# Spectral Networks and Locally Connected Networks on Graphs

**Joan Bruna**
New York University
bruna@cims.nyu.edu

**Wojciech Zaremba**
New York University
woj.zaremba@gmail.com

**Arthur Szlam**
The City College of New York
aszlam@ccny.cuny.edu

**Yann LeCun**
New York University
yann@cs.nyu.edu

## Abstract

Convolutional Neural Networks are extremely efficient architectures in image and audio recognition tasks, thanks to their ability to exploit the local translational invariance of signal classes over their domain. In this paper we consider possible generalizations of CNNs to signals defined on more general domains without the action of a translation group. In particular, we propose two constructions, one based upon a hierarchical clustering of the domain, and another based on the spectrum of the graph Laplacian. We show through experiments that for low-dimensional graphs it is possible to learn convolutional layers with $O(1)$ parameters, resulting in efficient deep architectures.

## 1 Introduction

Convolutional neural networks have been extremely succesful in machine learning problems where the coordinates of the underlying data representation have a grid structure (in 1,2 and 3 dimensions) and the data to be studied in those coordinates has translational equivariance/invariance w.r.t. the grid. On a regular grid, a CNN is able to exploit several structures that play nicely together to greatly reduce the number of parameters in the system:

1. The translation structure, allowing the use of filters instead of generic linear maps.
2. The metric on the grid, allowing compactly supported filters.
3. The multiscale dyadic clustering of the grid, allowing subsampling.

If there are $n$ coordinates on a grid of dimension $d$, a fully connected network would need $O(n^2)$ parameters for each feature map. Using arbitrary filters instead of generic fully connected layers reduces the complexity to $O(n)$ parameters per feature map, as does using the metric structure by building a "locally connected" net [6, 11]. Using the two together gives $O(1)$ parameters. Finally, using the multiscale dyadic clustering allows each succesive layer to use a factor of $2^d$ less (spatial) coordinates per filter.

In this work, we will discuss constructions of deep neural networks on graphs other than regular grids, perhaps without any group structure. We will show that we are able to make use of (1) or (2) and (3) to get $O(n)$ parameters per filter, and a factor less coordinates per level; and we will discuss the question of obtaining $O(1)$ parameters per filter. These constructions allow efficient forward propagation and can be applied to datasets with very large number of coordinates.

The grid will be replaced by a weighted graph $W$ (an $m \times m$ symmetric and nonnegative matrix), with nodes indexed by a set $\Omega$.

### 1.1 Locality via $W$

The notion of locality can be generalized easily in the context of a graph. Indeed, the weights in a graph determine a notion of locality. For example, a straightforward way to define neighborhoods on $W$ is to set a threshold $\delta$ and take neighborhoods $N_\delta(j) = \{i \in \Omega : W_{ij} > \delta\}$. We can restrict attention to sparse "filters" with receptive fields given by these neighborhoods to get locally connected networks, reducing the number of parameters in a filter layer to $O(n)$ if the neighborhoods are sparse.

### 1.2 Harmonic analysis on weighted graphs

The combinatorial Laplacian $L = D - W$ or graph Laplacian $\mathcal{L} = I - D^{-1/2}WD^{-1/2}$ are generalizations of the Laplacian on the grid; and frequency and smoothness w.r.t. $W$ are interrelated through these operators [1, 14]. For simplicity, here we use the combinatorial Laplacian. If $f$ is an $m$ vector, a natural definition of the smoothness functional $||\nabla f||_W^2$ at a node $i$ is

$$||\nabla f||_W^2(i) = \sum_j W_{ij}[f(i) - f(j)]^2,$$

and

$$||\nabla f||_W^2 = \sum_i \sum_j W_{ij}[f(i) - f(j)]^2,$$

With this definition, the smoothest vector is a constant:

$$v_0 = \underset{f \in \mathbb{R}^m \ ||f||=1}{\arg\min} ||\nabla f||_W^2 = (1/\sqrt{m})1_m.$$

Each succesive

$$v_i = \underset{f \in \mathbb{R}^m \ ||f||=1 \ f \perp \{v_0, \ldots, v_{i-1}\}}{\arg\min} ||\nabla f||_W^2$$

is an eigenvector of $L$, and the eigenvalues $\lambda_i$ allow the smoothness of a vector $f$ to be read off from the coefficients of $f$ in $[v_0, ...v_{m-1}]$. Thus, just an in the case of the grid, where the eigenvectors of the Laplacian are the Fourier vectors, diagonal operators on the spectrum of the Laplacian modulate the smoothness of their operands. Moreover, using these diagonal operators reduces the number of parameters of a filter from $m^2$ to $m$.

These three structures above are all tied together through Laplacian on the grid:

1. Filters are multipliers on the eigenvalues of the Laplacian.
2. Functions that are smooth w.r.t. grid metric have coefficients with quick decay in the basis of eigenvectors.
3. The eigenvectors of the subsampled Laplacian are the low frequency eigenvectors of the Laplacian.

