Multi-Scale Change Point Detection in Multivariate Time Series

Zahra Ebrahimzadeh, Samantha Kleinberg
Department of Computer Science
Stevens Institute of Technology
Hoboken, NJ 07030
{zebrahim, samantha.kleinberg}@stevens.edu

Abstract

A core problem in time series data is learning when things change. This problem is especially challenging when changes appear gradually and at varying timescales, such as in health. Convolutional Neural Networks (CNNs) have the potential to recognize and localize complex patterns, but are sensitive to scale. We propose a new class of scale and shift invariant neural networks that augment CNNs with trainable wavelet layers. Experimentally, we demonstrate that this approach can be used to more accurately detect gradual change points in multivariate time series.

1 Introduction

Detecting changes in time-series data has important applications in various domains, and is especially important in health, where changes in measurements from a patient may indicate onset of illness or increasing illness severity. Change point detection aims to find these critical points where a system's structure or parameters change. For multivariate data this problem is challenging, as changes may occur at different timescales, on a subset of signals, and with different durations.

Change point detection has mainly been treated as a problem of finding when something anomalous, or not explained by the existing model, occurs. However, these approaches have only worked for abrupt changes. At the same time, there have been significant advances in pattern recognition due to Convolutional Neural Networks (CNN) [LB+95] on many tasks [KGB14, KSH12]. A key advantage of CNNs is that they learn shift-invariant transforms to recognize and localize complex patterns, which could be applied to find patterns of changes. However, they are sensitive to scale, which limits their use for detecting change points in realistic multivariate data where changes occur at multiple scales. Wavelets, on the other hand, can capture such transitions.

We propose Deep Wavelet Networks (DWNs), which add a trainable wavelet layer to CNNs to enable gradual change point detection at multiple scales. This approach frees the user from having to pick the right window size for a sliding window, or the right kernel size and pooling stride for CNN. Our proposed neural wavelet layers consist of multi-resolution convolutions with learnable kernels, and we show their effectiveness in gradual change point detection. We introduce our approach, and show its improvement over the state of the art using simulated datasets.

2 Related Work

Most change point detection algorithms are based on time series modeling, which requires prior domain knowledge. Sequence models such as Hidden Markov Models ([MAL15]) are common choices, but are limited to explain simple patterns. Bayesian Online Change Point Detection BOCPD [AM07] uses an online algorithm to identify the likelihood of a change at any given timepoint

relative to a model. Many works rely on statistical tests that measure divergence between adjacent intervals of observed data [CMO16, BO16, IPK16], but again these require prior models.

Machine learning methods on the other hand, can identify patterns that are difficult to explicitly model. For instance, [GKVL06] uses a one-class SVM to classify seizure onset and [SWLD13] later incorporated temporal dependency. However, if some examples of the event of interest (i.e., change points to be detected) are available, these approaches cannot take advantage of that.

Supervised learning methods have been proposed ([YKNS13, HRVB13]) to address these issues, but do not explicitly account for gradual changes. [HMEYC14] aims to detect gradual change points by analyzing prediction error of a learned model. [BF$^+$17] extends BOCPD to detect abnormal segments, as opposed to points, to detect gradual changes. We exploit wavelets to explicitly process time series in a multi-resolution way, which results in scale-invariant change point detection.

Nonparametric methods have been used to address similar problems [CTK$^+$17, JNIH16, QAWZ15, GMRS16, LXDS15]. Although these methods do not rely on assumptions about data distribution, they are less efficient with large data. We instead use deep neural networks, which can learn complex patterns such as objects in images [KSH12], while being computationally efficient at test time.

3 Method

We propose a novel family of deep learning architectures called a Deep Wavelet Network (DWN), which combine our proposed Neural Wavelet Layer (NWL) with a CNN.

Neural Wavelet Layer First, we replace the convolution operation conventionally used in a CNN, with a wavelet transform. This Neural Wavelet Layer (NWL) can be seen as a set of multi-scale convolutions, each with a learnable kernel. The wavelet layer takes as input a multivariate time series (one dimension and multiple channels), and produces multiple feature maps, each of which is a pyramid of convolution responses (fig. 1).

