The World of Graph Neural Networks: From the Mystery of Generalization to Foundational Limitations

Gitta Kutyniok

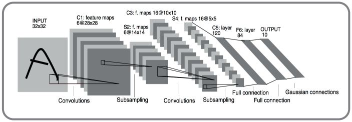
(Ludwig-Maximilians-Universität München and University of Tromsø)

Mathematics for Deep Learning Opening Workshop
University of Bath, April 21 – 22, 2022

The Dawn of Deep Learning in Public Life

Self-Driving Cars

Telecommunication/
Speech Recognition



Legal Issues

Health Care

Impact on Mathematical Problem Settings

Some Examples:

- Inverse Probleme/Imaging Science *(2012–)*
  - $\rightsquigarrow$ *Denoising*
  - $\rightsquigarrow$ *Edge Detection*
  - $\rightsquigarrow$ *Inpainting*
  - $\rightsquigarrow$ *Classification*
  - $\rightsquigarrow$ *Superresolution*
  - $\rightsquigarrow$ *Limited-Angle Computed Tomography*
  - $\rightsquigarrow$ ...

Impact on Mathematical Problem Settings

Some Examples:

- Inverse Probleme/Imaging Science *(2012–)*
  $\rightsquigarrow$ *Denoising*
  $\rightsquigarrow$ *Edge Detection*
  $\rightsquigarrow$ *Inpainting*
  $\rightsquigarrow$ *Classification*
  $\rightsquigarrow$ *Superresolution*
  $\rightsquigarrow$ *Limited-Angle Computed Tomography*
  $\rightsquigarrow$ ...

- Numerical Analysis of Partial Differential Equations *(2017–)*
  $\rightsquigarrow$ *Black-Scholes PDE*
  $\rightsquigarrow$ *Allen-Cahn PDE*
  $\rightsquigarrow$ *Parametric PDEs*
  $\rightsquigarrow$ ...

Problem with Trustworthiness

By *Linda Geddes* 5th December 2018

Computers can be made to see a sea turtle as a gun or hear a concerto as someone's voice, which is raising concerns about using artificial intelligence in the real world.

MACHINE MINDS | ARTIFICIAL INTELLIGENCE BBC

Role of Mathematics

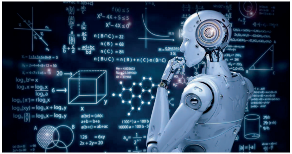
Two Key Challenges for Mathematics:

Mathematics for Deep Learning!

- Can we derive a deep mathematical understanding of deep learning?
- How can we make deep learning more robust?
- ...

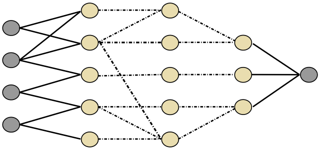
Deep Learning for Mathematics!

- How can we use deep learning to improve imaging science?
- Can we develop superior PDE solvers via deep learning?
- ...



Delving Deeper into Deep Neural Networks...

Definition of a Deep Neural Network



Definition:

Assume the following notions:

- $d \in \mathbb{N}$: Dimension of input layer.
- L: Number of layers.
- $\rho : \mathbb{R} \to \mathbb{R}$: (Non-linear) function called *activation function*.
- $T_\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$, $\ell = 1, \ldots, L$, where $T_\ell x = W^{(\ell)} x + b^{(\ell)}$

Then $\Phi : \mathbb{R}^d \to \mathbb{R}^{N_L}$ given by

$$\Phi(x) = T_L \rho(T_{L-1} \rho(\ldots \rho(T_1(x)))), \quad x \in \mathbb{R}^d,$$

is called *(deep) neural network (DNN)*.

Second Appearance of Neural Networks

Key Observations by Y. LeCun et al. (around 2000):

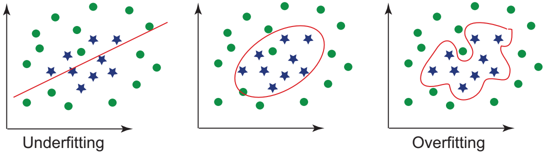
- Drastic improvement of computing power.
  $\rightsquigarrow$ *Networks with hundreds of layers can be trained.*
  $\rightsquigarrow$ *Deep Neural Networks!*
- Age of Data starts.
  $\rightsquigarrow$ *Vast amounts of training data is available.*

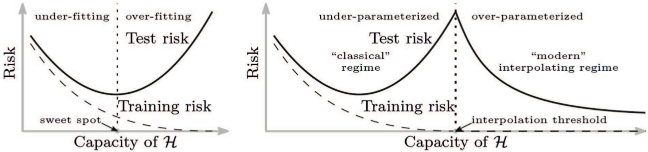
Second Appearance of Neural Networks

Key Observations by Y. LeCun et al. (around 2000):

- Drastic improvement of computing power.
  - $\rightsquigarrow$ *Networks with hundreds of layers can be trained.*
  - $\rightsquigarrow$ *Deep Neural Networks!*
- Age of Data starts.
  - $\rightsquigarrow$ *Vast amounts of training data is available.*

Surprising Phenomenon:





(Source: Belkin, Hsu, Ma, Mandal; 2019)

- **Expressivity:**
  - Which *aspects of a neural network architecture* affect the performance of deep learning?

$\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*

Mathematics for Deep Learning

- **Expressivity:**
  - Which *aspects of a neural network architecture* affect the performance of deep learning?

$\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*

- **Learning:**
  - Why does *stochastic gradient descent* converge to good local minima despite the non-convexity of the problem?

$\rightsquigarrow$ *Algebraic/Differential Geometry, Optimal Control, Optimization, ...*

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Mathematics for Deep Learning

- **Expressivity:**
  - Which *aspects of a neural network architecture* affect the performance of deep learning?

$\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*

- **Learning:**
  - Why does *stochastic gradient descent* converge to good local minima despite the non-convexity of the problem?

