

---

# Deep Networks on Graph-Structured Data

---

000  
001  
002  
003  
004  
005  
006  
007  
008  
009  
010  
011  
012  
013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046  
047  
048  
049  
050  
051  
052  
053  
**Anonymous Author(s)**

Affiliation  
Address  
email

## Abstract

### 1 Introduction

In recent times, deep learning models have proven extremely successful in a wide variety of tasks, from computer vision and acoustic modeling, to natural language processing [8]. At the core of their success lies an important assumption on the statistical properties of the data, namely the *stationarity* and the *compositionality* through local statistics, which is present in natural images, video, or speech. These properties are exploited efficiently by ConvNets [7, 6], which are designed to extract local features that are shared across the signal domain. Thanks to this, they are able to greatly reduce the number of parameters in the network with respect to generic Deep architectures, without sacrificing the capacity to extract informative statistics from the data. Similarly, Recurrent Neural Nets (RNNs) trained on temporal data implicitly assume a stationary distribution.

One can think of such data examples as being signals defined on a low-dimensional grid. In that case, stationarity is well defined via the natural translation operator on the grid, locality is defined via the metric of the grid, and compositionality is obtained from downsampling, or equivalently thanks to the multi-resolution property of the grid. However, there exist many examples of data that lack the underlying low-dimensional grid structure. For example, text documents represented as bags of words can be thought of as signals defined on a graph whose nodes are vocabulary terms and whose weights represent some similarity measure between terms, such as co-occurrence statistics. In medicine, a patient's gene expression data can be viewed as a signal defined on the graph imposed by the regulatory network. In fact, computer vision and audio, which are the main focus of research efforts in deep learning, only represent a special case of data defined on an extremely simple low-dimensional graph. These complex graphs might be of higher dimension, and the statistical properties of data defined on such graphs might not satisfy the stationarity, locality and compositionality assumptions previously described. For those type of data of dimension  $N$ , deep learning strategies are reduced to learning with fully-connected layers, which have  $O(N^2)$  parameters, and regularization is carried out via weight decay and dropout [14].

When the graph structure of the input is known, [2] introduced a model to generalize ConvNets using low learning complexity, similar to what a convnet uses, that was demonstrated on simple low-dimensional graphs. In this work, we are interested in generalizing ConvNets to high-dimensional, general datasets, and, most importantly, to the setting where the graph structure is not known a priori. In this context, learning the graph structure amounts to estimating the similarity matrix, which has complexity  $O(N^2)$ . One may therefore wonder whether the graph estimation followed by graph convolutions offers advantages with respect to learning directly from the data with fully connected layers. We attempt to answer this question experimentally and to set-up baselines for future reference.

We explore these approaches in two areas of application for which it has not been possible to apply convolutional networks before: text categorization and bioinformatics. Our results show that

054 our method is capable of matching or outperforming large, fully-connected networks trained with  
055 dropout using fewer parameters. Our main contributions can be summarized as follows:  
056

- 057 • We extend the ideas from [2] to large scale classification problems, on Object Recognition,  
058 text categorization and bioinformatics.
- 059 • We consider the most general setting where no prior information on the graph structure is  
060 available, and propose unsupervised and supervised graph estimation strategies.
- 061 • Finally, we introduce an alternative formulation for efficient learning in graph-structured  
062 domains which works directly in the feature domain.  
063

064 The rest of the paper is structured as follows. Section 2 reviews similar works in the literature. Sec-  
065 tion 3 discusses generalizations of convolutions on graphs, and Section 4 addresses the question of  
066 graph estimation. Finally, Section ?? shows numerical experiments on large scale object recogniton,  
067 text categorization and bioinformatics.

## 069 2 Related Work 070

071 There have been several works which have explored architectures using the so-called local receptive  
072 fields [5, 4, 11], mostly with applications to image recognition. In particular, [4] proposes a scheme  
073 to learn how to group together features based upon a measure of similarity that is obtained in an  
074 unsupervised fashion. However, it does not attempt to exploit any weight-sharing strategy.