The NWL uses the filter bank technique for discrete wavelet transform. Given a pair of separating convolutional kernels (typically a low-pass and a high-pass kernel), it convolves the signal with both, outputs the high-pass response, and down-samples the low-pass response for the next iteration. This process is repeated, with each iteration yielding a higher level of the output pyramid. The difference between NWL and the classical wavelet transform is that the filter bank is initialized with random numbers and trained, using backpropagation, with the rest of the network. NWL takes as input a multivariate time series $X \in \mathbb{R}^{T \times c}$, where T is the length of the time series and c is the number of channels/dimensions/variables. With kernels $K_L, K_U \in \mathbb{R}^{\tau \times c}$, it computes L_1 and U_1, such that

$$L_1 = \omega(X * K_L) \quad , \quad U_1 = \omega(X * K_U), \tag{1}$$

where $*$ denotes convolution and ω indicates downsampling. A downsampling by a factor of 2 can be implemented by discarding every other row of L_1 and U_1. Finer downsampling functions can be implemented by linear interpolation. Now (1) can be rewritten as:

$$L_1[t] = \sum_{i=(-\tau)}^{\tau} \sum_{j=1}^{c} X[2t{+}i, j] \times K_L[\tau{+}i{+}1, j] \qquad U_1[t] = \sum_{i=(-\tau)}^{\tau} \sum_{j=1}^{c} X[2t{+}i, j] \times K_U[\tau{+}i{+}1, j]. \tag{2}$$

Note that $2t$ denotes downsampling by 2. Given L_k and U_k it computes L_{k+1} and U_{k+1} such that:

$$L_{k+1} = \omega(L_k * K_L) \quad , \quad U_{k+1} = \omega(U_k * K_U). \tag{3}$$

This operation is repeated for a predefined number of levels, or until the length of L_κ and U_κ are smaller than a threshold. The output of this layer is the union of all U_k and the last L_κ stacked together. In practice, following CNNs, we use multiple filter pairs to do different wavelet transforms in parallel, and produce a set of feature maps, each being a pyramid of the same size. This can also be represented as a single pyramid, each of whose elements is a vector instead of a real value.

The advantage of an NWL over a convolution layer is that wavelets can encode input with multiple granularities at once, whereas it takes convolution multiple layers to hierarchically decrease the granularity. Hence, a convolutional layer cannot explicitly detect the same pattern at different scales. This is crucial because in practice, similar types of changes may happen at different scales. NWL

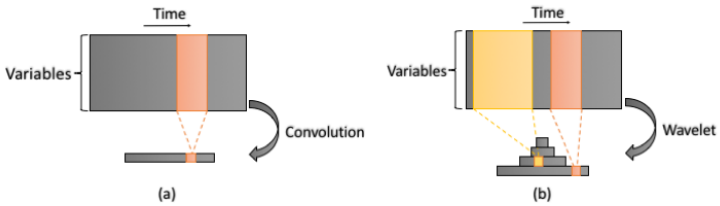


Figure 1: Illustration of (a) convolutional layer; (b) the proposed Neural Wavelet Layer.

can also be used as a layer of a deep network, in composition with other neural layer types such as convolutional and fully connected layers. For example, the input to a wavelet layer can be the output of a convolutional layer. To apply a convolution on the output of a wavelet layer, one should apply the convolution on each level of the wavelet pyramid, and then integrate the result by concatenation.

Deep Wavelet Networks We propose a Deep Wavelet Network (DWN) as a composition of an NWL and multiple convolutional layers. In our experiments, an input time series with arbitrary size is first transformed through a NWL into a pyramid-shaped representation. Then each level of this pyramid is fed into a CNN. The same CNN architecture with the same variables is applied on different levels of the pyramid. A logistic regression layer is built on the output of each CNN level.

The output of this network is a pyramid of classification scores. Next, we upsample each level to be the same size as the lowest level of the pyramid, and combine all layers using arithmetic mean. This will produce classification scores for each time window, at the lowest level of granularity. We use fine-grained ground truth at the same level of granularity to train this network end-to-end. We optimize the cross entropy loss using stochastic gradient descent, to find parameters of the CNN and NWL (wavelet kernels K_L and K_U).

