

Poster Abstract: Distributed Fault Detection using a Recurrent Neural Network

Oliver Obst
CSIRO ICT Centre
Locked Bag 17, North Ryde, NSW 1670, Australia
oliver.obst@csiro.au

ABSTRACT

In long-term deployments of sensor networks, monitoring the quality of gathered data is a critical issue. Over the time of deployment, sensors are exposed to harsh conditions, causing some of them to fail or to deliver less accurate data. If such a degradation remains undetected, the usefulness of a sensor network can be greatly reduced. We present an approach that learns spatio-temporal correlations between different sensors, and makes use of the learned model to detect misbehaving sensors by using distributed computation and only local communication between nodes. We introduce SODESN, a distributed recurrent neural network architecture, and a learning method to train SODESN for fault detection in a distributed scenario. Our approach is evaluated using data from different types of sensors and is able to work well even with less-than-perfect link qualities and more than 50% of failed nodes.

1. INTRODUCTION

Wireless sensor networks (WSN) are increasingly being deployed over extended periods of time [1]. For this work, we are interested in detecting problems in long-term deployments of WSN that manifest in changes of sensor readings for some of the nodes of an entire network as a result of a sensor fault. We present an approach that is able to learn spatio-temporal correlations in sensor readings and make use of them for detecting anomalies in a decentralized way. Instead, sensor nodes participate in a large, distributed recurrent neural network, where each of the sensor nodes hosts only a few neural units and communicates only with its local

neighbor sensor nodes. Our neural network approach is inspired by echo state networks (ESN) [3], a recurrent neural network approach which has shown to be successful in learning even complex time series.

2. SPATIALLY ORGANIZED DISTRIBUTED ECHO STATE NETWORKS

To distribute a recurrent neural network over a WSN, connections between units have to be restricted to the spatial neighborhood of sensor nodes in order to avoid unrestricted global communication. We also would like to retain the efficient learning of ESN. Therefore, we create neural units on each sensor node, and follow the original idea of ESN in that all connections between internal units are randomly initialized and fixed. Connecting units only to spatial neighbors on different devices leads to our idea of spatially organized distributed echo state networks (SODESN), where the underlying communication structure of the sensor network prohibits to use arbitrary synaptic connections.

In a setup with M sensor nodes, each node m hosts K_m input units, N_m hidden units, and L_m output units. The total number of neural units in the network is K , N , and L , respectively. In addition, we use proxy units to store activations from connected neighbor units. This eliminates the need for tight synchronization as long as nodes use the same update interval, and helps to ease the effect of link failures.

2.1 Initializing an untrained SODESN

To set up the untrained SODESN, we create neural units on each sensor node. We construct local internal connection matrices \mathbf{W}_j with a specified density, and scale each of them so that the spectral radius is smaller than one. In addition, we create sparse random connections between internal units on neighbor devices, represented by connection from proxy units for incoming connections, and references to sensor nodes and respective proxy units for outgoing connections. Local input connection matrices \mathbf{W}_j^{in} with random weights fully connect input units to all local internal units on

the node. For output units, we create local random matrices $\mathbf{W}_j^{\text{out}}$ to provide them with input from input units, proxy units and internal units.

3. SODESN: A TRAINING ALGORITHM

The standard training approach for ESN [2] cannot be applied to our architecture, because it assumes that output units can be connected to any of the input or the hidden units. In SODESN, we want to connect output units only to local input, internal or proxy units.

Therefore, training is executed in two steps: as a first step, we sample a matrix \mathbf{M} of internal network states during training, and a matrix \mathbf{T} of output activations. For each time step of the training data, we collect a vector of internal activations and a vector of output activations from our SODESN. The sampled vectors are stored in new rows of \mathbf{M} and \mathbf{T} . With N the total number of hidden units, L the number of output units, and S the number of training steps, the final sizes of \mathbf{M} and \mathbf{T} are $S \times N$ and $S \times L$, respectively.

In a second step, we compute the output weights w_{ij}^{out} to let the training time series $\mathbf{d}(n)$ for each output unit j approximate a linear combination of the internal activations $\mathbf{x}(n)$. ‘‘Approximate’’ means to minimize the mean squared error on the training signal, which, in the case of ESN, can be achieved by multiplying the pseudoinverse of \mathbf{M} with \mathbf{T} : $(\mathbf{W}^{\text{out}})^t = \mathbf{M}^{-1}\mathbf{T}$. In SODESN, however, this operation is not possible, because it will create connections from all internal units to all the output units. A solution to the problem is to adapt the output weights locally, by using local connection matrices \mathbf{M}_j and \mathbf{T}_j for each sensor node j . \mathbf{M}_j contains only activations of local input, internal and proxy units, while \mathbf{T}_j contains output activations of the local output units. For each node, we compute a local output connection matrix as $(\mathbf{W}_j^{\text{out}})^t = \mathbf{M}_j^{-1}\mathbf{T}_j$.

3.1 Distributed fault detection

Using SODESN for fault detection involves making predictions on each sensor node. It requires also to set a threshold for sensor readings to be considered abnormal, for example a deviation exceeding the maximum deviation of predictions on the test set. If the difference between the prediction and the actual reading exceeds the specified threshold, an alarm is raised by the node.

4. EXPERIMENTS

We evaluated our approach in simulations where we used data from a local weather station (Department of Physical Geography of Macquarie University [4]) with several sensors measuring the air temperature, soil temperatures at 1cm, 5cm, 10cm, 20cm, and 50cm respectively, radiation, and infrared. The simulated setup consisted of 8 sensor nodes arranged in a 2 by 4 grid where

each node has one of the sensors and can communicate with its nearest neighbors. The data we used was taken in 15 minute intervals.

We experimentally evaluated the amount of training data needed, the number of units in the neural network, and the robustness against link failures. In addition, we computed a benchmark against standard ESN.

5. RESULTS AND CONCLUSIONS

In our work, we introduce SODESN, a novel distributed recurrent neural network architecture for creating models of dynamical systems. We describe an offline learning approach for SODESN that is closely related to training ESN and inherits the low computational complexity of the original approach. We then presented an approach to train SODESN for fault detection in WSN, where predictions of sensor values are made based on information from neighbor nodes.

Our evaluations on real-world data show that our approach can be used to build models of dynamic time series and help to detect sensor faults. We have shown that the approach is robust to WSN link failures through its distributed computation and local communication. SODESN outperform a comparable, centralized approach assuming realistic link qualities. Using only local communication also contributes to SODESN scaling well with an increasing number of WSN nodes.

We have also shown that our approach is robust against multiple node failures. In our evaluation using the predictions of failed sensors as input, 50% of the sensors failed without affecting prediction quality, and the performance degraded gracefully up to slightly more than 80% failed nodes.

6. REFERENCES

- [1] P. Corke, P. Valencia, P. Sikka, T. Wark, and L. Overs. Long-duration solar-powered wireless sensor networks. In *EmNets '07: Proceedings of the 4th workshop on Embedded networked sensors*, pages 33–37, 2007.
- [2] H. Jaeger. Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the echo state network approach. GMD Report 159, Fraunhofer AIS, 2002.
- [3] H. Jaeger and H. Haas. Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication. *Science*, 304(5667):78–80, 2004.
- [4] Macquarie University, Sydney. Automatic weather station of the Department of Physical Geography. Web page: <http://aws.mq.edu.au/>, 2008. Last checked: 3. Oct. 2008.