075 Recently, [2] proposed a generalization of convolutions to graphs via the Graph Laplacian. By  
076 identifying a linear, translation invariant operator in the grid (the Laplacian operator), with its coun-  
077 terpart in a general graph (the Graph Laplacian), one can view convolutions as the family of linear  
078 transforms commuting with the Laplacian. By combining this commutation property with a rule  
079 to find localized filters, the model requires only  $O(1)$  parameters per “feature map”. However,  
080 this construction requires prior knowledge of the graph structure, and was shown only on simple,  
081 low-dimensional graphs. More recently, [10] introduced Shapenet, another generalization of Con-  
082 volutions on non-Euclidean domains, based on geodesic polar coordinates, which was successfully  
083 applied to shape analysis, and allows comparison across different manifolds. However, it also re-  
084 quires prior knowledge of the manifolds.

085 The graph or similarity estimation aspects have also been extensively studied in the past. For in-  
086 stance, [12] studies the estimation of the graph from a statistical point of view, through the identi-  
087 fication of a certain graphical model using  $\ell_1$  penalized logistic regression. Also, [3] considers the  
088 problem of learning a deep architecture through a series of Haar contractions, which are learnt using  
089 an unsupervised pairing criteria over the features.

## 091 3 Generalizing Convolutions in Graphs 092

### 093 3.1 Spectral Networks 094

095 Our work builds upon [2] which introduced spectral networks. We recall the definition here and its  
096 main properties.

097 A spectral network generalizes a convolutional network through the Graph Fourier Transform, which  
098 is in turn defined via a generalization of the Laplacian operator on the grid to the graph Laplacian.  
099 An input vector  $x \in \mathbb{R}^N$  is seen as a signal defined on a graph  $G$  with  $N$  nodes.

100 **Definition 1.** Let  $W$  be a  $N \times N$  similarity matrix representing an undirected graph  $G$ , and let  
101  $L = D - W$  be its graph Laplacian with  $D = W \cdot \mathbf{1}$ , with eigenvectors  $U = (u_1, \dots, u_N)$ . Then  
102 a graph convolution of input signals  $x$  with filters  $g$  on  $G$  is defined by  $x *_G g = U^T (Ux \odot Ug)$ ,  
103 where  $\odot$  represents a point-wise product.

104 Here, the unitary matrix  $U$  plays the role of the Fourier Transform in  $\mathbb{R}^d$ . There are several ways  
105 of computing the graph Laplacian  $L$  [1]. In this paper, we choose the normalized version  $L =$   
106  $I - D^{-1/2}WD^{-1/2}$ , where  $D$  is a diagonal matrix with entries  $D_{ii} = \sum_j W_{ij}$ . Note that in the case  
107 where  $W$  represents the lattice, from the definition of  $L$  we recover the discrete Laplacian operator

108  $\Delta$ . Also note that the Laplacian commutes with the translation operator, which is diagonalized in  
 109 the Fourier basis. It follows that the eigenvectors of  $\Delta$  are given by the Discrete Fourier Transform  
 110 (DFT) matrix. We then recover a classical convolution operator that noting that convolutions are by  
 111 definition linear operators that diagonalize in the Fourier domain (also known as the Convolution  
 112 Theorem [9]).

113 Learning filters in a Graph thus amounts to learning spectral multipliers  $w_g = (w_1, \dots, w_N)$   
 114

$$115 \quad x *_G g := U^T(\text{diag}(w_g)Ux) . \\ 116$$

117 Extending the convolution to inputs  $x$  with multiple input channels is straightforward. If  $x$  is a signal  
 118 with  $M$  input channels and  $N$  locations, we apply the transformation  $U$  on each channel, and then  
 119 use multipliers  $w_g = (w_{i,j} ; i \leq N, j \leq M)$ .