### 1.3 Multiresolution Analysis on Graphs

CNNs reduce the size of the grid via pooling and subsampling layers. These layers are possible because of the natural multiscale clustering of the grid: they input all the feature maps over a cluster, and output a single feature for that cluster. On the grid, the dyadic clustering behaves nicely w.r.t. the metric and the Laplacian (and so with the translation structure). There is a large literature on forming multiscale clusterings on graphs, see for example [10, 14, 4, 8]; finding multiscale clusterings that are provably guaranteed to behave well w.r.t. Laplacian on the graph is still an open area of research. In this work we will use a naive agglomerative method.

### 1.4 Contributions

Our contributions are summarized as follows:

- we use robust/simple geometry to construct $O(n)$ networks: we show that from a very weak geometric structure it is possible to obtain efficient architectures using $O(n)$ parameters, that we validate on low-dimensional graph datasets.

- We introduce a construction using $O(1)$ parameters which we empirically verify, and we discuss its connections with a harmonic analysis problem on graphs.

## 2 Spatial and Spectral Constructions

As above, let $W$ be a weighted graph with index set denoted by $\Omega$, and let $V$ be the eigenvectors of the graph Laplacian, ordered by eigenvalue.

### 2.1 Spectral construction

Given a weighted graph, we can try to generalize a convolutional net by operating on the spectrum of the weights (i.e. operating on the Laplacian). Suppose have a real valued nonlinearity $h : \mathbb{R} \to \mathbb{R}$; we can replace standard average pooling/subsampling by a dropping a set number of coefficients in $V$. That is, a layer of the network consists of

$$y_{k+1} \leftarrow D_k h(U_k F_k y_k) \tag{2.1}$$

with $y_0 = V^T x$, and where $x$ is the input into the network. Here

$$F_k = \begin{pmatrix} F_{k,1,1} & \dots & F_{k,1,f_{k-1}} \\ \vdots & & \vdots \\ F_{k,f_k,1} & \dots & F_{k,f_k,f_{k-1}} \end{pmatrix}, \tag{2.2}$$

where $F_{k,i,j}$ is a $d_k \times d_k$ diagonal matrix with the spectral multipliers on the diagonal, $f_k$ is the number of filter maps on the $k$th level, and $d_k$ is the number of spectral values to keep. $U_k$ is a block diagonal matrix with $f_k \times f_k$ blocks, each of which is the first $d_k$ columns of $V$, and $D_k$ is a block diagonal matrix with $f_k \times f_k$ blocks, each of which is the first $d_{k+1}$ rows of $V^T$. This means that $y_k$ is each of the $f_{k-1}$ filter outputs concatenated.

If the graph has an underlying group invariance this construction can discover it; the best example being the standard CNN; see 2.1.1. However, in many cases the graph does not have a group structure, or the group structure does not interact correctly with the Laplacian, and so we cannot think of each filter as passing a template across $\Omega$ and recording the correlation of the template with that location; $\Omega$ may not be homogenous in a way that allows this to make sense.

This construction can suffer from the fact that many reasonably nice graphs have meaningful eigenvectors only for the very top of the spectrum. Even when the individual high frequency eigenvectors are not meaningful, a cohort of high frequency eigenvectors may contain meaningful information. However this construction may not be able to access this information because it is nearly diagonal at the highest frequencies.

Finally, it is not obvious how to do either the fprop or the backprop efficiently while applying the nonlinearity on the space side, as we have to make the expensive multiplications by $U_k$ and $D_k$; and it is not obvious how to do standard nonlinearities on the spectral side. However, see 3.1

#### 2.1.1 Rediscovering standard CNN's

A simple, and in some sense universal, choice of weight matrix in this construction is the covariance of the data. Let $X = (x_k)_k$ be the input data distribution, with $x_k \in \mathbb{R}^n$. If each coordinate $j = 1 \dots n$ has the same variance,

$$\sigma_j^2 = E\left(|x(j) - E(x(j))|^2\right) ,$$

then diagonal operators on the Laplacian simply scale the principal components. While this may seem trivial, it is well known that the principal components of the set of images of a fixed size are (experimentally) the Fourier functions, organized by frequency. This can be explained by noticing that images are translation invariant, and hence the covariance operator

$$\Sigma(j,j) = E\left((x(j) - E(x(j)))(x(j') - E(x(j')))\right)$$

satisfies $\Sigma(j,j') = \Sigma(j-j')$, hence it is diagonalized by the Fourier basis. Moreover, since nearby pixels are more correlated than far away pixels, the principal components of the covariance are essentially ordered from low to high frequencies, which is consistent with the standard group structure of the Fourier basis. The upshot is that the the construction in 2.1 using the covariance as the data recovers a standard convolutional net without any prior knowledge.