$\rightsquigarrow$ *Algebraic/Differential Geometry, Optimal Control, Optimization, ...*

- **Generalization:**
  - What is the *role of depth*?
  - Why do large neural networks *not overfit*?

$\rightsquigarrow$ *Learning Theory, Probability Theory, Statistics, ...*

Mathematics for Deep Learning

- **Expressivity:**
  - Which *aspects of a neural network architecture* affect the performance of deep learning?

  $\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*

- **Learning:**
  - Why does *stochastic gradient descent* converge to good local minima despite the non-convexity of the problem?

  $\rightsquigarrow$ *Algebraic/Differential Geometry, Optimal Control, Optimization, ...*

- **Generalization:**
  - What is the *role of depth*?
  - Why do large neural networks *not overfit*?

  $\rightsquigarrow$ *Learning Theory, Probability Theory, Statistics, ...*

- **Explainability:**
  - Why did a trained deep neural network *reach a certain decision*?
  - Which *features of data* are learned by deep architectures?

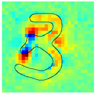
  $\rightsquigarrow$ *Information Theory, Uncertainty Quantification, ...*

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

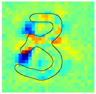
Explainability

Main Goal: We aim to *understand* decisions of "black-box" predictors!

map for digit 3



map for digit 8



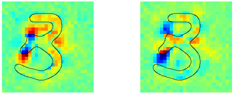
Selected Questions:

- What is relevance in a *mathematical sense*?
- What about a theory for *optimal relevance maps*?

Explainability

Main Goal: We aim to *understand* decisions of "black-box" predictors!

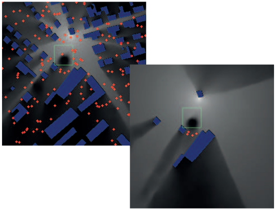
map for digit 3 map for digit 8



Selected Questions:

- What is relevance in a *mathematical sense*?
- What about a theory for *optimal relevance maps*?

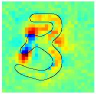
Rate-Distortion Explanation & CartoonX *(Kolek, Nguyen, Levie, Bruna, K; 2021):*



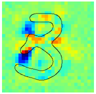
Explainability

Main Goal: We aim to *understand* decisions of "black-box" predictors!

map for digit 3



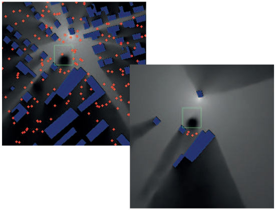
map for digit 8



Selected Questions:

- What is relevance in a *mathematical sense*?
- What about a theory for *optimal relevance maps*?

Rate-Distortion Explanation & CartoonX *(Kolek, Nguyen, Levie, Bruna, K; 2021):*



Vision for the Future:

Human-like answer to any question about a decision!

Deep Learning for Mathematical Problem Settings

- **Inverse Problems:**
  - How do we *optimally combine* deep learning with model-based approaches?

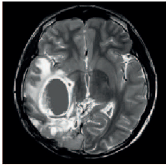
Deep Learning for Mathematical Problem Settings

- **Inverse Problems:**
  - How do we *optimally combine* deep learning with model-based approaches?
- **Partial Differential Equations:**
  - Why do neural networks perform well in *very high-dimensional environments*?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

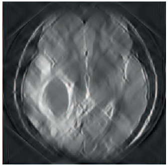
Deep Learning for Mathematical Problem Settings

- **Inverse Problems:**
  - How do we *optimally combine* deep learning with model-based approaches?
- **Partial Differential Equations:**
  - Why do neural networks perform well in *very high-dimensional environments*?
  - Are neural networks capable of *replacing highly specialized numerical algorithms* in natural sciences?

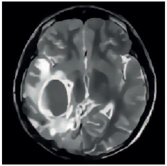
Deep Microlocal Reconstruction *(Andrade-Loarca, K, Öktem, Petersen; 2022):*



Original



Sparse Regularization/Shearlets

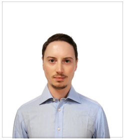


Deep Microlocal Reconstruction

Let's now consider Graph Neural Networks

Collaborators:

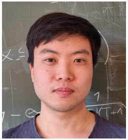
Michael Bronstein
(Imperial College London)



Lorenzo Bucci
(U. della Svizzera italiana)

Wei Huang
(U. della Svizzera italiana)

Ron Levie
(LMU Munich/Technion)

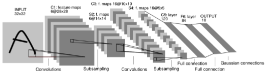


Yunseok Lee
(LMU Munich)

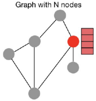
Sohir Maskey
(LMU Munich)

Some Facts about Graph Convolutional Neural Networks

Graph convolutional neural networks generalize classical CNNs to signals over graph domains. [Sperduti, Starita; 1997], [Gori, Monfardini, Scarselli; 2005], [Bruna, Zaremba, Szlam, LeCun; 2013], [Masci, Boscaini, Bronstein, Vandergheynst; 2015], ...



LeCun et al. 1989

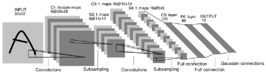


Graph signal: s : graph nodes $\to \mathbb{R}^c$

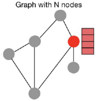
Graph CNN: graph signal $\to$ convolution $\to$ activation $\to$ pooling $\to \ldots$

Some Facts about Graph Convolutional Neural Networks

Graph convolutional neural networks generalize classical CNNs to signals over graph domains. [Sperduti, Starita; 1997], [Gori, Monfardini, Scarselli; 2005], [Bruna, Zaremba, Szlam, LeCun; 2013], [Masci, Boscaini, Bronstein, Vandergheynst; 2015], ...



