PERMUTOHEDRAL LATTICE CNNS

Martin Kiefel, Varun Jampani and Peter V. Gehler
Max Planck Institute for Intelligent Systems
Tübingen, 72076, Germany
{martin.kiefel, varun.jampani, peter.gehler}@tuebingen.mpg.de

ABSTRACT

This paper presents a convolutional layer that is able to process sparse input features. As an example, for image recognition problems this allows an efficient filtering of signals that do not lie on a dense grid (like pixel position), but of more general features (such as color values). The presented algorithm makes use of the permutohedral lattice data structure. The permutohedral lattice was introduced to efficiently implement a bilateral filter, a commonly used image processing operation. Its use allows for a generalization of the convolution type found in current (spatial) convolutional network architectures.

1 INTRODUCTION

In the domain of image recognition, the convolutional layer of a CNN today is almost exclusively associated with a spatial convolution in the image domain. In this work we will take a more signal theoretic viewpoint of the convolutional operation and present an algorithm that allows to process also sparse input data. This work is inspired by the use of special data structures (Adams et al., 2010) for bilateral filters (Aurich & Weule, 1995; Smith & Brady, 1997; Tomasi & Roberto, 1998) and generalizes it for the use of convolutional architectures.

Although the approach presented here is more general, the following two scenarios are instructive. Consider that at training time we have access to full resolution images to train a classifier. At test time only a random number of pixels from the test image is available. In other words, we sample the signal differently during training and test time. For a traditional CNN this would require a preprocessing step, for example to map from subsets of pixels to a dense grid that is the image. In our view there is no change, it is not required that we have a dense grid and access to all pixels of the image. That is the integration domain does not change. This is one example of sparsity, here we deal with a set of pixels, whose values are RGB and features are position. Similarly, color information can be used to define the filtering operation as well. One can devise a convolution with a domain respecting color *and* location information (or color alone). One view from the image processing community is that of an *edge-aware* filter, the filter will be adaptive to the color and/or gradient of the image. RGB values do not lie on a regular dense grid, therefore a direct expansion of the spatial convolution is not applicable.

This approach falls into line with the view on encoding invariants (Mallat, 2012). It is possible to encode our knowledge invariants that we have about the data with the new way of looking at the data. Encoded in a spatial convolution is the prior knowledge about translation invariance. How to encode roation invariance, how similarity in color space? In the view we take here these are simply convolutions over different domains. A grid based convolution cannot easily be used to work with the sparse data (an interpolation might be needed) but the permutohedral lattice provides the right space and allows efficient implementations. Therefore the runtime is comparable to the ones of spatial convolutions, depending on the size of the invariants to include and can simply be used as a replacement of the traditional layers.

2 PERMUTOHEDRAL LATTICE CONVOLUTION

We propose a convolution operation of a d-dimensional input space that entirely works on a lattice. Input data is a tuple (f_i, v_i) of feature locations $f_i \in \mathbb{R}^d$ and corresponding signal values $v_i \in$

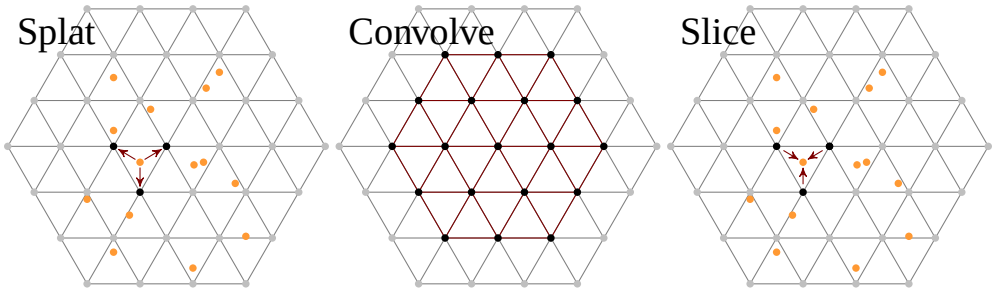

Figure 1: The permutohedral convolution consists of three steps: first the samples are splatted onto the lattice, then a convolution operates on the lattice considering a margin of $s = 2$ neighbors of a node, and finally the result of the convolution is transformed back to the output samples.