### 2.1.2 $O(1)$ construction with smooth spectral multipliers

In the standard grid, we do not need a parameter for each Fourier function because the filters are compactly supported in space, but in this construction, each filter requires one parameter for each eigenvector on which it acts. Even if the filters were compactly supported in space in this construction, we still would not get less than $O(n)$ parameters per filter because the spatial response would be different at each location.

One possibility for getting around this is to generalize the duality of the grid. On the Euclidian grid, the decay in the spatial domain is translated into smoothness in the Fourier domain, and viceversa. It results that a funtion $x$ which is spatially localized has a smooth frequency response $\hat{x} = V^T x$. The eigenvectors of the Laplacian can be thought of as being arranged on a grid of the same size as the spatial coordinates.

In order to define a notion of smoothness, it is necessary to consider a topology in the domain of spectral coordinates. As previously discussed, on regular grids the topology is given by the notion of frequency, but this notion cannot be directly generalized to general graphs.

A particularly simple and navie choice consists in choosing a 1 dimensional topology obtained by ordering the eigenvectors according to their eigenvalues, which will be tested in the experiments of section 4. In this setting, the diagonal of each filter $F_{k,i,j}$ is parametrized by

$$\text{diag}(F_{k,i,j}) = \alpha_{k,i,j}\mathcal{K}\,,$$

where $\mathcal{K}$ is a cubic spline kernel and $\alpha_{k,i,j}$ are the spline coefficients. If one seeks to have filters with constant spatial support (ie, whose support is independent of the input size $n$), it follows that one can choose a sampling step $\Delta = O(n)$ in the spectral domain, which results in a constant number $\delta = n\Delta^{-1} = O(1)$ of coefficients $\alpha_{k,i,j}$ per filter.

Although results from section 4 seem to indicate that the 1d topology given by the spectrum of the Laplacian is efficient at creating spatially localized filters, a fundamental question is how to define a dual graph capturing the geometry of spectral coordinates. A possible algorithmic stategy could be to consider an input distribution $X = (x_k)_k$ consisting on spatially localized signals and to construct a dual graph $\widehat{W}$ by measuring the similarity of in the spectral domain: $\widehat{X} = V^T X$. The similarity could be measured for instance with $E((|\hat{x}| - E(|\hat{x}|))^T(|\hat{x}| - E(|\hat{x}|))$.

## 2.2 Spatial construction (locally connected networks)

Instead of generalizing a CNN by finding spectral multipliers, one can work entirely in space. We will need a multiscale clustering $\Omega_k$ and neighborhoods at each scale, given by the rows of the matrices $W_k$ (the rows and columns of $W_k$ are indexed by the clusters in $\Omega_{k-1}$, where $\Omega_0 = \Omega$. With these in hand, a layer of the net is

$$x_{k+1} \leftarrow L_k h(F_k x_k). \tag{2.3}$$

In this construction, $x_k$ is $d_{k-1} \times f_{k-1}$ vector, where $d_k$ is the number of clusters at level $k$, $f_k$ is the number of filters at level $k$, and $x_k$ is the filter responses vertically concatenated. Furthermore,

$$F_k = \begin{pmatrix} F_{k,1,1} & \dots & F_{k,1,f_{k-1}} \\ \vdots & & \vdots \\ F_{k,f_k,1} & \dots & F_{k,f_k,f_{k-1}} \end{pmatrix}, \tag{2.4}$$

where $F_{k,i,j}$ is a $d_{k-1} \times d_{k-1}$ sparse matrix with nonzero entries in the same locations as $w_{k-1}$. $h$ is again a componentwise nonlinearity, and $L_k$ outputs the result of a pooling operation over each cluster in $\Omega_k$.

In the current code, to build $W_k$ and $\Omega_k$ we use the following construction:

$$A_k(i,j) = \sum_{s \in \Omega_k(i)} \sum_{t \in \Omega_k(j)} W_{k-1}(s,t), \tag{2.5}$$

and $W_k = \text{rownormalize}(A)$, and $\Omega_k$ is found as an $\epsilon$ covering for $W_k$.

## 3 Relationship with previous work

There is a large literature on building wavelets on graphs, see for example [12, 5, 2, 3, 7]. A wavelet basis on a grid, in the language of neural networks, is a linear autoencoder with certain provable regularity properties (in particular, when encoding various classes of smooth functions, sparsity is gauranteed). The forward propagation in a classical wavelet transform strongly resembles the forward propagation in a neural network, except that there is only one filter map at each layer (and it is usually the same filter at each layer), and the output of each layer is kept, rather than just the output of the final layer. Classically, the filter is not learned, but constructed to facilitate the regularity proofs.

In the graph case, the goal is the same; except that the smoothness on the grid is replaced by smoothness on the graph. As in the classical case, most works have tried to construct the wavelets explicitly (that is, without learning), based on the graph, so that the corresponding autencoder has the correct sparsity properties. In this work, and the recent work [12], the "filters" are constrained by construction to have some of the regularity properties of wavelets, but are also trained so that they are appropriate for a task separate from (but perhaps related to) the smoothness on the graph. Whereas [12] still builds a (sparse) linear autoencoder that keeps the basic wavelet transform setup, this work focuses on nonlinear constructions; and in particular, tries to build analogues of CNN's.