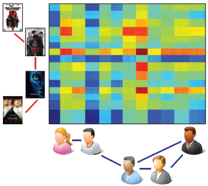
LeCun et al. 1989



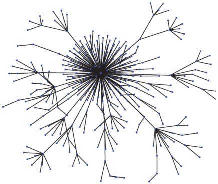
Graph signal: s : graph nodes $\to \mathbb{R}^c$

Graph CNN: graph signal $\to$ convolution $\to$ activation $\to$ pooling $\to \ldots$

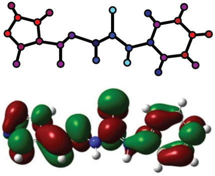
Some Applications:



Recommender system

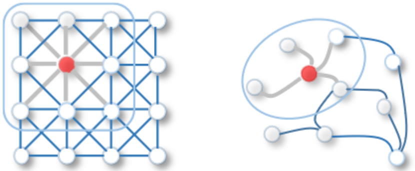


Fake news detection



Chemistry

Two Approaches to Convolution on Graphs



Spatial Approaches:

- Sliding window
- Aggregating feature information from the neighbors of each node

Spectral Approaches:

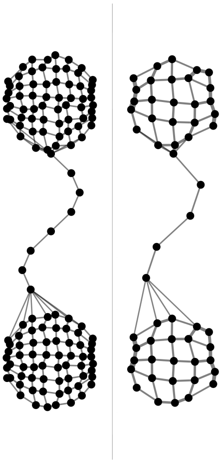
- Convolution theorem
- Defined in frequency domain
- Filter = multiplication in the frequency domain

Transferability of Spectral-based GCNNs

A Special Form of Generalization Capability

Desirable Feature:

Graph convolutional neural networks should *generalize* to graphs and signals unseen in the training set.



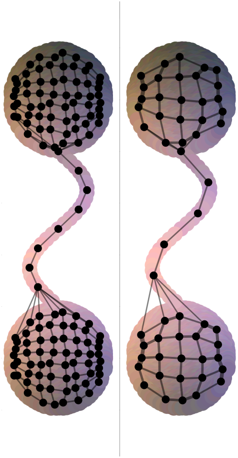
A Special Form of Generalization Capability

Desirable Feature:

Graph convolutional neural networks should *generalize* to graphs and signals unseen in the training set.

The Concept of Transferability:

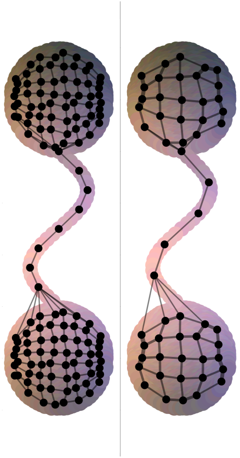
If two graphs *model the same phenomenon*, a fixed filter/Graph CNN should have approximately the same repercussion on both graphs.



A Special Form of Generalization Capability

Desirable Feature:

Graph convolutional neural networks should *generalize* to graphs and signals unseen in the training set.



The Concept of Transferability:

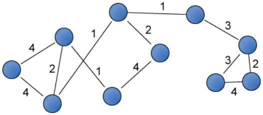
If two graphs *model the same phenomenon*, a fixed filter/Graph CNN should have approximately the same repercussion on both graphs.

We prove transferability for spectral graph filters/Graph CNNs!

Graph Theory

Notation:

We will in the following consider *undirected weighted graphs* $G = \{V, E, W\}$, where



- $V = \{1, \ldots, N\}$ are the *vertices*,
- $E \subset V^2$ are the *edges*,
- W is the *adjacency matrix*, i.e.,

$$w_{i,j} = 0, \quad \text{if} \quad (i,j) \notin E,$$
$$w_{i,j} > 0, \quad \text{if} \quad (i,j) \in E,$$

- the *degree matrix* is given by

$$D = \mathrm{diag}\left\{\sum_{j \neq i} w_{i,j}\right\}_{i=1}^{N}.$$

Graph Laplacian: Oscillations on Graphs

Definition: Let D be the degree matrix and W the adjacency matrix. Then the *unnormalized Graph Laplacian* is defined by

$$\Delta_u = D - W$$

and the *normalized Graph Laplacian* is given by

$$\Delta_n = D^{-1/2} \Delta_u D^{-1/2}.$$

As a generic notation, we will in the following use Δ.

Graph Laplacian: Oscillations on Graphs

Definition: Let D be the degree matrix and W the adjacency matrix. Then the *unnormalized Graph Laplacian* is defined by

$$\Delta_u = D - W$$

and the *normalized Graph Laplacian* is given by

$$\Delta_n = D^{-1/2} \Delta_u D^{-1/2}.$$

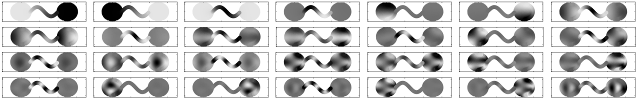
As a generic notation, we will in the following use Δ.

Remark: The Graph Laplacian Δ is self-adjoint. We will denote its

- eigenvalues by $\{\lambda_j\}_j \rightsquigarrow$ *Frequencies*,
- eigenvectors by $\{u_j\}_j \rightsquigarrow$ *Fourier modes*.

The graph Laplacian Δ encapsulates the geometry of the graph!

Graph Laplacian Eigenvectors



Spectral Graph Convolution

Definition:

Letting $\{u_j\}_j$ denote the eigenvectors of the graph Laplacian, we define the *spectral graph convolution operator* by

$$Cf = \sum_j c_j \langle f, u_j \rangle u_j.$$

Spectral Graph Convolution

Definition:

Letting $\{u_j\}_j$ denote the eigenvectors of the graph Laplacian, we define the *spectral graph convolution operator* by

$$Cf = \sum_j c_j \langle f, u_j \rangle u_j.$$

Problem with the Implementation:

- Computationally demanding
  - Eigendecomposition is slow.
  - No general FFT for graphs.
- Not transferable
  - The eigendecomposition is not stable to graph perturbations.
  - A fixed filter has different repercussions on similar graphs.

