STRUCTURED SEQUENCE MODELING WITH GRAPH CONVOLUTIONAL RECURRENT NETWORKS

Youngjoo Seo
EPFL, Switzerland
youngjoo.seo@epfl.ch

Michaël Defferrard
EPFL, Switzerland
michael.defferrard@epfl.ch

Pierre Vandergheynst
EPFL, Switzerland
pierre.vandergheynst@epfl.ch

Xavier Bresson
NTU, Singapore
xavier.bresson@ntu.edu.sg

ABSTRACT

This paper introduces Graph Convolutional Recurrent Network (GCRN), a deep learning model able to predict structured sequences of data. GCRN is a generalization of classical recurrent neural networks (RNN) to data structured by any arbitrary graph. Such structured sequences can be series of frames in videos, spatio-temporal measurements on a network of sensors, or random walks on a vocabulary graph for natural language modeling. The proposed model combines convolutional neural networks (CNN) on graphs to identify spatial structures and RNN to find dynamic patterns. We study two possible architectures of GCRN, and apply the models to two practical problems: predicting moving MNIST data, and modeling natural language with the Penn Treebank dataset [TO CHECK!!!]. Experiments show that exploiting simultaneously graph spatial and dynamic information about data can improve both precision and learning speed.

1 INTRODUCTION

Many real-world data can be cast as structured sequences, with spatio-temporal sequences being a special case. A well-studied example of spatio-temporal data are videos, where succeeding frames share temporal and spatial structures. Many works, such as Donahue et al. (2015); Karpathy & Fei-Fei (2015); Vinyals et al. (2015), leveraged a combination of CNN and RNN to exploit such spatial and temporal regularities. Their models are able to process possibly time-varying visual inputs for variable-length prediction. These neural network architectures consist of combining a CNN for visual feature extraction followed by a RNN for sequence learning. Such architectures have been successful used for video activity recognition, image captioning and video description.

More recently, interest has grown in properly fusing the CNN and RNN models for spatio-temporal sequence modeling. Inspired from language modeling, Ranzato et al. (2014) proposed a model to represent complex deformations and motion patterns by discovering both spatial and temporal correlations. They showed that prediction of the next video frame and interpolation of intermediate frames can be achieved by building a RNN-based language model on the visual words obtained by quantizing the image patches. Their highest-performing model, recursive CNN (rCNN), uses convolutions for both inputs and states. Shi et al. (2015) then proposed the convolutional LSTM network (convLSTM), a recurrent model for spatio-temporal sequence modeling which uses 2D-grid convolution to leverage the spatial correlations in input data. They successfully applied their model to the prediction of the evolution of radar echo maps for precipitation nowcasting.

The spatial structure of many important problems may however not be as simple as regular grids. For instance, the data measured from meteorological stations lie on a irregular grid, i.e. a network of heterogeneous spatial distribution of stations. More challenging, the spatial structure of data may not even be spatial, as it is the case for social or biological networks. Eventually, the interpretation that sentences can be regarded as random walks on vocabulary graphs, a view popularized by Mikolov et al. (2013), allows us to cast language analysis problems as graph-structured sequence models.

This work leverages on the recent models of Defferrard et al. (2016); Ranzato et al. (2014); Shi et al. (2015) to design the GCRN model for modeling and predicting time-varying graph-based data. The core idea is to merge CNN for graph-structured data and RNN to identify simultaneously meaningful spatial structures and dynamic patterns. A generic illustration of the proposed GCRN architecture is given by Figure 1.