120 However, for each feature map  $g$  we need convolutional kernels are typically restricted to have small  
 121 spatial support, independent of the number of input pixels  $N$ , which enables the model to learn a  
 122 number of parameters independent of  $N$ . In order to recover a similar learning complexity in the  
 123 spectral domain, it is thus necessary to restrict the class of spectral multipliers to those corresponding  
 124 to localized filters.

125 For that purpose, we seek to express spatial localization of filters in terms of their spectral multipli-  
 126 ers. In the grid, smoothness in the frequency domain corresponds to the spatial decay, since  
 127

$$128 \quad \left| \frac{\partial^k \hat{x}(\xi)}{\partial \xi^k} \right| \leq C \int u^k |x(u)| du , \\ 129 \\ 130$$

131 where  $\hat{x}(\xi)$  is the Fourier transform of  $x$ . In [2] it was suggested to use the same principle in a  
 132 general graph, by considering a smoothing kernel  $\mathcal{K} \in \mathbb{R}^{N \times N_0}$ , such as splines, and searching for  
 133 spectral multipliers of the form

$$134 \quad w_g = \mathcal{K} \tilde{w}_g . \\ 135 \\ 136$$

137 The algorithm which implements the graph convolution is described in 1.  
 138

---

#### 139   **Algorithm 1** Train Graph Convolution Layer

---

- 140   1: Given GFT matrix  $U$ , interpolation kernel  $K$ , weights  $w$ .
  - 141   2: **Forward Pass:**
  - 142   3: Fetch input batch  $x$  and gradients w.r.t outputs  $\nabla y$ .
  - 143   4: Compute interpolated weights:  $w_{f'f} = \mathcal{K} w_{\tilde{f}'f}$ .
  - 144   5: Compute output:  $y_{sf'} = U^T \left( \sum_f U x_{sf} \odot w_{f'f} \right)$ .
  - 145   6: **Backward Pass:**
  - 146   7: Compute gradient w.r.t input:  $\nabla x_{sf} = U^T \left( \sum_{f'} \nabla y_{sf'} \odot w_{f'f} \right)$
  - 147   8: Compute gradient w.r.t interpolated weights:  $\nabla w_{f'f} = U^T \left( \sum_s \nabla y_{sf'} \odot x_{sf} \right)$
  - 148   9: Compute gradient w.r.t weights  $\nabla w_{\tilde{f}'f} = K^T \nabla w_{f'f}$ .
- 

### 153   **3.2 Pooling with Hierarchical Graph Clustering**

154 In image and speech applications, and in order to reduce the complexity of the model, it is often  
 155 useful to trade-off spatial resolution with feature resolution as the representation becomes deeper.  
 156 For that purpose, pooling layers compute statistics in local neighborhoods, such as the average  
 157 amplitude, energy or maximum activation.

158 The same layers can be defined in a Graph by providing the equivalent notion of neighborhood.  
 159 In this work, we construct such neighborhoods at different scales using multi-resolution spectral  
 160 clustering [15], and consider both average and max-pooling as in standard convolutional network  
 161 architectures.

162    **4 Graph Construction**  
 163

164    **4.1 Using Prior Knowledge**  
 165

166    Whereas some recognition tasks in non-Euclidean domains, such as those considered in [2] or [10],  
 167    might have a prior knowledge of the graph structure of the input data, many other real-world appli-  
 168    cations do not have such knowledge. It is thus necessary to estimate a similarity matrix  $W$  from the  
 169    data before constructing the spectral network.

170    We consider in this paper two possible graph constructions, one unsupervised by measuring joint  
 171    feature statistics, and another one supervised using an initial network as a proxy for the estimation.  
 172