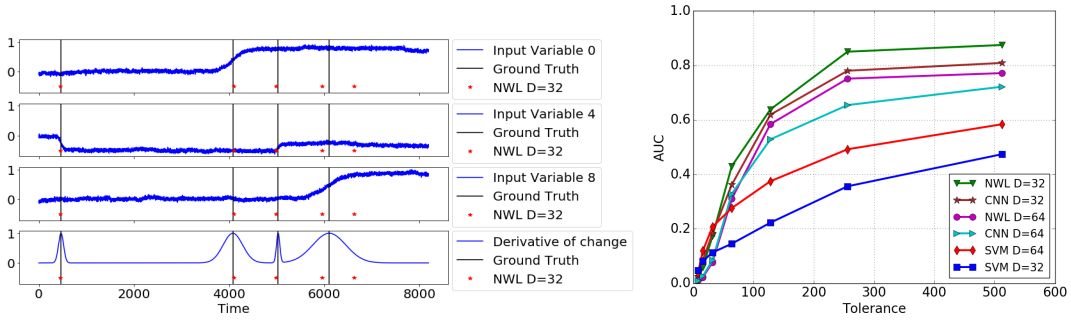
4 Experiment

Baseline We use two baselines: CNN, and SVM with sliding window feature extraction. For CNN we use the same architecture as DWN, but without the wavelet layer. Thus, we feed the input time series to a CNN with same convolutional kernels sizes as the DWN, and use the output to determine change points. We do not compare against BOCPD as it is meant for univariate time series. For the SVM, we use different window sizes (32 and 64) to simulate the same down-sampling ratios as DWN/CNN. We extracted features from each window by splitting it into two equal segments and concatenating the mean of all signal dimensions at each segment.

Implementation Both the proposed method and CNN baseline were implemented with Tensorflow [AAB+16]. We report results for the following two convolutional architectures, where we use the notation $x : y : z$ for a convolutional layer where x is the kernel size, y is the number of output feature maps, and z is the pooling stride. The architectures are 1) $[5 : 32 : 2], [5 : 64 : 2], [5, 64 : 2], [5 : 64 : 2], [5 : 64 : 2]$ and 2) $[5 : 32 : 4], [5 : 64 : 4], [5, 64 : 4]$. In all experiments, we used ReLU as activation for all convolution layers, and a final layer with shape $[1 : 1 : 1]$ and sigmoid activation to classify changes for each time step.

Due to five pooling layers with a stride of 2, the first architecture produces an output which is 32 times less granular in time compared to its input. We define parameter D to be the downsampling ratio of a model. Our two architectures have $D = 32$ and 64, respectively. These architectures were mainly used to measure the sensitivity of our method to the number of layers, and ratio of downsampling.

For DWN, we apply a wavelet with kernels of shape $[5 : d : 2]$ before the convolution layer. Here d is the number of variables of the input time series. The number of levels of the wavelet pyramid was set such that the output of the top level, after applying all convolution and pooling layers is length one. For our synthetic data and architecture above, the levels are 6 and 5 respectively.



(a) (b)

Figure 2: (a) A visualization of the detected changes on the synthetic dataset. Only 3 out of 12 dimensions of a time series were depicted; (b) AUC results for different early/late detection Tolerances for the proposed method and baselines.

Evaluation We evaluate all approaches using precision and recall for detecting change points. Since changes may not exactly match the detected time points, we use a tolerance parameter T for how close a detected change must be to a true change to be considered a match. To go from classification scores to changes we apply non-maximum suppression and a threshold on the output of the network, then match the changes detected to the nearest true change within T timesteps. Precision is defined as (correct change points/all inferred change points) while recall is (correct change points/all true change points). We report area under the curve (AUC) for the resulting precision-recall curves, as a function of the tolerance.

Synthetic Dataset and Results We generated a dataset of 1000 multivariate time series that contains many types of changes to challenge our algorithm. Each time series instance has 12 dimensions with 8192 timepoints, each dimension was a combination of white and red noises. Four change points, randomly distributed in time, were added to each time series. Each change point is a gradual shift in the mean of 4 randomly chosen dimensions, with randomly chosen speed (duration of change) and amount of shift. Thus, each change may have a different impact on the signals. Ground truth is the generated change point time (used for evaluation), and the change probability used for training, which ranges from 0 (stationary) to 1 (mean changing). The simulated time series together with ground truth and detections is shown in Figure 2a.

We train all models on 900 randomly chosen time series and test on the other 100. Figure 2b shows AUC for all algorithms and architectures, with different detection tolerance. Once the tolerance reaches 64, DWN has the highest AUC, reaching 87% at the tolerance=512. The top three approaches were DWN with $D = 32$, CNN with $D = 32$ and NWL with $D = 64$. There was a significant gap between these and CNN with D=64, showing that the wavelets may make our approach less sensitive to this parameter. As shown in Figure 2a, DWN finds the peak of the derivative of each change with high accuracy. When the change is extremely gradual, the detected timepoint may be shifted slightly. However, when changes have such a long duration, for practical applications this shift is likely acceptable and further we note that in the example shown we detect the change slightly before it happens. The false positive at the right of the same figure further we identify when the change starts and ends, though the ground truth only assigned one time to the change.