Spectral Graph Convolution

Definition:

Letting $\{u_j\}_j$ denote the eigenvectors of the graph Laplacian, we define the *spectral graph convolution operator* by

$$Cf = \sum_j c_j \langle f, u_j \rangle u_j.$$

Problem with the Implementation:

- Computationally demanding
  - Eigendecomposition is slow.
  - No general FFT for graphs.
- Not transferable
  - The eigendecomposition is not stable to graph perturbations.
  - A fixed filter has different repercussions on similar graphs.

Solution: Implement convolution using functional calculus!

Functional Calculus

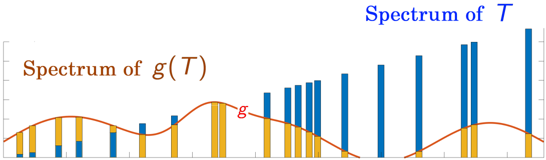
Definition:

Let T be a self-adjoint operator with discrete spectrum

$$Tv = \sum_j \lambda_j \langle v, u_j \rangle u_j.$$

A function $g : \mathbb{R} \to \mathbb{C}$ of T is then defined via

$$g(T)v = \sum_j g(\lambda_j) \langle v, u_j \rangle u_j.$$



Functional Calculus

Definition:

Let T be a self-adjoint operator with discrete spectrum

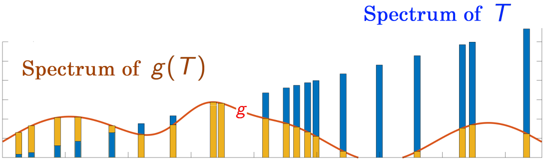
$$Tv = \sum_j \lambda_j \langle v, u_j \rangle u_j.$$

A function $g : \mathbb{R} \to \mathbb{C}$ of T is then defined via

$$g(T)v = \sum_j g(\lambda_j) \langle v, u_j \rangle u_j.$$

Remark:

If $g(\lambda) = \frac{\sum_{l=0}^{L} c_l \lambda^l}{\sum_{l=0}^{L} d_l \lambda^l}$, then $g(T) = \left(\sum_{l=0}^{L} c_l T^l\right) \left(\sum_{l=0}^{L} d_l T^l\right)^{-1}$.



Spectral Filtering using Functional Calculus

Functional Calculus Filters:

The functional calculus for $g : \mathbb{R} \to \mathbb{C}$ applied to the graph Laplacian yields

$$g(\Delta)f = \sum_j g(\lambda_j) \langle f, u_j \rangle u_j.$$

Recall:

The previous implementation used

$$Cf = \sum_j c_j \langle f, u_j \rangle u_j.$$

Spectral Filtering using Functional Calculus

Functional Calculus Filters:

The functional calculus for $g : \mathbb{R} \to \mathbb{C}$ applied to the graph Laplacian yields

$$g(\Delta)f = \sum_j g(\lambda_j) \langle f, u_j \rangle u_j.$$

Recall:

The previous implementation used

$$Cf = \sum_j c_j \langle f, u_j \rangle u_j.$$

Advantages of Functional Calculus Viewpoint:

This approach...

- *...solves the instability problem* (Levie, Isufi, K; 2019).
- *...solves the computational problem*, if g is a rational function.

Towards Transferability

LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN

Three Approaches to Transferability

Stability under Perturbation [Levie, Isufi, K; 2019], [Kenlay, Thanou, Dong; 2021]**:**

- Two graphs which are small perturbations of each other.

Topological Space Sampling [Levie, Huang, Bucci, Bronstein, K; 2019],[Keriven, Bietti, Vaiter; 2020]**:**

- Two graphs which sample the same underlying continuous space.

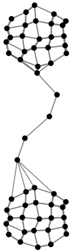
Graphon Approach [Ruiz, Chamon, Ribeiro; 2020], [Maskey, Levie, K; 2021]**:**

- Two graphs that come from the same sequence that converges to a graphon in a homomorphism density sense.

Topological Space Sampling

Interpretation:

- Weighted graphs:
  $\rightsquigarrow$ *Points and strength of correspondence between pairs of points.*



Topological Space Sampling

Interpretation:

- Weighted graphs:
  $\rightsquigarrow$ *Points and strength of correspondence between pairs of points.*
- Metric spaces:
  $\rightsquigarrow$ *Points and distances.*

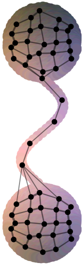
Interpretation:

- Weighted graphs:
  $\rightsquigarrow$ *Points and strength of correspondence between pairs of points.*
- Metric spaces:
  $\rightsquigarrow$ *Points and distances.*

Our Viewpoint:

Think of graphs as discretizations of metric spaces

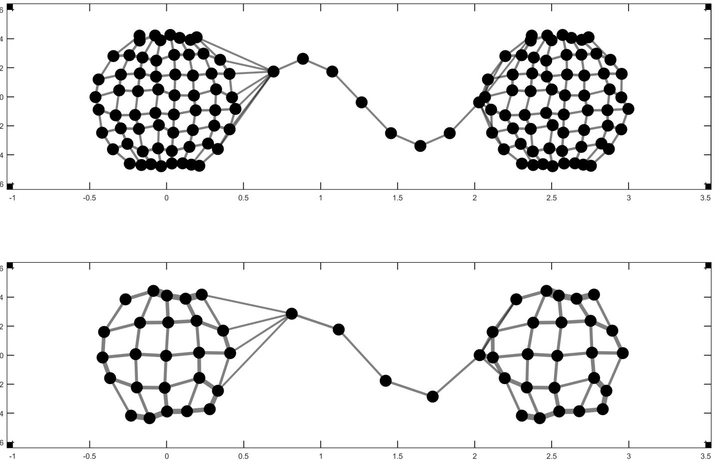
$$\text{distance} \nearrow \iff \text{edge weight} \searrow$$



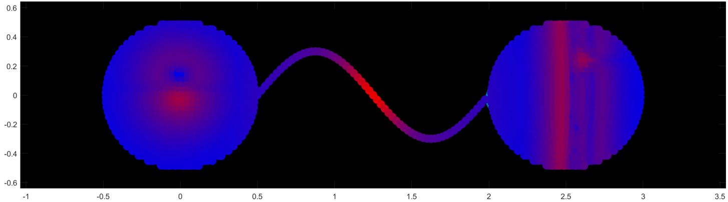
Graphs that represent the same phenomenon are discretizations of the same metric space!