$\mathbb{R}$. Importantly, this does not assume the feature locations f_i to be sampled on a regular grid, for example f_i can be position and RGB value. We then map the input signal to a regular structure, the so-called permutohedral lattice. A convolution then operates on the constructed lattice and the result is mapped back to the output space. Hence, the entire operation consists of three stages (see fig. 1): *splat* (the mapping to the lattice space), *convolution* and *slice* (the mapping back from the lattice). This strategy has already been used to implement fast Gaussian filtering (Paris & Durand, 2009; Adams et al., 2010; 2009). Here we generalize it to arbitrary convolutions.

The permutohedral lattice is the result of the projection of the set $\mathbb{Z}^{d+1}$ onto a plane defined by its orthogonal vector $\mathbf{1} \in \mathbb{R}^{d+1}$. This d dimensional plane is embedded into $\mathbb{R}^{d+1}$. The lattice points tessellate the subspace with regular cells. Given a point from the embedding space, it is efficient to find the enclosing simplex of the projection onto the plane. We will represent a sparse set of points from $\mathbb{R}^d$ by a sparse set of simplex corners in the lattice. Importantly, the number of corners does not grow exponentially with the dimension d as it would for an axis-align simplex representation. We continue to describe the different parts of the permutohedral convolution.

The splat and slice operations take the role of an interpolation between the different signal representations. All input samples that belong to a cell adjacent to lattice point j are summed up and weighted with the barycentric coordinates to calculate the value $l_j = \sum_{i \in C(j)} b_{i,j} v_i$. This is the splatting operation. The barycentric coordinates $b_{i,j}$ depend on both the corner point j and the feature location f_i. The reverse operation slice finds an output value v'_k by using its barycentric coordinates inside the lattice simplex and sums over the corner points $v'_k = \sum_{j \in C(k)} b_{k,j} l'_j$.

The convolution is then performed on the permutohedral lattice. It uses a convolution kernel w_n to compute $l'_{j'} = \sum_{(n,j) \in N(j')} w_n l_j$. The convolution kernel w_n is problem specific and its domain is restricted to the set of neighboring lattice points $N(j)$. For bilateral filters, this is set to be a Gaussian filter, here we learn the kernel values using back-propagation.

The size of the neighborhood takes a similar role as the filter size (spatial extent) of the grid-based CNN. A transitional convolutional kernel which considers s sampled points to either side has $(2s + 1)^d \in \mathcal{O}(s^d)$ parameters. A comparable filter on the permutohedral lattice with a s neighborhood has $(s + 1)^{d+1} - s^{d+1} \in \mathcal{O}(s^d)$ elements.

3 Sparse CNNs and Encoding Invariants

The permutohedral convolution can be used as a new building block in a CNN architecture. We will omit the derivation of the gradients for the filter elements with respect to the output of such a new layer due to space constraints. We will discuss two possible application scenarios.

First, as mentioned before we are free to change the sampling of the input signal of a lattice-based convolution. The choice of the sampling is problem specific. Missing measurements or domain specific sampling techniques that gather more information in highly discriminant areas are only two possible scenarios. Furthermore, as we will show in our experiments the method is robust in cases where train-time sampling and test-time sampling do not coincide.

LeNet	Permutohedral	Subsample (100%)	Subsample (60%)	Subsample (20%)
0.9919	0.9903	0.9745	0.9537	0.8689

Table 1: Classification accuracy of the conducted experiments compared to the LeNet (LeCun et al., 1998) implementation that is part of the caffe implementation (Jia et al., 2014).

Second, the proposed method provides a tool to encode additional data invariances in a principled way. A common technique to include domain knowledge is to artificially augment the training set with deformations that leave the output signal invariant, such as translations, rotations, or nosiy versions.

A feature mapping Φ is invariant with respect to a transformation L and a signal v if $\Phi(v) \approx \Phi(v_L)$. In the case where L belongs to a set of translations a possible invariant feature is the convolution with a window function w (given its support has the right size) $\Phi(v, s) = \int w(t)v(s-t)\mathrm{d}t$. The same idea can be applied to the more general case and again calculating a mean with the help of a window function:

$$\Phi(v, L) = \int w(M)v(M^{-1}L)\mathrm{d}M.$$

We can use the permutohedral convolution to encode invariances like rotation and translation. Approximating the above integral by a finite sum and using lattice points as integration samples we arrive at $\Phi(v, L) \approx \sum_{\text{lattice: } M} w_M v(M^{-1}L)$. We further approximate $v(M^{-1}L)$ with look-up at a lattice point location.