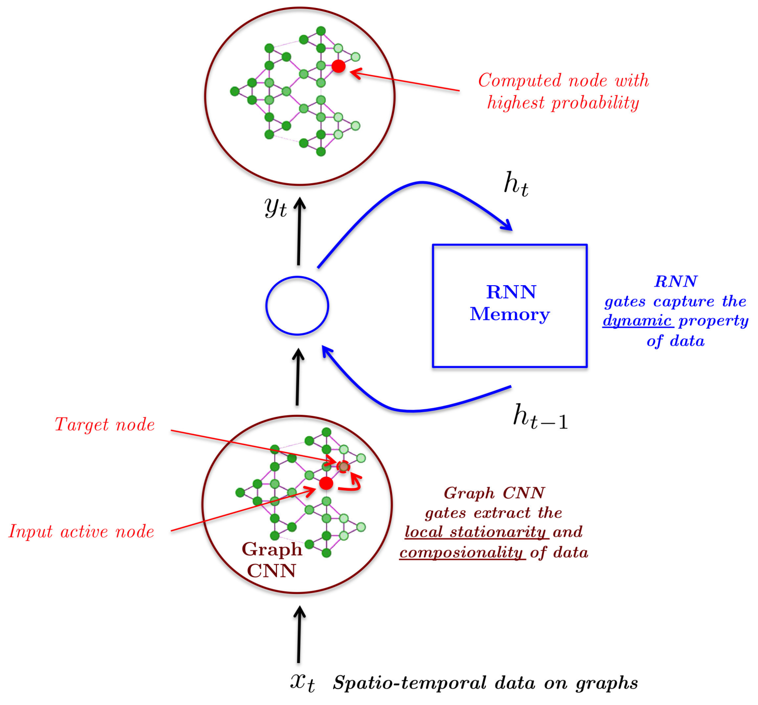
Computed node with highest probability
h_t
y_t
RNN Memory
RNN gates capture the dynamic property of data
h_{t-1}
Target node
Input active node
Graph CNN
Graph CNN gates extract the local stationarity and composionality of data
x_t Spatio-temporal data on graphs

Figure 1: Illustration of the proposed model for spatio-temporal prediction of graph-structured data. The technique combines at the same time CNN on graphs and RNN. RNN can be easily exchanged with LSTM or GRU networks.

2 PRELIMINARIES

2.1 STRUCTURED SEQUENCE MODELING

Sequence modeling is the problem of predicting the most likely future length-K sequence given the previous J observations:

$$\hat{x}_{t+1}, \ldots, \hat{x}_{t+K} = \underset{x_{t+1},\ldots,x_{t+K}}{\arg\max} \; P(x_{t+1}, \ldots, x_{t+K} | x_{t-J+1}, \ldots, x_t), \tag{1}$$

where $x_t \in \mathbf{D}$ is an observation at time t and $\mathbf{D}$ denotes the domain of the observed features. The archetypal application being the n-gram language model (with $n = J + 1$), where $P(x_{t+1}|x_{t-J+1}, \ldots, x_t)$ models the probability of word x_{t+1} to appear conditioned on the past J words in the sentence (Graves, 2013).

In this paper, we are interested in special structured sequences, i.e. sequences where features of the observations x_t are not independent but linked by pairwise relationships. Such relationships are universally modeled by weighted graphs.

Data x_t can be viewed as a graph signal, i.e. a signal defined on an undirected and weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$, where $\mathcal{V}$ is a finite set of $|\mathcal{V}| = n$ vertices, $\mathcal{E}$ is a set of edges and $A \in \mathbb{R}^{n\times n}$ is a weighted adjacency matrix encoding the connection weight between two vertices. A signal $x_t : \mathcal{V} \rightarrow \mathbb{R}^{d_x}$ defined on the nodes of the graph may be regarded as a matrix $x_t \in \mathbb{R}^{n\times d_x}$ whose column i is the d_x-dimensional value of x_t at the i^{th} node. While the number of free variables in a structured sequence of length K is in principle $\mathcal{O}(n^K {d_x}^K)$, we seek to exploit the structure of the space of possible predictions to reduce the dimensionality and hence make those problems more tractable.

2.2 Long Short-Term Memory

A special class of recurrent neural networks (RNN) that prevents the gradient from vanishing too quickly is the popular long short-term memory (LSTM) introduced by Hochreiter & Schmidhuber (1997). This architecture has proven stable and powerful for modeling long-range dependencies in various general-purpose sequence modeling tasks (Graves, 2013; Srivastava et al., 2015; Sutskever et al., 2014). A fully-connected LSTM (FC-LSTM) may be seen as a multivariate version of LSTM where the input $x_t \in \mathbb{R}^{d_x}$, cell output $h_t \in [-1,1]^{d_h}$ and states $c_t \in \mathbb{R}^{d_h}$ are all vectors. In this paper, we follow the FC-LSTM formulation of Graves (2013), that is:

$$\begin{aligned}
i &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + w_{ci} \odot c_{t-1} + b_i), \\
f &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + w_{cf} \odot c_{t-1} + b_f), \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c), \\
o &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + w_{co} \odot c_t + b_o), \\
h_t &= o \odot \tanh(c_t),
\end{aligned} \tag{2}$$

