CP on different kinds of changepoints, such as level shifts, and on a larger set of real life datasets, including multivariate timeseries. We hope that VAE-CP, due to its flexibility, is able to perform well, across a large number of different synthetic and real life changepoint detection problems.

References

Adams, R. P. and MacKay, D. J. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

Chang, W.-C., Li, C.-L., Yang, Y., and Póczos, B. Kernel change-point detection with auxiliary deep generative models. *arXiv preprint arXiv:1901.06077*, 2019.

Cleland, I., Han, M., Nugent, C., Lee, H., McClean, S., Zhang, S., and Lee, S. Evaluation of prompted annotation of activity data recorded from a smart phone. *Sensors*, 14 (9):15861–15879, 2014.

De Ryck, T., De Vos, M., and Bertrand, A. Change point detection in time series data using autoencoders with a time-invariant representation. *arXiv preprint arXiv:2008.09524*, 2020.

Deldari, S., Smith, D. V., Xue, H., and Salim, F. D. Time-series change point detection with self-supervised contrastive predictive coding. *arXiv preprint arXiv:2011.14097*, 2020.

Detommaso, G., Hoitzing, H., Cui, T., and Alamir, A. Stein variational online changepoint detection with applications to hawkes processes and neural networks. *arXiv preprint arXiv:1901.07987*, 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Flynn, T. and Yoo, S. Change detection with the kernel cumulative sum algorithm. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 6092–6099. IEEE, 2019.

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

Han, J., Lee, K., Tong, A., and Choi, J. Confirmatory bayesian online change point detection in the covariance structure of gaussian processes. *arXiv preprint arXiv:1905.13168*, 2019.

Healy, J. D. A note on multivariate cusum procedures. *Technometrics*, 29(4):409–412, 1987.

Killick, R., Fearnhead, P., and Eckley, I. A. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500): 1590–1598, 2012.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kurt, B., Yıldız, Ç., Ceritli, T. Y., Sankur, B., and Cemgil, A. T. A bayesian change point model for detecting sip-based ddos attacks. *Digital Signal Processing*, 77:48–62, 2018.

Lavielle, M. and Teyssiere, G. Adaptive detection of multiple change-points in asset price volatility. In *Long memory in economics*, pp. 129–156. Springer, 2007.

Li, S.-Y., Tang, L.-C., and Ng, S.-H. Nonparametric cusum and ewma control charts for detecting mean shifts. *Journal of Quality Technology*, 42(2):209–226, 2010.

Manogaran, G. and Lopez, D. Spatial cumulative sum algorithm with big data analytics for climate change detection. *Computers & Electrical Engineering*, 65:207–221, 2018.

Page, E. S. Continuous inspection schemes. *Biometrika*, 41 (1/2):100–115, 1954.

Pagotto, A. *ocp: Bayesian Online Changepoint Detection*, 2019. URL `https://CRAN.R-project.org/package=ocp`. R package version 0.1.1.

Panda, S. P. and Nayak, A. K. Automatic speech segmentation in syllable centric speech recognition system. *International Journal of Speech Technology*, 19(1):9–18, 2016.

Saatçi, Y., Turner, R. D., and Rasmussen, C. E. Gaussian process change point models. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 927–934. Citeseer, 2010.

Taylor, S. J. and Letham, B. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.

Valdez-Vivas, M., Gocmen, C., Korotkov, A., Fang, E., Goenka, K., and Chen, S. A real-time framework for detecting efficiency regressions in a globally distributed codebase. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 821–829, 2018.

van den Burg, G. J. and Williams, C. K. An evaluation of change point detection algorithms. *arXiv preprint arXiv:2003.06222*, 2020.

Wang, X. and Gupta, A. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE international conference on computer vision*, pp. 2794–2802, 2015.

Zhu, Y., Min, M. R., Kadav, A., and Graf, H. P. S3vae: Self-supervised sequential vae for representation disentanglement and data generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6538–6547, 2020.

A. Datasets

A.1. Synthetic Data

Ten Synthetic Datasets are generated, with segments containing trend and seasonality as described in the main section. The seasonality is always weekly. The trends changes between segments are drawn from a normal distribution.

$$\beta_s = \beta_{s-1} + N(0, \sigma^2) \tag{7}$$

The length of the timeseries is 500, and the intervals between changepoints are drawn from a geometric distribution, with an average interval length of $\tau_{dist} = 125$.

$$\tau_s = \tau_{s-1} + G(\frac{1}{\tau_{dist}}) \tag{8}$$

The intercept c_s is adjusted to keep the segments continuous.

We report the average F-score, for the ten synthetic datasets in the results.

A.2. Real Datasets

We use three real datasets, which have been used in Burg and Williams(van den Burg & Williams, 2020) and open sourced in the TCPD package in github(https://github.com/alan-turing-institute/TCPD). We use the manual annotations and evaluation method in the TCPD package. For the construction dataset, all of the annotations miss the obvious sharp movement in trend (from increase to decrease), around 150. Hence, we add another manual annotation of our own, in addition to the existing ones([95, 120, 155, 215]. For the other datasets, we use the existing annotations. For evaluation, we exactly follow Burg and Williams(van den Burg & Williams, 2020).

B. Baseline Methods

BOCPD We use the implementation of BOCPD, in the OCP package in R(Pagotto, 2019). Our parameters and results are exactly identical with Burg and Williams(van den Burg & Williams, 2020) (default F-score), who evaluate multiple datasets using OCP.

Prophet We use the open source python implementation of Prophet(https://github.com/facebook/prophet). Again, we use arguments exactly the same as Burg and Williams(van den Burg & Williams, 2020)(there are some minor differences in results, because they use the R implementation), with all the seasonalities set to ”auto” and changepoint range to 1.0. We have experimented with reducing the changepoint prior scale parameter, but this has not helped. For the synthetic data, we give Prophet an advantage, by setting weekly seasonality as True, since we know this data has weekly seasonality.

TIRE We implement TIRE(De Ryck et al., 2020) with the exact parameters specified in the example notebook, from the open source package(https://github.com/deryckt/TIRE).

C. Architecture Choices

We use a window size $w = 20$. We also set $L = 20$, which is the region in which we search for a neighboring window. This parameter is critical, and choosing it to be higher than the seasonality helps in erasing the effects of seasonality. Our goal is to learn a latent representation, that is representative of the segment, and controlling L helps us achieve that.

Both for our encoder and decoder, we use a fully connected layer, with 64 units in the hidden layer. We have also experimented with RNNs in the encoder, similar to KL-CPD(Chang et al., 2019), but this has not worked well for us.We train the network for 300 epochs.