Consider the case of rotation and translation invariance. More intuitively, we stack rotated versions of the input images onto each other in a 3 dimensional space – 2 dimensions for the location of a sample and 1 dimension for the rotation of the image. A grid-based convolution would not work here because the rotated image points might not coincide with a grid anymore. Filtering in the permutohedral space naturally respects the augmented feature space.

4 Experiments

We take a reference implementation of LeNet (LeCun et al., 1998) that is part of the caffe project (Jia et al., 2014) on the MNIST dataset as a starting point for the following experiments. The permutohedral convolutional layer is also implemented in this framework.

We first compare the LeNet in terms of test-time accuracy when substituting only the first of the convolutional layers with a (position only) permutohedral layer and leave the rest identical. Table 1 shows that a similar performance is achieved, so it seems model flexibility is not lost.

One of the strengths of the proposed method is that it does not depend on a regular grid sampling as the tranditional convolution operators. We highlight this feature with the following experiment. We randomly sample continuous points in the input image, use their interpolated values as signal and continuous positions as features. This mimicks sub-sampling of a high-dimensional signal, interestingly, we could train models with a different amount of sub-sampling than at test time. The permutohedral representation is robust with respect to this sparse input signal. Table 1 shows experiments with different signal degradation levels.

In the future we will investigate the change in classification performance with the new layer inside a network architecture that directly models domain knowledge.

5 Conclusion

This paper presents a generalization of the convolutional operation to sparse input signals. We envision many consequences of this work. Consider signals that are naturally represented as measurements instead of images, like MRT scan readings. The permutohedral lattice filtering avoids the pre-processing assembling operation into a dense image, it is possible to work on the measured sparse signal directly. Another promising use of this filter is to encode scale invariance, typically

this is encoded by presenting multiple scaled versions of an image to several branches of a network. The convolution presented here can be defined on the continuous range of image scales without a finite subselection. In summary, this algorithm allows to encode prior knowledge about the observed signal to define the domain of the convolution. The typical spatial filter of CNNs is a particular type of prior knowledge, we generalize this to sparse signals.

REFERENCES

Adams, Andrew, Gelfand, Natasha, Dolson, Jennifer, and Levoy, Marc. Gaussian kd-trees for fast high-dimensional filtering. In *ACM SIGGRAPH 2009 Papers*, SIGGRAPH '09, pp. 21:1–21:12, New York, NY, USA, 2009.

Adams, Andrew, Baek, Jongmin, and Davis, Myers Abraham. Fast high-dimensional filtering using the permutohedral lattice. *Comput. Graph. Forum*, 29(2):753–762, 2010.

Aurich, Volker and Weule, Jörg. Non-linear Gaussian filters performing edge preserving diffusion. In *Mustererkennung 1995, 17. DAGM-Symposium, Bielefeld, 13.-15. September 1995, Proceedings*, pp. 538–545, 1995.

Jia, Yangqing, Shelhamer, Evan, Donahue, Jeff, Karayev, Sergey, Long, Jonathan, Girshick, Ross, Guadarrama, Sergio, and Darrell, Trevor. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

Mallat, Stéphane. Group invariant scattering. *Communications in Pure and Applied Mathematics*, 65(10):1331–1398, 2012.

Paris, Sylvain and Durand, Frédo. A fast approximation of the bilateral filter using a signal processing approach. *International Journal of Compututer Vision*, 81(1):24–52, January 2009.

Smith, Stephen M. and Brady, J. Michael. SUSAN – a new approach to low level image processing. *Int. J. Comput. Vision*, 23(1):45–78, May 1997. ISSN 0920-5691.

Tomasi, Carlo and Roberto, Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the Sixth International Conference on Computer Vision*, ICCV '98, pp. 839–846, Washington, DC, USA, 1998. IEEE Computer Society.