where $\odot$ denotes the Hadamard product, $\sigma(\cdot)$ the sigmoid function $\sigma(x) = 1/(1+e^{-x})$ and $i, f, o \in [0,1]^{d_h}$ are the input, forget and output gates. The weights $W_{x\cdot} \in \mathbb{R}^{d_h \times d_x}$, $W_{h\cdot} \in \mathbb{R}^{d_h \times d_h}$, $w_{c\cdot} \in \mathbb{R}^{d_h}$ and biases $b_i, b_f, b_c, b_o \in R^{d_h}$ are the model parameters.[1] Such a model is called fully-connected because the dense matrices W linearly combine all the components of x and h. The optional peephole connections $w_{c\cdot} \odot c_t$, introduced by Gers & Schmidhuber (2000), have been found to improve performance on certain tasks.

2.3 Convolutional Neural Networks on Graphs

Generalizing convolutional neural networks (CNNs) to arbitrary graphs is a recent area of interest. Two approaches have been explored in the literature: (i) a generalization of the spatial definition of a convolution (Masci et al., 2015; Niepert et al., 2016) and (ii), a multiplication in the graph Fourier domain by the way of the convolution theorem (Bruna et al., 2014; Defferrard et al., 2016). Masci et al. (2015) introduced a spatial generalization of CNNs to 3D meshes. The authors used geodesic polar coordinates to define convolution operations on mesh patches, and formulated a deep learning architecture which allows comparison across different meshes. Hence, this method is tailored to manifolds and is not directly generalizable to arbitrary graphs. Niepert et al. (2016) proposed a spatial approach which may be decomposed in three steps: (i) select a node, (ii) construct its neighborhood and (iii) normalize the selected sub-graph, i.e. order the neighboring nodes. The extracted patches are then fed into a conventional 1D Euclidean CNN. As graphs generally do not possess a natural ordering (temporal, spatial or otherwise), a labeling procedure should be used to impose it. Bruna et al. (2014) were the first to introduce the spectral framework described below in the context of graph CNNs. The major drawback of this method is its $\mathcal{O}(n^2)$ complexity, which was overcome with the technique of Defferrard et al. (2016), which offers a linear complexity $\mathcal{O}(|\mathcal{E}|)$ and provides strictly localized filters. Kipf & Welling (2016) took a first-order approximation of the spectral filters proposed by Defferrard et al. (2016) and successfully used it for semi-supervised classification of nodes. While we focus on the framework introduced by Defferrard et al. (2016), the proposed model is agnostic to the choice of the graph convolution operator $*_\mathcal{G}$.

As it is difficult to express a meaningful translation operator in the vertex domain (Bruna et al., 2014; Niepert et al., 2016), Defferrard et al. (2016) chose a spectral formulation for the convolution operator on graph $*_\mathcal{G}$. By this definition, a graph signal $x \in \mathbb{R}^{n \times d_x}$ is filtered by a non-parametric kernel $g_\theta(\Lambda) = \operatorname{diag}(\theta)$, where $\theta \in \mathbb{R}^n$ is a vector of Fourier coefficients, as

$$y = \theta *_\mathcal{G} x = g_\theta(L)x = g_\theta(U\Lambda U^T)x = Ug_\theta(\Lambda)U^T x \in \mathbb{R}^{n \times d_x}, \tag{3}$$

where $U \in \mathbb{R}^{n \times n}$ is the matrix of eigenvectors and $\Lambda \in \mathbb{R}^{n \times n}$ the diagonal matrix of eigenvalues of the normalized graph Laplacian $L = I_n - D^{-1/2}AD^{-1/2} = U\Lambda U^T \in \mathbb{R}^{n \times n}$, where I_n is the identity matrix and $D \in \mathbb{R}^{n \times n}$ is the diagonal degree matrix with $D_{ii} = \sum_j A_{ij}$ (Chung,

[1]A practical trick is to initialize the biases b_i, b_f and b_o to one such that the gates are initially open.