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

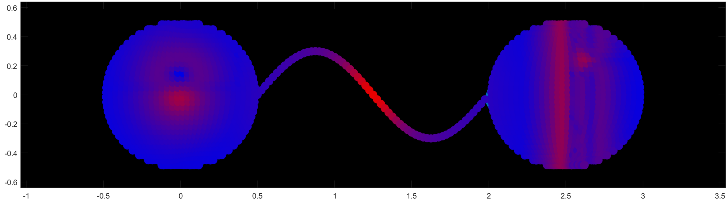
Comparing the Repercussion of a Filter on Two Graphs



Comparing the Repercussion of a Filter on Two Graphs

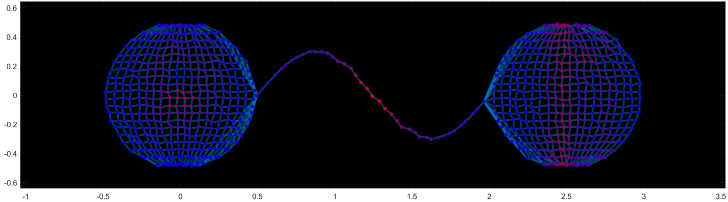


Take a generic signal $f : \mathcal{M} \to \mathbb{C}$

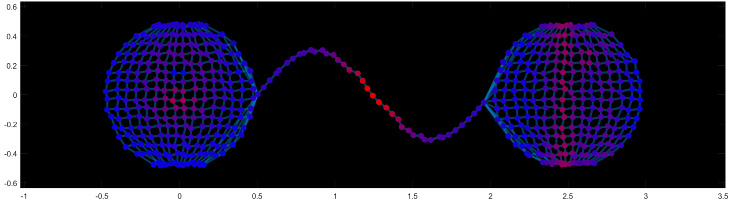


LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

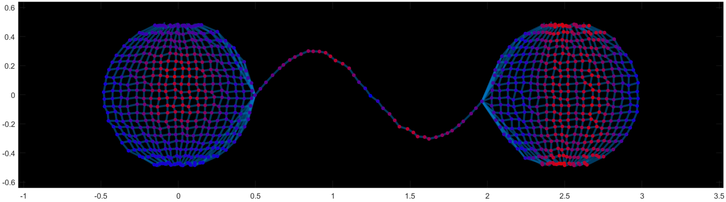
Comparing the Repercussion of a Filter on Two Graphs



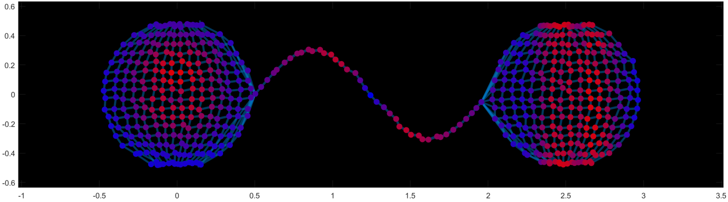
Sample to both graphs $\quad S_1 f : G_1 \to \mathbb{C}, \quad S_2 f : G_2 \to \mathbb{C}$



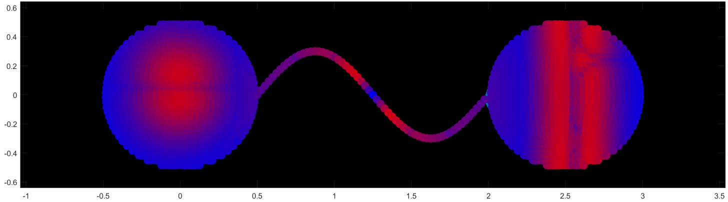
Comparing the Repercussion of a Filter on Two Graphs



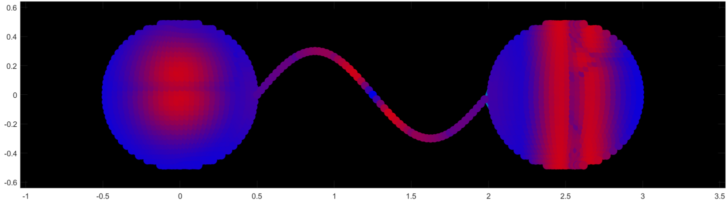
Apply both graph filters $g(\mathbf{\Delta}_1)S_1 f, \quad g(\mathbf{\Delta}_2)S_2 f$



Comparing the Repercussion of a Filter on Two Graphs



Interpolate back to $L^2(\mathcal{M})$ to get $\|R_1 g(\boldsymbol{\Delta}_1) S_1 f - R_2 g(\boldsymbol{\Delta}_2) S_2 f\| \approx 0$

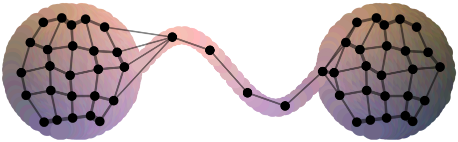


DSP Framework akin to the Nyquist–Shannon Approach

Our New Setting:

- *Analogue domain*: Borel space $\mathcal{M}$, with Laplacian $\mathcal{L}$.
- *Digital domains*: Graphs G with graph Laplacians Δ.
- *Paley Wiener spaces*: Band-limited spaces corresponding to $\mathcal{L}$.
- *Sampling operators*: $S^\lambda : PW(\lambda) \to L^2(G)$.
- *Interpolation operator*:

$$R^\lambda := (S^\lambda P(\lambda))^* := (S^\lambda P_{PW(\lambda)})^* : L^2(G) \to PW(\lambda).$$



What is Transferability precisely?