173    **4.2 Unsupervised Graph Estimation**  
 174

175    Given data  $X \in \mathbb{R}^{L \times N}$ , where  $L$  is the number of samples and  $N$  the number of features, the  
 176    simplest approach to estimating a graph structure from the data is to consider a distance between  
 177    features  $i$  and  $j$  given by

$$d(i, j) = \|X_i - X_j\|^2,$$

178    where  $X_i$  is the  $i$ -th column of  $X$ . While correlations are typically sufficient to reveal the intrinsic  
 179    geometrical structure of images [13], the effects of higher-order statistics might be non-negligible in  
 180    other contexts, especially in presence of sparsity. Indeed, in many situations the pairwise Euclidean  
 181    distances might suffer from unnormalized measurements. Several strategies and variants exist to  
 182    gain some robustness, for instance replacing the Euclidean distance by the  $Z$ -score (thus renormaliz-  
 183    ing each feature by its standard deviation), the “square-correlation” (computing the correlation of  
 184    squares of previously whitened features), or the mutual information.  
 185

186    This distance is then used to build a Gaussian diffusion Kernel [1]

$$\omega(i, j) = \exp^{-\frac{d(i, j)}{\sigma^2}}. \quad (1)$$

187    In our experiments, consider the variant of self-tuning diffusion kernel [16]  
 188

$$\omega(i, j) = \exp^{-\frac{d(i, j)}{\sigma_i \sigma_j}},$$

189    where  $\sigma_i$  is computed as the distance  $d(i, i_k)$  corresponding to the  $k$ -th nearest neighbor  $i_k$  of feature  
 190     $i$ .

191    The main advantage of (1) is that it does not require labeled data. Therefore, it is possible to estimate  
 192    the similarity using several datasets that share the same features, for example in text classification.  
 193

194    **4.3 Supervised Graph Estimation**  
 195

196    As discussed in the previous section, the notion of feature similarity is not well defined, as it depends  
 197    on our choice of kernel and criteria. Therefore, in the context of supervised learning, the relevant  
 198    statistics from the input signals might not correspond to our imposed similarity criteria. It may thus  
 199    be interesting to ask for the feature similarity that best suits a particular classification task.

200    A particularly simple approach is to use a fully connected network to determine the feature similarity.  
 201    Given a training set with normalized<sup>1</sup> features  $X \in \mathbb{R}^{L \times N}$  and labels  $y \in \{1, \dots, C\}^L$ , we initially  
 202    train a fully connected network  $\phi$  with  $K$  layers of weights  $W_1, \dots, W_K$ , using standard ReLU  
 203    activations and dropout. We then extract the first layer features  $W_1 \in \mathbb{R}^{N \times M_1}$ , where  $M_1$  is the  
 204    number of first-layer hidden features, and consider the distance  
 205

$$d_{sup}(i, j) = \|W_{1,i} - W_{1,j}\|^2, \quad (2)$$

206    that is then fed into the Gaussian kernel as in (1). The interpretation is that the supervised crite-  
 207    rion will extract through  $W_1$  a collection of linear measurements that best serve the classifica-  
 208    tion task. Thus two features are similar if the network decides to extract similar collection of linear  
 209    measurements.

---

210    <sup>1</sup>In our experiments we simply normalized each feature by its standard deviation, but one could also whiten  
 211    completely the data.

Another supervised strategy that further exploits the dependencies learnt at the deeper layers of the network consists in using the gradients of the supervised loss function with respect to the inputs as discriminative information. More concretely, if  $\mathcal{L}(X_l, y_l, (W_k)_{k \leq K})$  is the loss function associated to training example  $l$  and  $Z_{l,i} = \nabla_{x_i} \mathcal{L}(X_l, y_l, (W_k)_{k \leq K})$  is the input gradient corresponding to that example with respect to the  $i$ -th feature, then we consider the distance

$$d_{gsup}(i, j) = \|Z_{\cdot,i} - Z_{\cdot,j}\|^2, \quad (3)$$

that is, we measure the average similarity between the gradients for features  $i$  and  $j$ . That is, the similarity of two features depends on their respective influence (averaged over the training set) to the outcome of the network.

These constructions can be seen as “distilling” the information learnt by a first network into a kernel. In the general case where no assumptions are made on the dimension of the graph, it amounts to extracting  $N^2$  parameters from the first learning stage (which typically involves a much larger number of parameters). If, moreover, we assume a low-dimensional graph structure of dimension  $m$ , then  $mN$  parameters are extracted by projecting the resulting kernel into its leading  $m$  directions.

