Dynamic Edge-Conditioned Filters in CNNs on Graphs
Martin Simonovsky, Nikos Komodakis

Motivation
CNNs are massively popular in tasks where the underlying data representation has a grid structure. But in many other tasks, graph-structured data is common: meshes & point clouds, knowledge bases, social graphs, chemical compounds,...

How to generalize CNNs to graph domains

How to organize weight-sharing in convolutions?

Observation: Graphs may carry a lot of additional information in edge attributes: scalar weights, relation types, mutual offset or overlap information, ...
We propose a convolution-like operation on graphs which
• can exploit edge attributes in any form processable by a neural network
• can be used on datasets with varying graph structures (spatial formulation)
• can be used to generalize the regular convolution on grids

ECC - Edge-Conditioned Convolution

• Notation: Directed graph $G = (V, E)$ with edge attributes (labels) $l: E \rightarrow \mathbb{R}^d$ and vertex signals (features) $X: V \rightarrow \mathbb{R}^m$ on network layer $l$

• Convolution: Weighted sum of signals over a neighbourhood $N(i) = \{j; (j, i) \in E\}$

$X'(i) = \frac{1}{|N(i)|} \sum_{j \in N(i)} w^l(i, j) X(j) + b^l$

• Key idea: Filters $\theta^l$ conditioned on the respective edge labels $l$, and computed dynamically using filter-generating network ($F^l$) $F^l: \mathbb{R}^d \rightarrow \mathbb{R}^{|E|\times m\times l-1}$

• Learned parameters $b^l$ and $w^l$ vs. generated convolution filters $\theta^l$

• Complexity: $\leq |E|$ evaluations of $F^l(\cdot, \cdot) + |E|$ matrix-vector multiplications

Network Architecture

Pooling Layer

• Signal on $G = G(l)$ aggregated onto the vertices of a new, coarsened graph $G(l)$

• Problem-specific meaningful downsampling of vertices, mapping $G(l) \rightarrow G(l-1)$, creating $G(l)$ and $L(l-1)$ (reduction)

Point Cloud Classification

• Directed graph construction from point cloud $P$: edges connect nearby points

$\{(j, i); \|x_j - x_i\| < r\}$, for $\delta = \rho_j - \rho_i$

• Coarsening: graphs built from point cloud pyramid created with VoxelGrid [2] downsampling, $M$ assigns points to the nearest centroid.

• Augmentation: rotation about up-axis, jitter scale, mirroring, point dropout

Architecture (Sydney): C16(16)-C32(16/0)-C64(16)-C128(16)-M0(75,1,5)-GAP(FC(48-16)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(d_{L-1}))

Graph Classification

• Coarsening [3]: vertices halved by the sign of the largest eigenvector of Laplacian, followed by kernel reduction with spectral sparsification of edges.

• Augmentation: randomized sparsification (fractional pooling [4] analogy)

• Configuration (NCL1): C16(16)-C32(16)-C64(16)-M0(75,1,5)-GAP(FC(48-16)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(128)-FC(d_{L-1}))

Generalization of Convolution on Grids

Demonstration in 1D

• Graph representation of convolution with a centered filter of size $s = 3$:

$X'(i) = \sum_{k=-s}^{s} F(x_i+k) W_i \cdot X(x_i+k)$

• Linear filter-generating network: $F_i(x_i, y) = W_i(x, y) = W(k)$

MNIST as point cloud

Train acc. | Test acc.
--- | ---
Full point cloud | 99.12 | 99.14
Sparse point cloud | 99.16 | 99.14
Baseline/one-hot | 99.53 | 99.37

Sampled first layer filters

www.github.com/mys007/enc martin.simonovsky@enpc.fr