Definition:
The *transferability error of the filter f* on the signal $s \in L^2(\mathcal{M})$, is now defined by

$$\|f(\mathcal{L})s - R^\lambda f(\Delta) S^\lambda s\|,$$

the *transferability error of the Laplacian* is defined by

$$\|\mathcal{L}s - R^l \Delta S^\lambda s\|,$$

and the *consistency error* is defined by

$$\|s - R^\lambda S^\lambda s\|.$$

What is Transferability precisely?

Definition:
The *transferability error of the filter f* on the signal $s \in L^2(\mathcal{M})$, is now defined by

$$\|f(\mathcal{L})s - R^\lambda f(\Delta) S^\lambda s\|,$$

the *transferability error of the Laplacian* is defined by

$$\|\mathcal{L}s - R^l \Delta S^\lambda s\|,$$

and the *consistency error* is defined by

$$\|s - R^\lambda S^\lambda s\|.$$

(Informal Version) Theorem (Levie, Huang, Bucci, Bronstein, K; 2020):

Transferability of Filter

$\leq$ *Transferability of Laplacian* $+$ *Consistency Error*

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Transferability of Functional Calculus GCNNs

Theorem (Levie, Huang, Bucci, Bronstein, K; 2020):
Consider two graphs G_j, $j = 1, 2$ and two graph Laplacians Δ_j, $j = 1, 2$, approximating the same Laplacian $\mathcal{L}$ in $\mathcal{M}$, and consider a ReLU graph CNN with Lipschitz filters. Further, let $G_{j,l}$ be the graph in layer l with graph Laplacians $\Delta_{j,l}$. Also, assume that, for all layers l, bands λ_l, and $j = 1, 2$,

$$\|S_{j,l}^{\lambda_l} \mathcal{L} P(\lambda_l) - \Delta_{j,l} S_{j,l}^{\lambda_l} P(\lambda_l)\| \leq \delta$$

and

$$\|P(\lambda_L) - R_{j,L}^{\lambda_L} S_{j,L}^{\lambda_L} P(\lambda_L)\| \leq \delta$$

for some $0 < \delta < 1$. Then, for all output-channels k and mappings $\Phi_{j,L}^k$ given by the graph CNN,

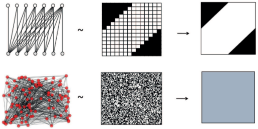
$$\begin{aligned} &\|R_{1,L}^{\lambda_L} \Phi_{1,L}^k S_{1,1}^{\lambda_0} P(\lambda_0) - R_{2,L}^{\lambda_L} \Phi_{2,L}^k S_{2,1}^{\lambda_0} P(\lambda_0)\| \\ &\qquad \leq 2\Big(LD\sqrt{\dim(PW(\lambda))} + L + 1\Big)\delta. \end{aligned}$$

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Further Results on Generalization Ability of GNNs

Graph Convolutional Neural Networks:

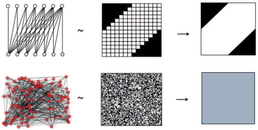
- *Similar results on transferability* for the *graphon* setting (Maskey, Levie, K; 2021).
- This builds on (Ruiz, Wang, Ribeiro; 2021).



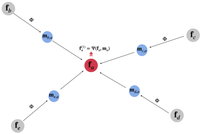
Further Results on Generalization Ability of GNNs

Graph Convolutional Neural Networks:

- *Similar results on transferability* for the *graphon* setting (Maskey, Levie, K; 2021).
- This builds on (Ruiz, Wang, Ribeiro; 2021).



Message Passing Graph Neural Networks:



- *Non-asymptotic generalization bounds*, only depending on the regularity of the network and space (Maskey, Levie, Lee, K; 2021).
- Builds on (Garg, Jegelka, Jaakkola; 2020), (Verma, Zhang; 2019), (Yehudai, Fetaya, Meirom, Chechik, Maron; 2022).

A Word of Caution: Computability Aspects

Collaborators:

Holger Boche
(TU Munich)

Adalbert Fono
(LMU Munich)

Problem with Computability

Problem with Computability

Computability on Digital Machines (informal):

A *computable problem (function)* is one for which the input-output relation can be computed on a digital machine for any given accuracy.

Problem with Computability

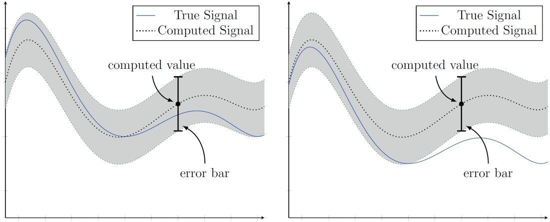
Computability on Digital Machines (informal):

A *computable problem (function)* is one for which the input-output relation can be computed on a digital machine for any given accuracy.

Theorem (Boche, Fono, K; 2022):

The solution of a finite-dimensional inverse problem is *not (Banach-Mazur/Turing-)computable* (by a deep neural network).

Illustration of the Problem:



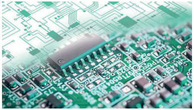
Some Thoughts on the Result

Remarks:

- *No algorithm exists*, which on digital hardware derives neural networks approximating the solution for any given accuracy.
- The output of trained neural networks *not reliable (no guarantees)*.
- This result could point towards why *instabilities* and *non-robustness* occurs for deep neural networks.

General Barrier:

- Limits of **computability** on today's hardware



What now?

Today computations performed almost exclusively on digital hardware!

What now?

Today computations performed almost exclusively on digital hardware!

Other Models of Computations:

- New emerging hardware
  - *Neuromorphic computing*: Elements of computer modeled after systems in the human brain and nervous system.
  - *Biocomputing*: Living cells as the substrate for performing human-defined computations
- Different models of computation required

What now?

Today computations performed almost exclusively on digital hardware!

Other Models of Computations:

- New emerging hardware
  - *Neuromorphic computing*: Elements of computer modeled after systems in the human brain and nervous system.
  - *Biocomputing*: Living cells as the substrate for performing human-defined computations
- Different models of computation required

Key Future Question:

Does the non-computability result also hold for different computation models such as analog computers as well?