Finally, observe that one could simply replace the eigen-basis  $U$  obtained by diagonalizing the graph Laplacian by an arbitrary unitary matrix, which is then optimized by back-propagation together with the rest of the parameters of the model. We do not report results on this strategy, although we point out that it was the same learning complexity as the Fully Connected network (requiring  $O(KN^2)$  parameters, where  $K$  is the number of layers and  $N$  is the input dimension).

## 5 Experiments

In order to measure the performance of spectral networks on real-world data and to explore the effect of the graph estimation procedure, we performed experiments on three datasets from computer vision, text categorization and computational biology.

### 5.1 Reuters

We used the same version of the Reuters dataset as in [?], which consists of training and test sets each containing 201,369 documents from 50 mutually exclusive classes. Each document is represented as a log-normalized bag of words for 2000 common non-stop words. As a baseline we used the fully-connected network of [?] with two hidden layers consisting of 2000 and 1000 hidden units regularized with dropout.

We based the spectral network architecture on that of a classical convolutional network, namely by interleaving graph convolution and graph pooling layers and ending with a fully connected layer. Performing pooling at the beginning of the network was especially important to reduce the dimensionality in the graph domain and alleviate the expensive Graph Fourier Transform operation. We chose hyperparameters by performing initial experiments on a validation set consisting of one-tenth of the training data. All architectures were then trained using the same hyperparameters, which can be found in the Supplementary Material. Since the experiments were computationally expensive, we did not train all models until full convergence. This enabled us to explore more model architectures and obtain a clearer understanding of the effects of graph construction.

Table 1 reports the classification results on the test set. We observe that, despite having 4 times less parameters, the spectral network achieves nearly the same performance as the Dropout Fully Connected network. Also, Figure ?? displays the train and test error in terms of the epoch number. It shows that the spectral model is less exposed to overfitting and therefore more robust to optimization choices. Table 1 also confirms the advantage of using a supervised graph estimation on this task. We interpret it due to the high degree of sparsity of the input features, making correlation measurements unfit to describe the joint dependencies for that class of distributions. Locally connected ????

### 5.2 Merck Molecular Activity Challenge

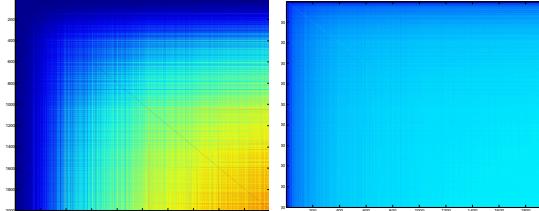
The Merck Molecular Activity Challenge is a computational biology benchmark where the task is to predict activity levels for various molecules based on the distances in bonds between different

270  
271  
272  
273

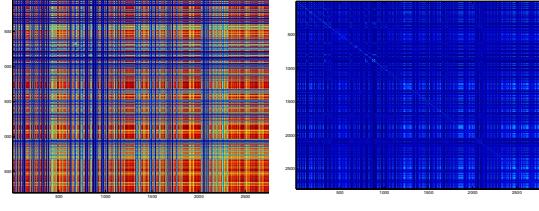
Graph	Architecture	# Params	Acc. (epoch 200)	Acc. (epoch 500)
-	FC-2000-1000-50	$8 \cdot 10^6$	70.2	<b>70.2</b>
Supervised	GC4-P4-FC-1000-50	$2 \cdot 10^6$	69.41	<b>70.0</b>
Supervised	GC8-P8-FC-1000-50	$2 \cdot 10^6$	69.15	-
Supervised	GC16-P4-GC16-P4-FC-1000-50	$2 \cdot 10^6$	68.68	-
Supervised	GC64-P8-GC64-P8-FC-1000-50	$2 \cdot 10^6$	68.63	-
Gaussian Kernel	GC4-P4-FC-1000-50	$2 \cdot 10^6$	64.14	-
Gaussian Kernel	GC8-P8-FC-1000-50	$2 \cdot 10^6$	64.79	-
Gaussian Kernel	GC16-P4-GC16-P4-FC-1000-50	$2 \cdot 10^6$	-	-
Gaussian Kernel	GC64-P8-GC64-P8-FC-1000-50	$2 \cdot 10^6$	-	-

284

285 Table 1: Results on Reuters Dataset  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295



300 (a) Global scaling..