### 3.1 Multigrid

We could improve both constructions, and to some extent unify them, with a multiscale clustering of the graph that plays nicely with the Laplacian. As mentioned before, in the case of the grid, the standard dyadic cubes have the property that the subsampling the Fourier functions on the grid to a coarser grid is the same as finding the Fourier functions on the coarser grid. This property would eliminate the annoying necessity of mapping the spectral construction to the finest grid at each layer to do the nonlinearity; and would allow us to interpret (via interpolation) the local filters at deeper layers in the spatial construction to be low frequency.

This kind of clustering is the underpinning of the multigrid method for solving discretized PDE's (and linear systems in general) [13]. There have been several papers extending the multigrid method, and in particular, the multiscale clustering(s) associated to the multigrid method, in settings more general than regular grids, see for example [10, 9] for situations as in this paper, and see [13] for the algebraic multigrid method in general. In this work, for simplicity, we use a naive multiscale clustering on the space side construction that is not guaranteed to respect the original graph's Laplacian, and no explicit spatial clustering in the spectral construction.

## 4 Numerical Experiments

We show experiments on two variations of the MNIST data set. In the first, we subsample the normal $28 \times 28$ grid to get $400$ coordinates. These coordinates still have a 2d structure, but it is not possible to use standard convolutions. We then make a dataset by placing $d = 4096$ points on the 2d unit sphere to get the set $S$, and project random MNIST images onto a random rotation of $S$ via bicubic interpolation. In all the experiments, we use Rectified Linear Units as nonlinearities. We train the models with cross-entropy loss, using a fixed learning rate of $0.1$ with momentum $0.9$.

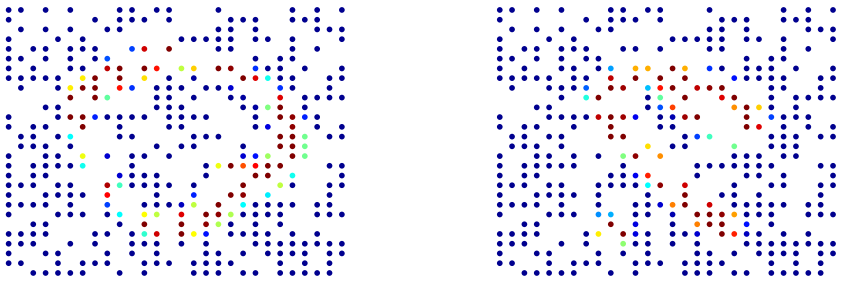
### 4.1 Subsampled MNIST

We apply the constructions from sections 2.1 and 2.2 to the subsampled MNIST dataset. Figure 4.1 shows examples of the resulting input signals, and figures 4.1, 4.1 show the hierarchical clustering constructed from the graph and some eigenfunctions of the graph Laplacian, respectively. The performance of various graph architectures is reported in Table 1. To serve as a baseline, we compute the standard Nearest Neighbor classifier, which performs slightly worse than in the full MNIST dataset ($2.8\%$). A two-layer Fully Connected neural network reduces the error to $1.8\%$. The geometrical structure of the data can be exploited with the CNN graph architectures. Local Receptive Fields adapted to the graph structure outperform the fully connected network. In particular, two layers of filtering and max-pooling define a network which efficiently aggregates information to the final classifier. The spectral construction performs slightly worse on this dataset. We considered

a frequency cutoff of $N/2 = 200$. However, the frequency smoothing architecture described in section 2.1.2, which contains the smallest number of parameters, outperforms the regular spectral construction.

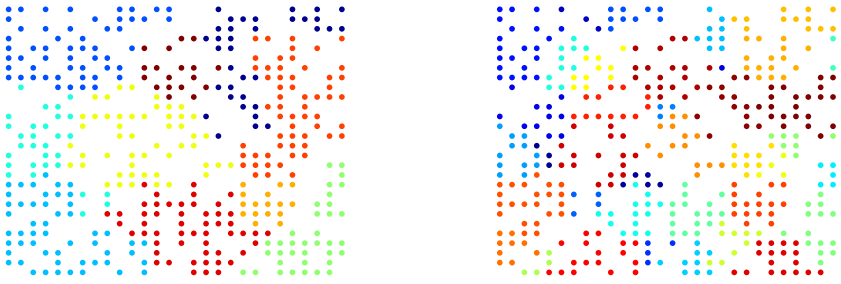
These results can be interpreted as follows. MNIST digits are characterized by localized oriented strokes, which require measurements with good spatial localization. Locally receptive fields are constructed to explicitly satisfy this constraint, whereas in the spectral construction the measurements are not enforced to become spatially localized. Adding the smoothness constraint on the spectrum of the filters improves classification results, since the filters are enforced to have better spatial localization.