What now?

Today computations performed almost exclusively on digital hardware!

Other Models of Computations:

- New emerging hardware
  - *Neuromorphic computing*: Elements of computer modeled after systems in the human brain and nervous system.
  - *Biocomputing*: Living cells as the substrate for performing human-defined computations
- Different models of computation required

Key Future Question:

Does the non-computability result also hold for different computation models such as analog computers as well?

Theorem (Boche, Fono, K; 2022):
The solution of a finite-dim. inverse problem is *computable* (by a deep neural network) *on an analog machine!*

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Conclusions

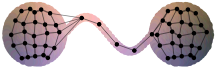
What to take Home...?

Deep Learning:

- *Stability* is a major concern!
- The amazing *generalization capability* is still a mystery!

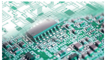
Transferability of Graph Convolutional Neural Networks:

- *Transferability* is a special type of generalization.
- We consider graphs as *discretizations of metric spaces*.
- We show *spectral GCNNs* (based on *functional calculus*) are transferable.
- Similar results: Graphs as *arising from a graphon*.



Generalization of Message Passing Graph Neural Networks:

- We consider graphs as *sampled from (continuous) models*.
- We derive *non-asymptotic* generalization bounds, *only depending on the regularity* of the network and space.



Caution: Problems with computability on digital hardware!

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

THANK YOU!

References available at:

`www.ai.math.lmu.de/kutyniok`

Survey Paper (arXiv:2105.04026):

Berner, Grohs, K, Petersen, *The Modern Mathematics of Deep Learning*.

Check related information on Twitter at:

`@GittaKutyniok`

Upcoming Book:

- Grohs and K, eds.
  Mathematical Aspects of Deep Learning
  Cambridge University Press, to appear.

Convergence of Δ to Metric-Measure Laplacians

Transferability of Filter
$\leq$ *Transferability of Laplacian* $+$ *Consistency Error*

Convergence of Δ to Metric-Measure Laplacians

Transferability of Filter

$\leq$ *Transferability of Laplacian* $+$ *Consistency Error*

Question:

Is it reasonable to assume that the transferability error of the Laplacian is small?

Convergence of Δ to Metric-Measure Laplacians

Transferability of Filter
$\leq$ *Transferability of Laplacian* $+$ *Consistency Error*

Question:

Is it reasonable to assume that the transferability error of the Laplacian is small?

Informal Statement (Levie, Huang, Bucci, Bronstein, K; 2020)**:**
If graphs are constructed by sampling random points from $\mathcal{M}$, then graph Laplacians Δ approximate the continuous Laplacian $\mathcal{L}$ with high probability $\Rightarrow$ *Transferability in high probability!*

Towards Transferability:

Graphon Approach

Graphons

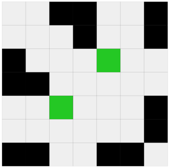
Definition:

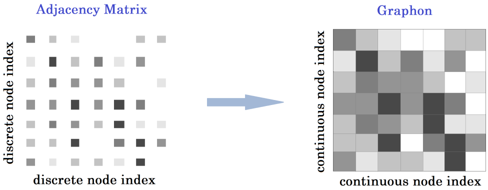
A *graphon* is a symmetric measurable function $W : [0,1]^2 \to [0,1]$.

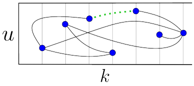
Intuition:

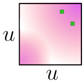
A graphon is understood as defining an exchangeable random graph model:

- Each vertex j of the graph is assigned an independent random value $x_j \sim U[0,1]$.
- Edge (i,j) is independently included in the graph with probability $W(x_i, x_j)$.



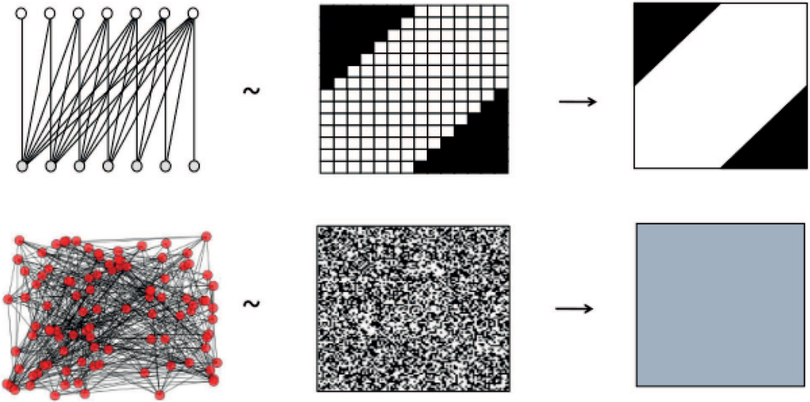






Examples of Graphons

Graphs, Empirical Graphons, and Limits:



Local Convergence: Homomorphism Density

Definition:
For F, G simple graphs, let $t(F, G)$ the probability that a random map $V(F) \to V(G)$ is a homomorphism. Then a sequence G_n is *convergent to a graphon W*, if

$$t(F, G_n) \to t(F, W) := \int_{[0,1]^{v(F)}} \prod_{i,j \in E} W(x_i, x_j) \prod_{i \in V} dx_i$$

for all simple graphs F. For a graph G, the *induced kernel W_G* is defined by

$$W_G(u, v) := \sum_{i,j \leq n} \Delta(i, j) \chi_{I_i}(u) \chi_{I_j}(v)$$

and the *Hilbert-Schmidt operator T_W* associated to a kernel W is given by

$$T_W \psi(v) := \int_0^1 W(u, v) \psi(u) du, \quad \psi \in L^2(0, 1).$$

$\rightsquigarrow$ *We can use functional calculus (filters)!*

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Transferability of Continuous Filters

Theorem (Maskey, Levie, K; 2021):
Let $(G_n)_n$ be a sequence of graphs with uniformly bounded Laplacians. Suppose that there exists a graphon W such that

$$G_n \to W$$

in homomorphism density. Let h be a continuous function. Then, there exists a sequence of permutations $(\pi_n)_n$ such that

$$h(T_{W_{\pi_n(G_n)}}) \to h(T_W)$$

in operator norm.