301 (b) Local scaling.



302 (c) Global scaling..



303 (d) Local scaling.

304 Figure 1: Similarity graphs for the Reuters (top) and Merck DPP4 (bottom) datasets.  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319

320  
321  
322  
323

Graph	Architecture	Parameters	Test $R^2$
-	FC-4000-2000-1000-1000-50	$22.1 \cdot 10^6$	0.2728
Supervised	GC64-P8-GC64-P8-1000-1000-1	$3.8 \cdot 10^6$	0.2629
Gaussian Kernel, global scaling	GC64-P8-GC64-P8-1000-1000-1	$3.8 \cdot 10^6$	0.1992
Gaussian Kernel, local scaling	GC64-P8-GC64-P8-1000-1000-1	$3.8 \cdot 10^6$	-

Table 2: Results on the Merck Molecular Activity Dataset

atoms. For our experiments we used the DPP4 dataset which has 8193 samples and 2796 features. We chose this dataset because it was one of the more challenging and was of relatively low dimensionality which made the spectral networks tractable. As a baseline architecture, we used the state-of-the-art network of [?] which has 4 hidden layers and is regularized using dropout and weight decay. As before, we used one-tenth of the training data to find good hyperparameters and explore architectures.

Table 2 displays the results on the test set. Similarly as in the Reuters dataset, we observe that the spectral network nearly matches the performance of the fully connected dataset, albeit having 4 times less parameters. We note however that this dataset suffers from a clear domain shift between train and test distributions (explain that train error is very different from test error). Again, the supervised kernel performs clearly better than unsupervised kernels, which again fail to capture the appropriate notion of similarity between features for the task. Figure ?? displays the train and test error curves, showing the same behavior as in the Reuters dataset. Locally Connected?

### 5.3 Imagenet

In the experiments above our graph construction relied on an approximate estimation from the data. To measure the influence of the graph construction compared to the filter learning in the frequency domain, we performed the same experiments on the ImageNet dataset for which the graph is already known, namely it is the 2-D grid. The spectral network was thus a convolutional network whose weights were defined in the frequency domain. Training was performed exactly as in Figure 1, except that the linear transformation was a Fast Fourier Transform.

Our network consisted of 4 convolution/ReLU/max pooling layers with 48, 128, 256 and 256 feature maps, followed by 3 fully-connected layers with 4096 hidden units each regularized with dropout. We trained two versions of the network: one classical convolutional network and one as a spectral network where the weights were defined in the frequency domain only and were interpolated using a spline kernel. Both networks were trained for 40 epochs over the ImageNet dataset where input images were scaled down to  $128 \times 128$  to accelerate training.

Graph	Architecture	Parameters	Test Accuracy (Top 5)	Test Accuracy (Top 1)
2-D Grid	Convolutional Network	$3.8 \cdot 10^6$	71.854	46.24
2-D Grid	Spectral Network	$3.8 \cdot 10^6$	71.998	46.71

We see that both models yield nearly identical performance. Interestingly, the spectral network learns faster than during the first part of training, although both networks converge around the same time. This requires further investigation.

Word datasets. Mention that Using Word embeddings is also a possibility for future work.

## 6 Discussion

The graph construction using the Supervised Graph Estimation. Related to discovering latent graphical models

Limitations: When is weight sharing appropriate/ not.

