Data Quality Network for Spatiotemporal Forecasting

Sungyong Seo, Yan Liu
Department of Computer Science
University of Southern California
{sungyons, yanliu.cs}@usc.edu

Arash Mohegh, George Ban-Weiss
Dept of Civil and Environmental Engineering
University of Southern California
{mohegh,banweiss}@usc.edu

Abstract

The importance of learning from spatiotemporal data has been growing with the increasing number of data sources distributed over space. While numerous studies have been done for analyzing the spatiotemporal signals, existing models have assumed that the heterogeneous data sources in spatial domain are equally reliable over time. In this paper, we propose the novel method that infers the time-varying data quality level based on the local variations of spatiotemporal signals without explicitly assigned labels. Furthermore, we extend the formulation of the quality level by combining with graph convolutional networks to exploit the efficient architecture. Finally, we evaluate the proposed method by simulating a forecasting task with real-world climate data.

1 Introduction

The recent advances with GPS-equipped technology have facilitated the collection of large spatiotemporal datasets. As the increasing number of spatiotemporal data, many have proposed representing this data as time-varying graph signals in various domains, such as sensor networks [8, 14], climate analysis [2, 6], and traffic control systems [5, 10]. Furthermore, the study of time-varying graph signals has been applied to biology [7, 11].

While many works have been proposed to exploit the spatial structures and temporal signals, most models have implicitly assumed that each signal source in a structure is equally reliable over time, and not differentiating data quality of different sources at different time steps. However, a large amount of data are from heterogenous sources that have different levels of noise [9, 12]. Moreover, the noises of each source can be time-varying due to abrupt malfunction of sensors. In order to fully utilize the credibility of each single source over time, it is important to be able to model a different level of data. However, relying on domain experts to manually assign such labels is laborious and expensive, making it extremely challenging to build a model that can differentiate between high and low quality sources in an unsupervised setting.

In this paper, we consider the learning problem of spatiotemporal signals, where temporal signals have been observed at different locations which can be represented by time-varying graph signals. Given a graph and observations, we propose a model that is able to exploit the structural information (spatial features) as well as the temporal behaviors (time series). On top of the time-varying graph signals, we implement graph-based neural networks which encode the level of data quality at each vertex into a scalar by looking at signals observed at the set of all the vertices adjacent to the vertex. Here is a summary of the key contributions of the paper.

- Define data quality level at a vertex in a graph using the local variation of the vertex.
- Combine the quality level with an objective function defined by a given task.
- Demonstrate that the data quality is an essential factor which can improve performance of forecasting.

Workshop on Deep Learning for Physical Sciences (DLPS 2017), NIPS 2017, Long Beach, CA, USA.

2 Preliminaries

Local Variation We focus on the graph signals defined on an undirected, weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, where $\mathcal{V}$ is a set of vertices with $|\mathcal{V}| = N$ and $\mathcal{E}$ is a set of edges. $\mathbf{W} \in \mathbb{R}^{N \times N}$ is a random-walk normalized weighted adjacency matrix which provides how two vertices are relatively close. When the elements W_{ij} are not be expliticly provided by dataset, the graph connectivity can be constructed by various distance metrics, such as Euclidean distance, cosine similarity, and a Gaussian kernel [1], on the vertex features $\mathbf{V} \in \mathbb{R}^{N \times F}$ where F is the dimension of the features.

Once all the structural connectivity is defined, the local variation can be defined by the edge derivative of a given graph signal $\mathbf{x} \in \mathbb{R}^N$ defined on every vertex [13].

$$\left.\frac{\partial \mathbf{x}}{\partial e}\right|_{e=(i,j)} = \sqrt{W_{ij}}(x(j) - x(i)), \tag{1}$$

where $e = (i, j)$ is defined as a direction of the derivative and $x(i)$ is a signal value on the vertex i. The graph gradient of $\mathbf{x}$ at vertex i can be defined by Eq. 1 over all edges joining the vertex i.

$$\nabla_i \mathbf{x} = \left(\left.\frac{\partial \mathbf{x}}{\partial e}\right|_{e=(i,j)} \middle| \, j \in \mathcal{N}_i \right), \tag{2}$$

where $\mathcal{N}_i$ is a set of neighbor vertices of the vertex i. While the dimesion of the graph gradient is different due to the different number of neighbors of each vertex, the local variation at vertex i can be defined by a norm of the graph gradient:

$$\|\nabla_i \mathbf{x}\|_2^2 = \sum_{j \in \mathcal{N}_i} W_{ij}(x(j) - x(i))^2, \tag{3}$$

Eq. 3 provides a measure of local variation of $\mathbf{x}$ at vertex i. As it indicates, if all neighboring signals of i are close to the signal at i, the local variation at i should be small and it means *less fluctuated signals* around the vertex. As Eq. 3 shows, the local variation is a function of a structural connectivity $\mathbf{W}$ and graph signals $\mathbf{x}$.