1997). Note that the signal x is filtered by g_θ with an element-wise multiplication of its graph Fourier transform $U^T x$ with θ (Shuman et al., 2013). Evaluating (3) is however expensive, as the multiplication with U is $\mathcal{O}(n^2)$. Furthermore, computing the eigendecomposition of L might be prohibitively expensive for large graphs. To circumvent this problem, Defferrard et al. (2016) parametrizes g_θ as a truncated expansion, up to order $K-1$, of Chebyshev polynomials T_k such that

$$g_\theta(\Lambda) = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\Lambda}), \tag{4}$$

where the parameter $\theta \in \mathbb{R}^K$ is a vector of Chebyshev coefficients and $T_k(\tilde{\Lambda}) \in \mathbb{R}^{n\times n}$ is the Chebyshev polynomial of order k evaluated at $\tilde{\Lambda} = 2\Lambda/\lambda_{max} - I_n$. The graph filtering operation can then be written as

$$y = \theta *_{\mathcal{G}} x = g_\theta(L)x = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L})x, \tag{5}$$

where $T_k(\tilde{L}) \in \mathbb{R}^{n\times n}$ is the Chebyshev polynomial of order k evaluated at the scaled Laplacian $\tilde{L} = 2L/\lambda_{max} - I_n$. Using the stable recurrence relation $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$ with $T_0 = 1$ and $T_1 = x$, one can evaluate (5) in $\mathcal{O}(K|\mathcal{E}|)$ operations, i.e. linearly with the number of edges. Note that as the filtering operation (5) is an order K polynomial of the Laplacian, it is K-localized and depends only on nodes that are at maximum K hops away from the central node, the K-neighborhood. is referred to Defferrard et al. (2016) for details and an in-depth discussion.

3 Related Works

Shi et al. (2015) introduced a model for regular grid-structured sequences, which can be seen as a special case of the proposed model where the graph is an image grid where the nodes are well ordered. Their model is essentially the classical FC-LSTM (2) where the multiplications by dense matrices W have been replaced by convolutions with kernels W:

$$\begin{aligned}
i &= \sigma(W_{xi} * x_t + W_{hi} * h_{t-1} + w_{ci} \odot c_{t-1} + b_i),\\
f &= \sigma(W_{xf} * x_t + W_{hf} * h_{t-1} + w_{cf} \odot c_{t-1} + b_f),\\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc} * x_t + W_{hc} * h_{t-1} + b_c),\\
o &= \sigma(W_{xo} * x_t + W_{ho} * h_{t-1} + w_{co} \odot c_t + b_o),\\
h_t &= o \odot \tanh(c_t),
\end{aligned} \tag{6}$$

where $*$ denotes the 2D convolution by a set of kernels. In their setting, the input tensor $x_t \in \mathbb{R}^{n_r \times n_c \times d_x}$ is the observation of d_x measurements at time t of a dynamical system over a spatial region represented by a grid of n_r rows and n_c columns. The model holds spatially distributed hidden and cell states of size d_h given by the tensors $c_t, h_t \in \mathbb{R}^{n_r \times n_c \times d_h}$. The size m of the convolutional kernels $W_{h\cdot} \in \mathbb{R}^{m\times m\times d_h \times d_h}$ and $W_{x\cdot} \in \mathbb{R}^{m\times m\times d_h\times d_x}$ determines the number of parameters, which is independent of the grid size $n_r \times n_c$. Earlier, Ranzato et al. (2014) proposed a similar RNN variation which uses convolutional layers instead of fully connected layers. The hidden state at time t is given by

$$h_t = \tanh(\sigma(W_{x2} * \sigma(W_{x1} * x_t)) + \sigma(W_h * h_{t-1})), \tag{7}$$

where the convolutional kernels $W_h \in \mathbb{R}^{d_h \times d_h}$ are restricted to filters of size 1x1 (effectively a fully connected layer shared across all spatial locations).

Observing that natural language exhibits syntactic properties that naturally combine words into phrases, Tai et al. (2015) proposed a model for tree-structured topologies, where each LSTM has access to the states of its children. They obtained state-of-the-art results on semantic relatedness and sentiment classification. Liang et al. (2016) followed up and proposed a variant on graphs. Their sophisticated network architecture obtained state-of-the-art results for semantic object parsing on four datasets. In those models, the states are gathered from the neighborhood by the way of a weighted sum with trainable weight matrices. Those weights are however not shared across the graph, which would otherwise have required some ordering of the nodes, alike any other spatial definition of graph convolution. Moreover, their formulations are limited to the one-neighborhood of the current node, with equal weight given to each neighbor.

Motivated by spatio-temporal problems like modeling human motion and object interactions, Jain et al. (2016) developed a method to cast a spatio-temporal graph as a rich RNN mixture which essentially associates a RNN to each node and edge. Again, the communication is limited to directly connected nodes and edges.

4 Proposed GCRN Models

We propose two natural architectures for GCRN that are quite natural, and investigate their performances on real-world applications in Section 5.