378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431

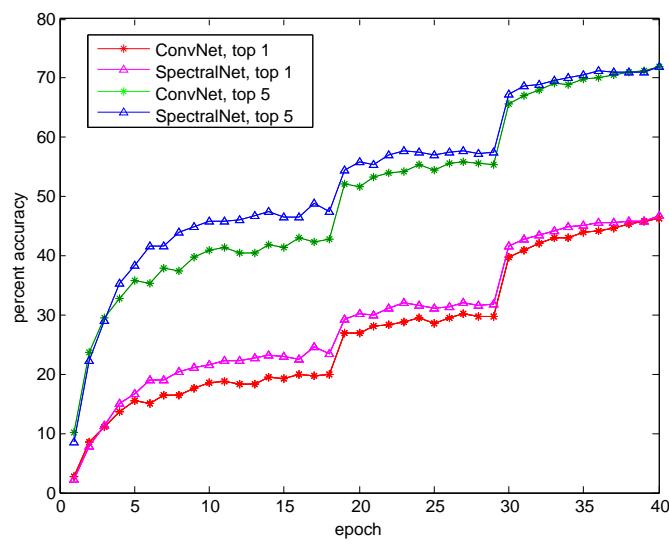


Figure 2: ConvNet vs. SpectralNet on ImageNet.

432 When is locality appropriate/not appropriate.

433

434 Complexity.

435 Alternatives we are

436

## 437 References

- 439 [1] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embed-  
440 ding and clustering. In *NIPS*, volume 14, pages 585–591, 2001.
- 441 [2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and deep  
442 locally connected networks on graphs. In *Proceedings of the 2nd International Conference on*  
443 *Learning Representations*, 2013.
- 444 [3] Xu Chen, Xiuyuan Cheng, and Stéphane Mallat. Unsupervised deep haar scattering on graphs.  
445 In *Advances in Neural Information Processing Systems*, pages 1709–1717, 2014.
- 446 [4] Adam Coates and Andrew Y Ng. Selecting receptive fields in deep networks. In *Advances in*  
447 *Neural Information Processing Systems*, pages 2528–2536, 2011.
- 448 [5] Karo Gregor and Yann LeCun. Emergence of complex-like cells in a temporal product network  
449 with local receptive fields. *arXiv preprint arXiv:1006.0448*, 2010.
- 450 [6] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel rahman Mohamed, Navdeep Jaitly,  
451 Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury. Deep  
452 neural networks for acoustic modeling in speech recognition. *Signal Processing Magazine*,  
453 2012.
- 454 [7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep  
455 convolutional neural networks. In F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger,  
456 editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran  
457 Associates, Inc., 2012.
- 458 [8] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–  
459 444, 05 2015.
- 460 [9] Stéphane Mallat. *A wavelet tour of signal processing*. Academic press, 1999.
- 461 [10] Jonathan Masci, Davide Boscaini, Michael M. Bronstein, and Pierre Vandergheynst. Shapenet:  
462 Convolutional neural networks on non-euclidean manifolds. *CoRR*, abs/1501.06297, 2015.
- 463 [11] Jiquan Ngiam, Zhenghao Chen, Daniel Chia, Pang W Koh, Quoc V Le, and Andrew Y Ng.  
464 Tiled convolutional neural networks. In *Advances in Neural Information Processing Systems*,  
465 pages 1279–1287, 2010.
- 466 [12] Pradeep Ravikumar, Martin J Wainwright, John D Lafferty, et al. High-dimensional ising  
467 model selection using  $\ell_1$ -regularized logistic regression. *The Annals of Statistics*, 38(3):1287–  
468 1319, 2010.
- 469 [13] Nicolas L Roux, Yoshua Bengio, Pascal Lamblin, Marc Joliveau, and Balázs Kégl. Learning  
470 the 2-d topology of images. In *Advances in Neural Information Processing Systems*, pages  
471 841–848, 2008.
- 472 [14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov.  
473 Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine  
474 Learning Research*, 15(1):1929–1958, 2014.
- 475 [15] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–  
476 416, 2007.
- 477 [16] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in neural  
478 information processing systems*, pages 1601–1608, 2004.

481

482

483

484

485