This fact is illustrated in figure 4.1. We verify that Locally Receptive fields encode different templates across different spatial neighborhoods, since there is no global strucutre tying them together. On the other hand, spectral constructions have the capacity to generate local measurements that generalize across the graph. When the spectral multipliers are not constrained, the resulting filters tend to be spatially delocalized, as shown in panels (c)-(d). This corresponds to the fundamental limitation of Fourier analysis to encode local phenomena. However, we observe in panels (e)-(f) that a simple smoothing across the spectrum of the graph Laplacian restores some form of spatial localization and creates filters which generalize across different spatial positions, as should be expected for convolution operators.



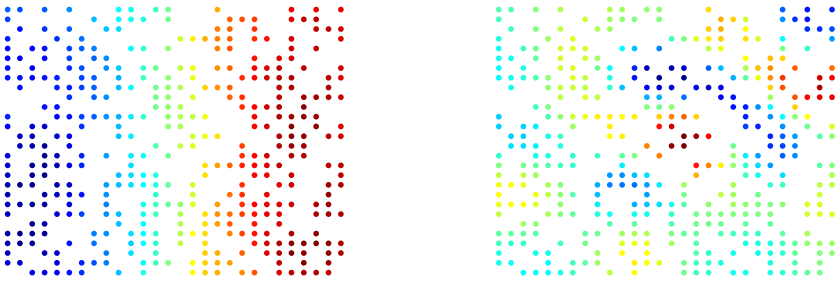
(a) (b)

Figure 1: Subsampled MNIST examples.



(a) (b)

Figure 2: Clusters obtained with the agglomerative clustering. (a) Clusters corresponding to the finest scale $k = 1$, (b) clusters for $k = 3$ .



(a) (b)

Figure 3: Examples of Eigenfunctions of the Graph Laplacian $v_2$, $v_{20}$.

Table 1: Classification results on MNIST subsampled on 400 random locations, for different architectures

| method | Parameters | Error |
|---|---|---|
| Nearest Neighbors | N/A | 4.11 |
| 400-FC800-FC50-10 | $3.6 \cdot 10^5$ | 1.8 |
| 400-LRF1600-MP800-10 | $7.2 \cdot 10^4$ | 1.8 |
| 400-LRF3200-MP800-LRF800-MP400-10 | $1.6 \cdot 10^5$ | **1.3** |
| 400-SP1600-10 ($d_1 = 300$, $\delta = n$) | $3.2 \cdot 10^3$ | 2.6 |
| 400-SP1600-10 ($d_1 = 300$, $\delta = 32$) | $1.6 \cdot 10^3$ | 2.3 |
| 400-SP4800-10 ($d_1 = 300$, $\delta = 20$) | $5 \cdot 10^3$ | 1.8 |

### 4.2 MNIST on the sphere

We test in this section the graph CNN constructions on another low-dimensional graph. In this case, we lift the MNIST digits to the sphere. The dataset is constructed as follows. We first sample 4096 random points $S = \{s_j\}_{j \leq 4096}$ from the unit sphere $S_2 \subset \mathbb{R}^3$. We then consider a reference frame $\mathbf{E} = (e_1, e_2, e_3)$ with $\|e_1\| = 1$, $\|e_2\| = 2$, $\|e_3\| = 3$ and a random covariance operator $\Sigma = (\mathbf{E} + W)^T(\mathbf{E} + W)$, where $W$ is a Gaussian iid matrix with variance $\sigma^2 < 1$. For each signal $x_i$ from the original MNIST dataset, we sample a covariance operator $\Sigma_i$ from the former distribution and consider its PCA basis $U_i$. This basis defines a point of view and in-plane rotation which we use to project $x_i$ onto $S$. Figure 4.2 shows examples of the resulting projected digits. Since the digits '6' and '9' are equivalent modulo rotations, we remove the '9' from the dataset. Figure 4.2 shows two eigenvectors of the graph Laplacian.

We first consider "mild" rotations with $\sigma^2 = 0.2$. The effect of such rotations is however not negligible. Indeed, table 2 shows that the Nearest Neighbor classifer performs considerably worse than in the previous example. All the neural network architectures we considered significatively improve over this basic classifier. Furthermore, we observe that both convolutional constructions match the fully connected constructions with far less parameters (but in this case, do not improve its performance). Figure 4.2 displays the filters learnt using different constructions. Again, we verify that the smooth spectral construction consistently improves the performance, and learns spatially localized filters, even using the naive 1d organization of eigenvectors, which detect similar features across different locations of the graph (panels (e)-(f)).