As a first model, we consider a natural end-to-end learning system, meaning that we stack a graph CNN, defined as (5), for feature extraction and a LSTM, defined as (2), for sequence learning:

$$\begin{aligned} x_t^{\text{CNN}} &= \text{CNN}_{\mathcal{G}}(x_t) \\ i &= \sigma(W_{xi}x_t^{\text{CNN}} + W_{hi}h_{t-1} + w_{ci} \odot c_{t-1} + b_i), \\ f &= \sigma(W_{xf}x_t^{\text{CNN}} + W_{hf}h_{t-1} + w_{cf} \odot c_{t-1} + b_f), \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t^{\text{CNN}} + W_{hc}h_{t-1} + b_c), \\ o &= \sigma(W_{xo}x_t^{\text{CNN}} + W_{ho}h_{t-1} + w_{co} \odot c_t + b_o), \\ h_t &= o \odot \tanh(c_t). \end{aligned} \tag{8}$$

In that setting, the input matrix $x_t \in \mathbb{R}^{n \times d_x}$ may represent the observation of d_x measurements at time t of a dynamical system over a network whose organization is given by a graph $\mathcal{G}$. x_t^{CNN} is the output of the graph CNN gate. For a proof of concept, we simply choose here $x_t^{\text{CNN}} = W^{\text{CNN}} *_{\mathcal{G}} x_t$, where $W^{\text{CNN}} \in \mathbb{R}^{K \times d_x \times d_x}$ are the Chebyshev coefficients for the graph convolutional kernels of support K. The model also holds spatially distributed hidden and cell states of size d_h given by the matrices $c_t, h_t \in \mathbb{R}^{n \times d_h}$. Peepholes are controlled by $w_{c\cdot} \in \mathbb{R}^{n \times d_h}$. The weights $W_{h\cdot} \in \mathbb{R}^{d_h \times d_h}$ and $W_{x\cdot} \in \mathbb{R}^{d_h \times d_x}$ are the parameters of the fully connected layers. Such architecture (8) may be enough to capture the data distribution by exploiting local stationarity and compositionality properties as well as the dynamic properties.

As a second model, we generalize the convLSTM model (6) to graphs by simply replacing the Euclidean 2D convolution $*$ by the graph convolution $*_{\mathcal{G}}$:

$$\begin{aligned} i &= \sigma(W_{xi} *_{\mathcal{G}} x_t + W_{hi} *_{\mathcal{G}} h_{t-1} + w_{ci} \odot c_{t-1} + b_i), \\ f &= \sigma(W_{xf} *_{\mathcal{G}} x_t + W_{hf} *_{\mathcal{G}} h_{t-1} + w_{cf} \odot c_{t-1} + b_f), \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc} *_{\mathcal{G}} x_t + W_{hc} *_{\mathcal{G}} h_{t-1} + b_c), \\ o &= \sigma(W_{xo} *_{\mathcal{G}} x_t + W_{ho} *_{\mathcal{G}} h_{t-1} + w_{co} \odot c_t + b_o), \\ h_t &= o \odot \tanh(c_t). \end{aligned} \tag{9}$$

In that setting, the support K of the graph convolutional kernels defined by the Chebyshev coefficients $W_{h\cdot} \in \mathbb{R}^{K \times d_h \times d_h}$ and $W_{x\cdot} \in \mathbb{R}^{K \times d_h \times d_x}$ determines the number of parameters, which is independent of the number of nodes n. In a distributed computing setting, K controls the communication overhead, i.e. the number of nodes a node i should exchange with in order to compute its local states.

The proposed blend of RNNs and graph CNNs is not limited to LSTMs and is straightforward to apply to any kind of recursive networks. For example, a vanilla RNN $h_t = \tanh(W_x x + W_h h)$ would be modified as

$$h_t = \tanh(W_x *_{\mathcal{G}} x_t + W_h *_{\mathcal{G}} h_{t-1}), \tag{10}$$

and a Gated Recurrent Unit (GRU) (Cho et al., 2014) as

$$\begin{aligned} z &= \sigma(W_{xz} *_{\mathcal{G}} x_t + W_{hz} *_{\mathcal{G}} h_{t-1}), \\ r &= \sigma(W_{xr} *_{\mathcal{G}} x_t + W_{hr} *_{\mathcal{G}} h_{t-1}), \\ \tilde{h} &= \tanh(W_{xh} *_{\mathcal{G}} x_t + W_{hh} *_{\mathcal{G}} (r \odot h_{t-1})), \\ h_t &= z \odot h_{t-1} + (1 - z) \odot \tilde{h}. \end{aligned} \tag{11}$$

As demonstrated by Shi et al. (2015), structure-aware LSTM cells can be stacked and used as sequence-to-sequence models using an architecture composed of an encoder, which processes the input sequence, and a decoder, which generates an output sequence. A standard practice for machine translation using RNNs (Cho et al., 2014; Sutskever et al., 2014).