Numerical Results

Rethinking Transferability

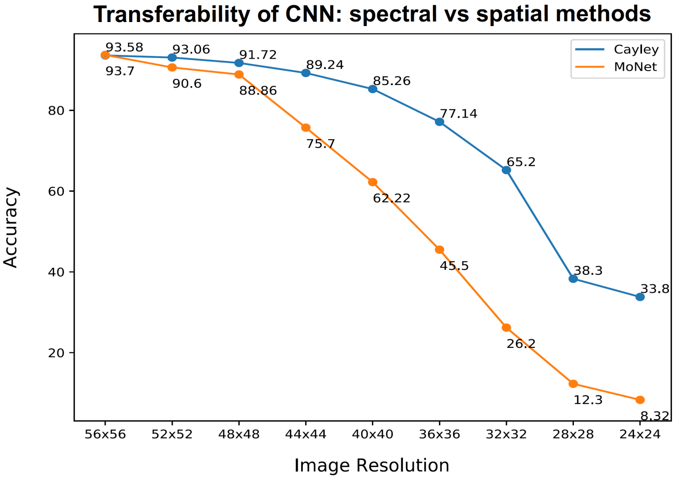
Graph CNNs can manage transferability in different ways!

- **Concept-Based Transferability:**
  - *Multi-graph training set*
  - The network learns "concepts" that promote transferability.
- **Principle Transferability:**
  - *Single or multi-graph training set*
  - A built-in capability of graph CNNs, independent of their specific filters, which requires no training.

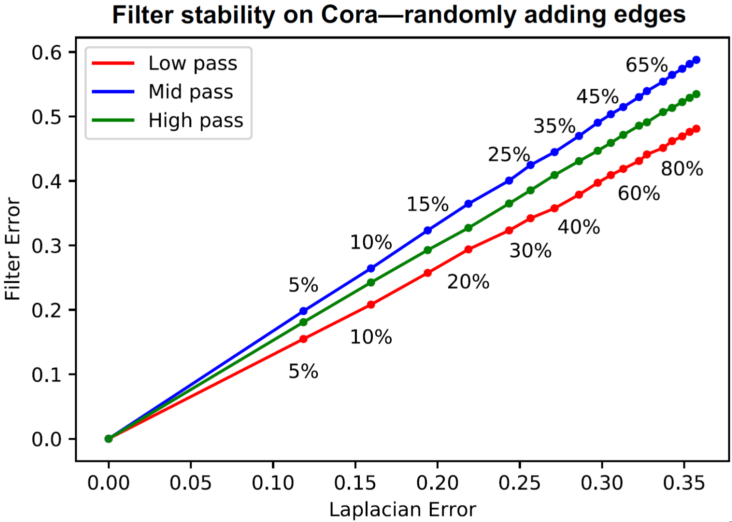
The success of spectral graph CNNs in multi-graph settings relies on both types of transferability!

Some Examples

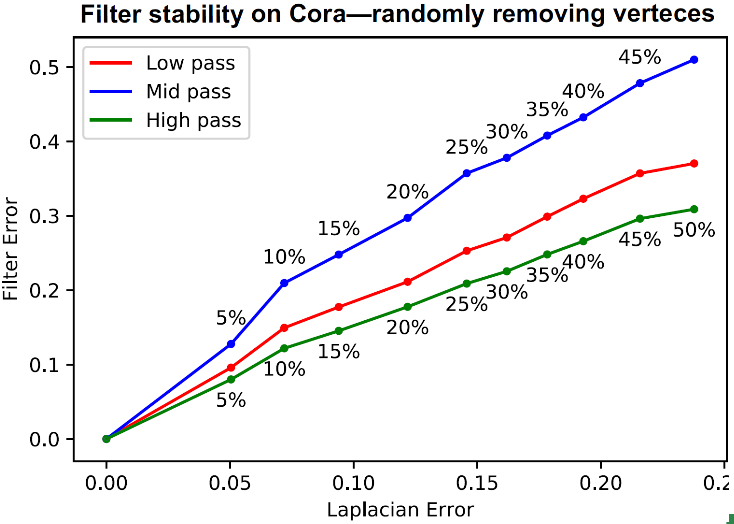
Isolate principle transferability from concept-based transferability:
Train the network on one single graph and test on other graphs.



Transferability under Graph Perturbation



Transferability under Graph Perturbation



An Experimental Study of Transferability

Spectral method were tested only in single-graph settings.

Benchmark ChebNet (Defferrard et al. 2016) in multi-graph settings:

- *Graph benchmarks*:

  Hu et al. Open Graph Benchmark: *Datasets for Machine Learning on Graphs.* 2020.

  Dwivedi et at. *Benchmarking Graph Neural Networks.* 2020.
- *Tasks*: graph regression, graph classification, node classification.
- *Rules*: different for each benchmark, e.g., budget of parameters, fixed number of layers, fixed hyperparameters, no specialized data augmentation techniques.

An Experimental Study of Transferability

Spectral method were tested only in single-graph settings.

Benchmark ChebNet (Defferrard et al. 2016) in multi-graph settings:

- *Graph benchmarks*:

  Hu et al. Open Graph Benchmark: *Datasets for Machine Learning on Graphs.* 2020.

  Dwivedi et at. *Benchmarking Graph Neural Networks.* 2020.
- *Tasks*: graph regression, graph classification, node classification.
- *Rules*: different for each benchmark, e.g., budget of parameters, fixed number of layers, fixed hyperparameters, no specialized data augmentation techniques.

$\rightarrow$ *ChebNet reaches state-of-the-art results (Nilsson, Bresson; 2020)*