Finally, we consider the uniform rotation case, where now the basis $U_i$ is a random basis of $\mathbb{R}^3$. In that case, the intra-class variability is much more severe, as seen by inspecting the performance of the Nearest neighbor classifier. All the previously described neural network architectures significantly improve over this classifier, although the performance is notably worse than in the mild rotation scenario. In this case, an efficient representation needs to be fully roto-translation invariant. Since

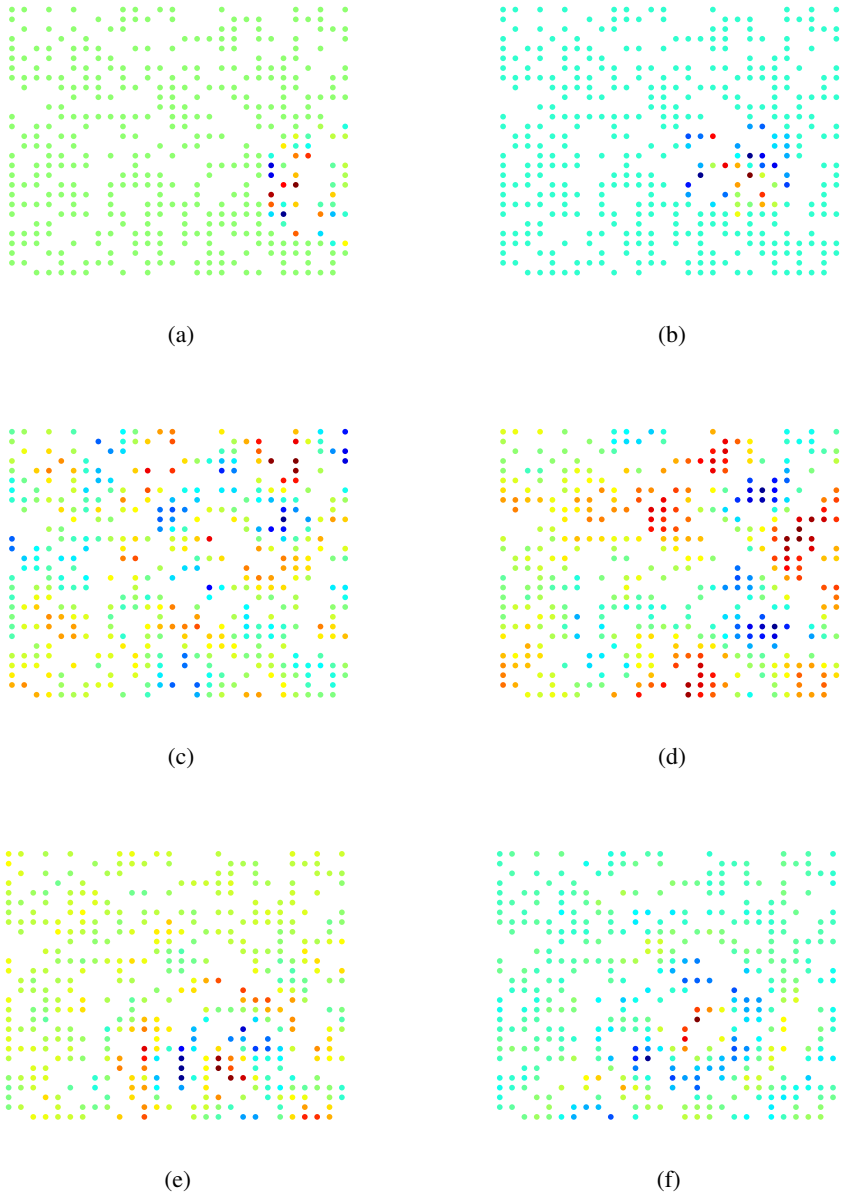
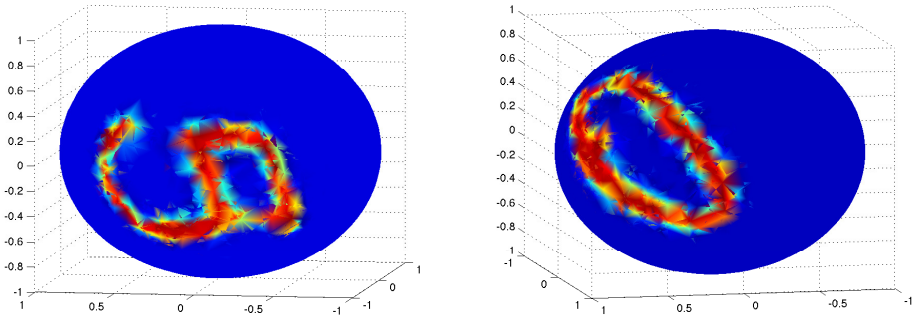


Figure 4: Subsampled MNIST learnt filters using spatial and spectral construction. (a)-(b) Two different receptive fields encoding the same feature in two different clusters. (c)-(d) Example of a filter obtained with the spectral construction. (e)-(f) Filters obtained with smooth spectral construction.

this is a non-commutative group, it is likely that deeper architectures perform better than the models considered here.