5 EXPERIMENTS

5.1 LANGUAGE MODELING ON PENN TREEBANK

The Penn Treebank dataset has 1,036,580 words, which is already split[2] into a training set of 887,521 words, a validation set of 70,390 words, and a test set of 78,669 words. The size of the vocabulary is 10,000.

[2]https://github.com/wojzaremba/lstm

Proposing Graph Convolutional Recurrent Neural Networks(GCRNN) applies two benchmark datasets: moving MNISTSrivastava et al. (2015) and Penn Tree BankMarcus et al. (1993). Experiments are mainly focuses on how the graph convolution give effects on RNN model by comparing conventional RNN models with GCRNN. In moving MNIST experiments, we shows 2D grid structure is one of special case of graph structure which GCRNN can easily fit to the model as convolutional RNN, while Language modeling on Penn Tree Bank demonstrates graph structure on words helps

Cost function is the cross-entropy defined as

5.2 Moving MNIST

empirical proof that graph RNN works on grid structured data

Following the experimental setup of Shi et al. (2015).

Our first experiment on the moving MNIST dataset Srivastava et al. (2015) shows the ability of our model (9) to learn spatio-temporal structures.

moving MNIST: similar as (6) convLSTM because of lack of orientation / node ordering

rotating MNIST: better thanks to that property

5.3 Meteorological Prediction

network of sensors (with local processing): model aggregates information from local neighborhood, good for distributed processing.

5.4 Language Modeling on Penn Treebank

Uses (10) followed by a softmax layer.

6 Conclusion and Future Work

We introduced the Graph Convolutional Recurrent Network, an architecture designed to model graph-structured and time-varying data. The model is an extension of Shi et al. (2015) which leverages recent advances in graph ConvNets (Defferrard et al., 2016). Numerical experiments have shown the capability of the model to ... We realized during this work that the evaluation of such generative models is hard as we don't have a proper cost function to evaluate the generated samples. To circumvent it, many researchers have moved to the generative adversarial networks (GANs) framework, a scheme introduced by Goodfellow et al. (2014) for training generative models where the goal is to fool a discriminator system that tries to separate data from the true distribution from data generated by the model. Such models are however difficult to train. Future works will both refine the model with advances in graph ConvNets, a recent area of interest, and apply the model to real-world problems.

References

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral Networks and Locally Connected Networks on Graphs. In *International Conference on Learning Representations (ICML)*, 2014.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv:1406.1078*, 2014.

F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Felix A Gers and Jürgen Schmidhuber. Recurrent nets that time and count. In *IEEE-INNS-ENNS International Joint Conference on Neural Networks*, 2000.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

Alex Graves. Generating sequences with recurrent neural networks. *arXiv:1308.0850*, 2013.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

Ashesh Jain, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-RNN: Deep Learning on Spatio-Temporal Graphs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv:1609.02907*, 2016.

Xiaodan Liang, Xiaohui Shen, Jiashi Feng, Liang Lin, and Shuicheng Yan. Semantic object parsing with graph lstm. *arXiv:1603.07063*, 2016.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 1993.

Jonathan Masci, Davide Boscaini, Michael M. Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *IEEE International Conference on Computer Vision (ICCV) Workshops*, 2015.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Estimation of Word Representations in Vector Space. In *International Conference on Learning Representations (ICLR)*, 2013.

Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning Convolutional Neural Networks for Graphs. In *International Conference on Machine Learning (ICML)*, 2016.

MarcAurelio Ranzato, Arthur Szlam, Joan Bruna, Michael Mathieu, Ronan Collobert, and Sumit Chopra. Video (language) modeling: a baseline for generative models of natural videos. *arXiv:1412.6604*, 2014.

Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

D. Shuman, S. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and other Irregular Domains. *IEEE Signal Processing Magazine*, 2013.

Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International Conference on Machine Learning (ICML)*, 2015.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems (NIPS)*, 2014.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Association for Computational Linguistics (ACL)*, 2015.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.