The concept of the local variation is easily generalized to multivariate graph signals of M different measures by repeatedly computing Eq. 3 over all measures.

$$\mathbf{L}_i = (\|\nabla_i \mathbf{x}^1\|_2^2, \cdots, \|\nabla_i \mathbf{x}^m\|_2^2, \cdots, \|\nabla_i \mathbf{x}^M\|_2^2), \tag{4}$$

where $\mathbf{x}^m \in \mathbb{R}^N$ corresponds the mth signals from multiple sensors. As Eq. 4 indicates, $\mathbf{L}_i$ is a M dimensional vector describing local variations at the vertex i with respect to the M different measures.

Finally, it is desired to represent Eq. 4 in a matrix form to be combined with graph convolutional networks later.

$$\mathbf{L}(\mathbf{W}, \mathbf{X}) = (\mathbf{D} + \mathbf{W})(\mathbf{X} \odot \mathbf{X}) - 2(\mathbf{X} \odot \mathbf{W}\mathbf{X}), \tag{5}$$

where $\mathbf{D}$ is a degree matrix defined as $D_{ii} = \sum_j W_{ij}$ and $\odot$ is an element-wise product operator. $\mathbf{X}$ is a $N \times M$ matrix describing multivariate graph signals on N vertices and $\mathbf{x}^m$ is a mth column of $\mathbf{X}$. $\mathbf{L} \in \mathbb{R}^{N \times M}$ is a local variation matrix and L_{im} is the local variation at the vertex i with respect to the mth signal.

Data Quality Level While the term of data quality has been used in various ways, it generally means "*fitness of use*" for intended purposes [3]. In this section, we will define the term under the property of data we are interested in and propose how to exploit the data quality level into a general framework.

Given a multivariate graph signal $\mathbf{X} \in \mathbb{R}^{N \times M}$ on vertices represented by a feature matrix $\mathbf{V} \in \mathbb{R}^{N \times F}$, we assume that a signal value at a certain vertex i should not be significantly different with signals of neighboring vertices $j \in \mathcal{N}_i$. This is a valid assumption if the value at the vertex i is dependent on (or function of) features of the vertex $\mathbf{V}_i$ when an edge weight is defined by a distance in the feature vector space between two vertices. In other word, if two vertices have similar features, they

are connected to form the graph structure and the signal values observed at the vertices are highly likely similar. There are a lot of domains which follow the assumption, for instance, geographical features (vertex features) and meteorological observations (graph signal) or sensory nervous features and received signals.

Under the assumption, we define the data quality at a vertex i as a function of local variations of i:

$$w_i = q(\mathbf{L}_i), \tag{6}$$

It is flexible to choose the function q. For example, w_i can be defined as an average of $\mathbf{L}_i$. If so, all measures are equally considered to compute the data quality at vertex i. In more general sense, we can introduce parameterized q by $\mathbf{\Phi}$ and learn the parameters through data. [4] propose a method that learns parameters for graph-based features by the layer-wise graph convolutional networks (GCNs) with arbitrary activation functions. For a single layer GCN on a graph, the latent representation of each node can be represented as:

$$\mathbf{Z} = \sigma(\hat{\mathbf{A}}\mathbf{X}\mathbf{\Theta}), \tag{7}$$

where $\hat{\mathbf{A}}$ provides *structural connectivities* of a graph and $\mathbf{\Theta}$ is a trainable parameter matrix. σ is an activation function. By stacking $\sigma(\hat{\mathbf{A}}\mathbf{X}\mathbf{\Theta})$, it is able to achieve larger receptive fields with multi-layer GCNs. See details in [4].

We can replace $\hat{\mathbf{A}}\mathbf{X}$ with $\mathbf{L}$ which is also a function of the weighted adjacency $\mathbf{W}$ and the graph signal $\mathbf{X}$. Note that values in row i of $\hat{\mathbf{A}}\mathbf{X}$ and $\mathbf{L}$ are a function of values at i as well as neighbors of i. Although $\mathbf{L}$ only exploits nearest neighbors (i.e., 1-hop neighbors), it is possible to consider K-hop neighbors to compute the local variations by stacking GCN before applying Eq. 3. The generalized formula for the data quality level can be represented as:

$$\mathbf{Z} = \sigma_K(\hat{\mathbf{A}}\sigma_{K-1}(\hat{\mathbf{A}}\cdots\sigma_1(\hat{\mathbf{A}}\mathbf{X}\mathbf{\Theta_1})\cdots\mathbf{\Theta}_{K-1})\mathbf{\Theta}_K), \tag{8}$$

$$\mathbf{w} = \sigma_L(\mathbf{L}(\mathbf{W},\mathbf{Z})\mathbf{\Phi}). \tag{9}$$

where K is the number of GCN layers and $\mathbf{w} = (w_1, w_2, \cdots, w_N)$ is the data quality level of each vertex incorporating K-hop neighbors. We propose some constraints to ensure that a higher w_i corresponds to *less fluctuated* around i. First, we constrain $\mathbf{\Phi}$ to be positive to guarantee that larger elements in $\mathbf{L}$ cause larger $\mathbf{L}\mathbf{\Phi}$ that are inputs of σ_L. Next, we use an activation function that is inversely proportional to an input, e.g., $\sigma_L(x) = \frac{1}{1+x}$, to meet the relation between the data quality w_i and the local variations $\mathbf{L}_i$. Once $\mathbf{w}$ is obtained, it will be combined with an objective function to assign a penalty for each vertex loss function.

3 Model and Experiments

DQ-LSTM Figure 1 illustrates how data quality networks with LSTM (DQ-LSTM) consists of submodules. Time-varying graph signals on N vertices can be represented as a tensor form, $\mathcal{X} \in \mathbb{R}^{N\times M\times T}$ where the total length of signals is T. First, the time-varying graph signals for each vertex are segmentized to be fed into LSTM. For example, $\mathcal{X}(i,:,h:h+k-1)$ is one of segmentized signals on vertex i. Second, the graph signals for all vertices at the last time stamp $\mathcal{X}(:,:,h+k-1)$ are used as an input of GCNs followed by DQN. Hence, we consider the data quality level by looking at the local variations of the last signals and the estimated quality level w_i is used to assign a weight on the loss function defined on the vertex i. We use the mean squared error loss function.

For each vertex i, DQ-LSTM repeatedly reads inputs, predicts outputs, and updates parameters as many as a number of segmentized length k time series.

$$\mathcal{L}_i = \frac{1}{n_i}\sum_{j=1}^{n_i} w_i \|\hat{\mathcal{X}}(i,:,k+j-1) - \mathcal{X}(i,:,k+j-1)\|_2^2 + \beta\|\mathbf{\Phi}\|_2^2 \tag{10}$$

where n_i is the number of segmentized series on vertex i and $\hat{\mathcal{X}}(i,:,k+j-1)$ is a predicted value from a fully connected layer (FC) which reduces a dimension of an output vector from LSTM. L_2 regularization is used to prevent overfitting. Then, the total loss function over all vertices is $\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_i$

Table 1: Forecasting mean absolute error (MAE)

	LSTM	**GCN-LSTM** (K=1)	**GCN-LSTM** (K=2)	**DQ-LSTM** (K=0)	**DQ-LSTM** (K=1)
WU7	0.5823 (0.0656)	0.5152 (0.0081)	0.5073 (0.0261)	0.5096 (0.0152)	**0.4788** (0.0111)
WU8	0.5911 (0.0221)	0.5356 (0.0398)	0.5151 (0.0272)	0.5087 (0.0117)	**0.4856** (0.0086)
WB7	0.4725 (0.0277)	0.4687 (0.0348)	0.4411 (0.0321)	0.4588 (0.0148)	**0.4108** (0.0129)
WB8	0.5435 (0.0376)	0.5412 (0.0483)	0.5296 (0.0164)	0.4602 (0.0440)	**0.4574** (0.0178)

Dataset In order to have a fair comparison, we use real-world meteorological measurements from two commercial weather service providing real-time weather information, Weather Underground(WU)[1] and WeatherBug(WB)[2]. Both services use observations from automated personal weather stations (PWS). In the dataset, all stations are distributed around Los Angeles County and geographical characteristics where the station is located at are provided. These characteristics would be used as a set of input features $\mathbf{V}$ to build a graph structure. The list of the static 11 characteristics is, *Latitude, Longitude, Elevation, Tree fraction, Vegetation fraction, Albedo, Distance from coast, Impervious fraction, Canopy width, Building height,* and *Canopy direction*. At each station, a number of meteorological signals are observed through the installed instruments. The types of the measurements are *Temperature, Solar radiation, Pressure*, and *Relative humidity*. 2 months observations from each service, July/2015 and August/2015, denoted as 7 and 8, respectively.