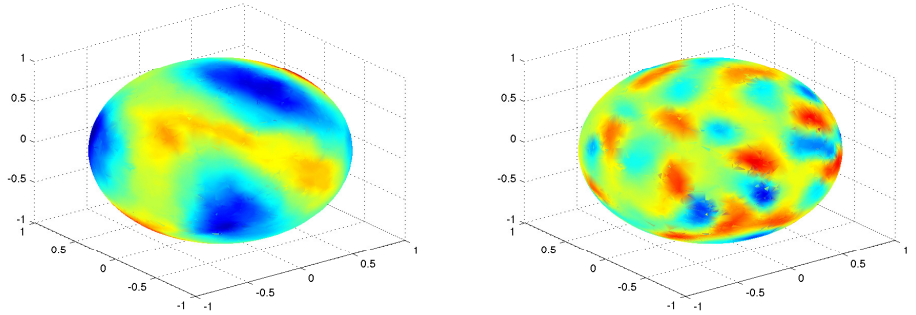
## 5 Conclusion

Using graph-based analogues of convolutional architectures can greatly reduce the number of parameters in a neural network without worsening (and often improving) the test error, while simultaneously giving a faster forward propagation. These methods can be scaled to data with a large number of coordinates that have a notion of locality.



(a) (b)

Figure 5: Examples of some MNIST digits on the sphere.



(a) (b)

Figure 6: Examples of Eigenfunctions of the Graph Laplacian $v_{20}$, $v_{100}$

There is much to be done here. We suspect with more careful training and deeper networks we can consistently improve on fully connected networks on "manifold like" graphs like the sampled sphere. Furthermore, we intend to apply these techniques to less artifical problems, for example, on netflix like recommendation problems where there is a biclustering of the data and coordinates. Finally, the fact that smoothness on the naive ordering of the eigenvectors leads to improved results and localized filters suggests that it may be possible to make "dual" constructions with $O(1)$ parameters per filter in much more generality than the grid.

Table 2: Classification results on the MNIST-sphere dataset generated using partial rotations, for different architectures

| method | Parameters | Error |
|---|---|---|
| Nearest Neighbors | N/A | 19 |
| 4096-FC2048-FC512-9 | $10^7$ | **5.6** |
| 4096-LRF4620-MP2000-FC300-9 | $8 \cdot 10^5$ | **6** |
| 4096-LRF4620-MP2000-LRF500-MP250-9 | $2 \cdot 10^5$ | 6.5 |
| 4096-SP32K-MP3000-FC300-9 ($d_1 = 2048$, $\delta = n$) | $9 \cdot 10^5$ | 7 |
| 4096-SP32K-MP3000-FC300-9 ($d_1 = 2048$, $\delta = 64$) | $9 \cdot 10^5$ | **6** |