5 Conclusion

We propose a novel class of deep neural networks that augment a CNN with a trainable wavelet transform. We show that the proposed networks are capable of learning shift and scale invariant transforms, which can be used to detect patterns at different scales. Experiments demonstrate that this architecture can be used for change point detection in multivariate time series even with gradual changes. DWN can potentially be applied to other tasks including visual object recognition, where objects might appear at different distances from the camera.

Acknowledgments

This work was supported in part by the NLM of the NIH under Award Number R01LM011826.

References

[AAB+16] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[AM07] Ryan Prescott Adams and David JC MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

[BF+17] Lawrence Bardwell, Paul Fearnhead, et al. Bayesian detection of abnormal segments in multiple time series. *Bayesian Analysis*, 12(1):193–218, 2017.

[BO16] Ian Barnett and Jukka-Pekka Onnela. Change point detection in correlation networks. *Scientific reports*, 6:18893, 2016.

[CMO16] Rodolfo C Cavalcante, Leandro L Minku, and Adriano LI Oliveira. Fedd: Feature extraction for explicit concept drift detection in time series. In *Neural Networks (IJCNN), 2016 International Joint Conference on Neural Networks*, pages 740–747. IEEE, 2016.

[CTK+17] Jedelyn Cabrieto, Francis Tuerlinckx, Peter Kuppens, Mariel Grassmann, and Eva Ceulemans. Detecting correlation changes in multivariate time series: A comparison of four non-parametric change point detection methods. *Behavior research methods*, 49(3):988–1005, 2017.

[GKVL06] Andrew B Gardner, Abba M Krieger, George Vachtsevanos, and Brian Litt. One-class novelty detection for seizure analysis from intracranial eeg. *Journal of Machine Learning Research*, 7(Jun):1025–1044, 2006.

[GMRS16] Sudipto Guha, Nina Mishra, Gourav Roy, and Okke Schrijvers. Robust random cut forest based anomaly detection on streams. In *International Conference on Machine Learning*, pages 2712–2721, 2016.

[HMEYC14] Maayan Harel, Shie Mannor, Ran El-Yaniv, and Koby Crammer. Concept drift detection through resampling. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1009–1017, 2014.

[HRVB13] Toby Hocking, Guillem Rigaill, Jean-Philippe Vert, and Francis Bach. Learning sparse penalties for change-point detection using max margin interval regression. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 172–180, 2013.

[IPK16] Tsuyoshi Idé, Dzung T Phan, and Jayant Kalagnanam. Change detection using directional statistics. In *IJCAI*, pages 1613–1619, 2016.

[JNIH16] Michael Jones, Daniel Nikovski, Makoto Imamura, and Takahisa Hirata. Exemplar learning for extremely efficient anomaly detection in real-valued time series. *Data Mining and Knowledge Discovery*, 30(6):1427–1454, 2016.

[KGB14] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[LB+95] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[LXDS15] Shuang Li, Yao Xie, Hanjun Dai, and Le Song. M-statistic for kernel change-point detection. In *Advances in Neural Information Processing Systems*, pages 3366–3374, 2015.

[MAL15] George D Montanez, Saeed Amizadeh, and Nikolay Laptev. Inertial hidden markov models: Modeling change in multivariate time series. In *AAAI*, pages 1819–1825, 2015.

[QAWZ15] Abdulhakim A Qahtan, Basma Alharbi, Suojin Wang, and Xiangliang Zhang. A pca-based change detection framework for multidimensional data streams: Change detection in multidimensional data streams. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 935–944. ACM, 2015.

[SWLD13] Yale Song, Zhen Wen, Ching-Yung Lin, and Randall Davis. One-class conditional random fields for sequential anomaly detection. In *IJCAI*, pages 1685–1691, 2013.

[YKNS13] Makoto Yamada, Akisato Kimura, Futoshi Naya, and Hiroshi Sawada. Change-point detection with feature selection in high-dimensional time-series data. In *IJCAI*, pages 1827–1833, 2013.