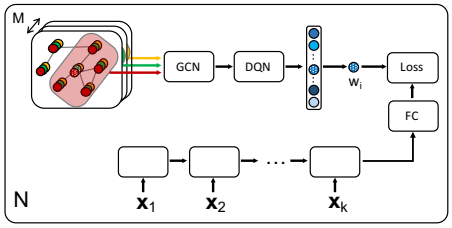

Figure 1: Architecture of DQ-LSTM that consists of GCNs followed by the data quality network DQN and LSTM. GCNs are for localized features from given graph signals (different colors indigate different signals) and DQN computes the data quality of each vertex. Each loss function on each vertex is weighted by the quality level w_i. Note that the dot-patterned circles denote the current vertex.

Forecasting experiment Since all models are dependent on the previous observations, we set the same lag length $k = 10$. Thus, k-steps past observations are fed into them to predict next values. $\beta = 0.05$ is used for the $L2$ regularization of the final layer. For each month dataset, we split it into three subsets, train(60%)/valid(20%)/test(20%) sets. Among given measurements, *Temperature* is used as a target measurement and we repeat 20 times with random initializations.

Experimental results are summarized in Table 1. We report the forecasting mean absolute error (MAE) with standard deviations. Overall, models considering a given graph structure outperform a simple LSTM. Note that the graph structure is not given explicitly but constructed by features of vertices. While the connectivities and weights are dependent on the distance function, it cleary shows that knowing neighbor signals of a given node can help predict next value of the node.

Although GCN is able to transfer a given signal of a vertex to a latent representation that is more compact and expressive, it is difficult to learn a mapping function to the data quality level from neighbor signals directly unlike DQ-LSTM which pre-transfers the signals to local variations explicitly. In other word, given signal X, what GCN learns is $w = f(X)$ where w is the data quality we want to infer from data, however, DQ-LSTM learns $w = g(Y = h(X))$ where Y is a local variation matrix given by X in a closed form, h. Thus, lower MAEs of DQ-LSTM verify that our assumption in Section 2 is valid and the local variations are a useful metric to measure the data quality level. It is also noticeable that DQ-LSTM with a GCN reports the lowest MAE among other models. This is because the additionally trainable parameters in GCN increase the number of nodes to compute better local variations.

[1]https://www.wunderground.com/

[2]http://weather.weatherbug.com/

References

[1] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*, pages 585–591, 2002.

[2] Siheng Chen, Aliaksei Sandryhaila, José MF Moura, and Jelena Kovacevic. Signal denoising on graphs via graph filtering. In *Signal and Information Processing (GlobalSIP), 2014 IEEE Global Conference on*, pages 872–876. IEEE, 2014.

[3] Joseph Juran and A Blanton Godfrey. Quality handbook. *Republished McGraw-Hill*, pages 173–178, 1999.

[4] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[5] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Graph convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

[6] Jonathan Mei and José MF Moura. Signal processing on graphs: Estimating the structure of a graph. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5495–5499. IEEE, 2015.

[7] Ali Yener Mutlu, Edward Bernat, and Selin Aviyente. A signal-processing-based approach to time-varying graph analysis for dynamic brain network identification. *Computational and mathematical methods in medicine*, 2012, 2012.

[8] Xuesong Shi, Hui Feng, Muyuan Zhai, Tao Yang, and Bo Hu. Infinite impulse response graph filters in wireless sensor networks. *IEEE Signal Processing Letters*, 22(8):1113–1117, 2015.

[9] Shuang Song, Kamalika Chaudhuri, and Anand Sarwate. Learning from data with heterogeneous noise using sgd. In *Artificial Intelligence and Statistics*, pages 894–902, 2015.

[10] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional neural network: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

[11] Qingbao Yu, Erik B Erhardt, Jing Sui, Yuhui Du, Hao He, Devon Hjelm, Mustafa S Cetin, Srinivas Rachakonda, Robyn L Miller, Godfrey Pearlson, et al. Assessing dynamic brain graphs of time-varying connectivity in fmri data: application to healthy controls and patients with schizophrenia. *Neuroimage*, 107:345–355, 2015.

[12] Chicheng Zhang and Kamalika Chaudhuri. Active learning from weak and strong labelers. In *Advances in Neural Information Processing Systems*, pages 703–711, 2015.

[13] Dengyong Zhou and Bernhard Schölkopf. A regularization framework for learning from graph data. In *ICML workshop on statistical relational learning and Its connections to other fields*, volume 15, pages 67–68, 2004.

[14] Xiaofan Zhu and Michael Rabbat. Graph spectral compressed sensing for sensor networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 2865–2868. IEEE, 2012.