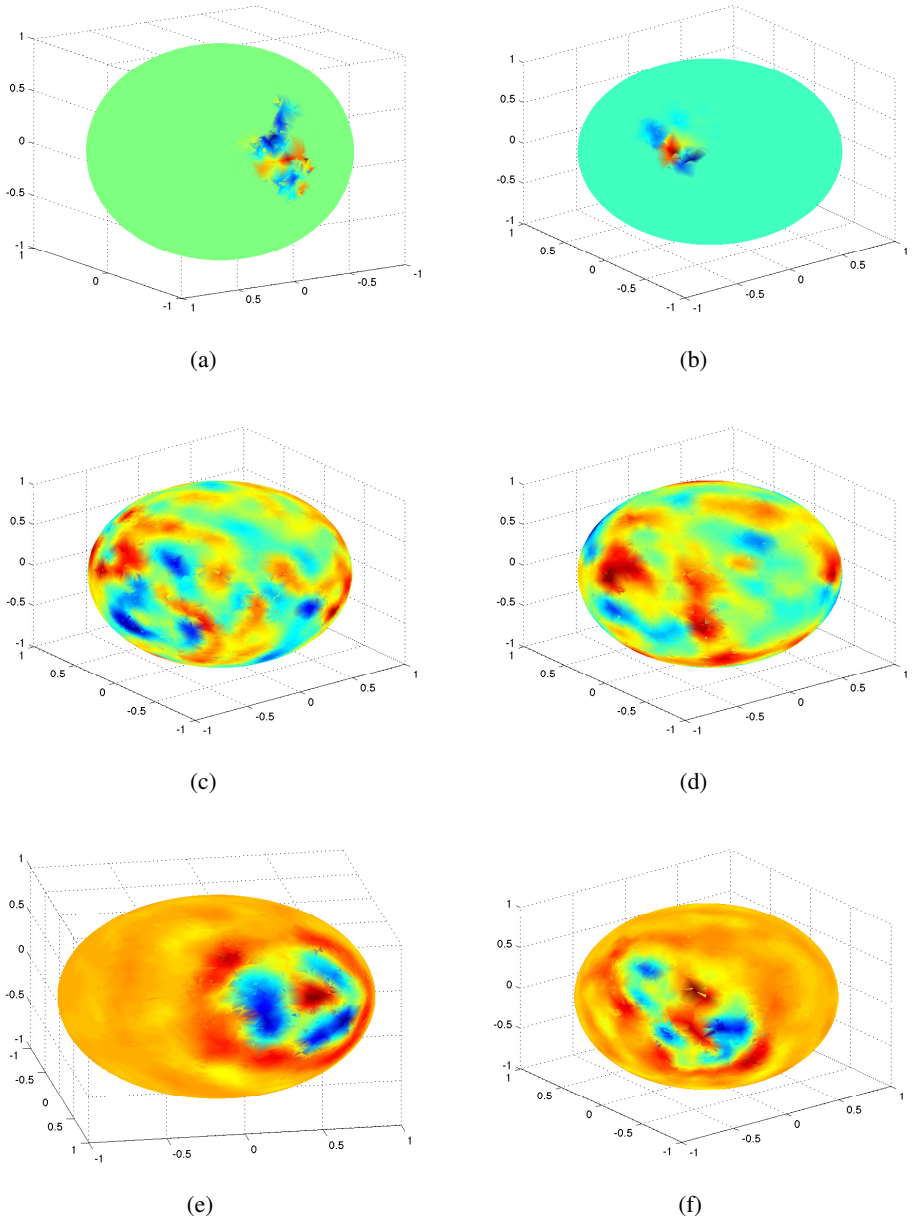


Figure 7: Filters learnt on the MNIST-sphere dataset, using spatial and spectral construction. (a)-(b) Two different receptive fields encoding the same feature in two different clusters. (c)-(d) Example of a filter obtained with the spectral construction. (e)-(f) Filters obtained with smooth spectral construction.

## References


[1] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society.

[2] R.R. Coifman and M. Maggioni. Diffusion wavelets. *Appl. Comp. Harm. Anal.*, 21(1):53–94, July 2006.

[3] Mark Crovella and Eric D. Kolaczyk. Graph wavelets for spatial traffic analysis. In *INFOCOM*, 2003.

[4] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. Weighted graph cuts without eigenvectors

Table 3: Classification results on the MNIST-sphere dataset generated using uniformly random rotations, for different architectures

| method | Parameters | Error |
|---|---|---|
| Nearest Neighbors | NA | 80 |
| 4096-FC2048-FC512-9 | $10^7$ | 52 |
| 4096-LRF4620-MP2000-FC300-9 | $8 \cdot 10^5$ | 61 |
| 4096-LRF4620-MP2000-LRF500-MP250-9 | $2 \cdot 10^5$ | 63 |
| 4096-SP32K-MP3000-FC300-9 ($d_1 = 2048$, $\delta = n$) | $9 \cdot 10^5$ | 56 |
| 4096-SP32K-MP3000-FC300-9 ($d_1 = 2048$, $\delta = 64$) | $9 \cdot 10^5$ | **50** |


a multilevel approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(11):1944–1957, November 2007.

[5] Matan Gavish, Boaz Nadler, and Ronald R. Coifman. Multiscale wavelets on trees, graphs and high dimensional data: Theory and applications to semi supervised learning. In Johannes Frnkranz and Thorsten Joachims, editors, *ICML*, pages 367–374, 2010.

[6] Karol Gregor and Yann LeCun. Emergence of complex-like cells in a temporal product network with local receptive fields. *CoRR*, abs/1006.0448, 2010.

[7] I. Guskov, W. Sweldens, and P. Schröder. Multiresolution signal processing for meshes. *Computer Graphics Proceedings (SIGGRAPH 99)*, pages 325–334, 1999.

[8] George Karypis and Vipin Kumar. Metis - unstructured graph partitioning and sparse matrix ordering system, version 2.0. Technical report, 1995.

[9] D. Kushnir, M. Galun, and A. Brandt. Efficient multilevel eigensolvers with applications to data analysis tasks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(8):1377–1391, 2010.

[10] Dan Kushnir, Meirav Galun, and Achi Brandt. Fast multiscale clustering and manifold identification. *Pattern Recognition*, 39(10):1876 – 1891, 2006. ¡ce:title¿Similarity-based Pattern Recognition¡/ce:title¿.

[11] Quoc V. Le, Jiquan Ngiam, Zhenghao Chen, Daniel Chia, Pang Wei Koh, and Andrew Y. Ng. Tiled convolutional neural networks. In *In NIPS*, 2010.

[12] Raif M. Rustamov and Leonidas Guibas. Wavelets on graphs via deep learning. In *NIPS*, 2013.

[13] Ulrich Trottenberg and Anton Schuller. *Multigrid*. Academic Press, Inc., Orlando, FL, USA, 2001.

[14] U. von Luxburg. A tutorial on spectral clustering. Technical Report 149, 08 2006.