TDS Archive · Following

A Brief Introduction to Geometric Deep Learning

AI for complex data

8 min read · Jul 26, 2022

Jason McEwen Following

Listen Share More

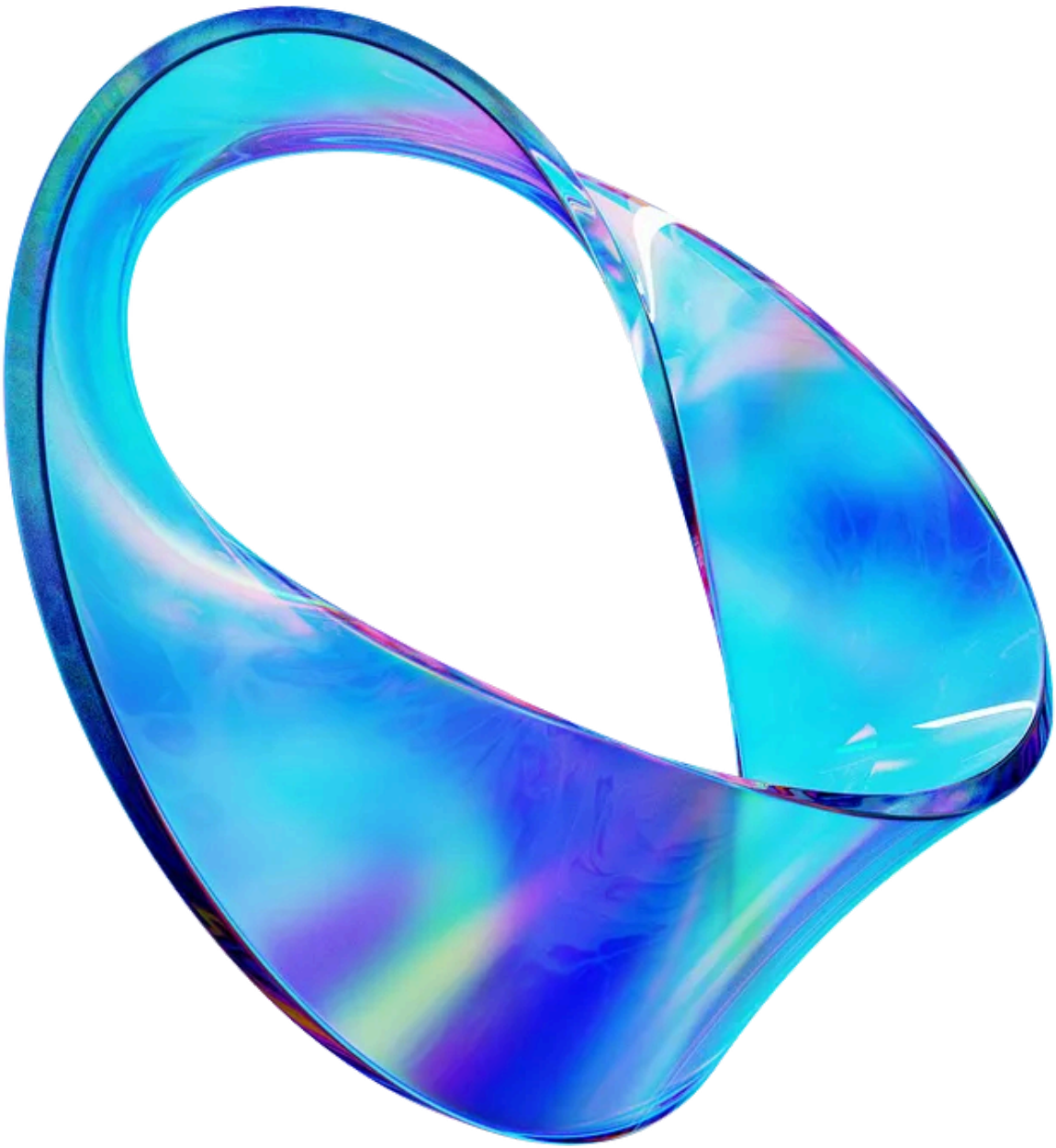


Photo by SIMON LEE on Unsplash

Deep learning is hard. While universal approximation theorems show that sufficiently complex neural networks can in principle approximate "anything", there is no guarantee that we can find good models.

Great progress in deep learning has nevertheless been made by judicious choice of model architectures. These model architectures encode inductive biases to give the model a helping hand. One of the most powerful inductive biases is to leverage notions of geometry, giving rise to the field of *geometric deep learning*.

The term geometric deep learning was first coined by Michael Bronstein, a pioneer of the field (see his posts for interesting insights on a lot of the latest deep learning research, as well as extensive overviews of the field). In this post, rather then getting deep into the technical weeds, we present a very brief introduction to geometric deep learning. We largely follow the excellent recent book by Bronstein and colleagues [1] but provide our own unique take, and focus on high level concepts rather than technical details.

Geometric Priors

> *Fundamentally, geometric deep learning invovles encoding a geometric understanding of data as an inductive bias in deep learning models to give them a helping hand.*

Our geometric understanding of the world is typically encoded through three types of *geometric priors*:

3. Multiscale representations

One of the most common geometric priors is to encode **symmetries and invariances** to different types of transformations. In physics, symmetries are typically expressed by the invariance of physical systems under transformations. If we know the real world exhibits certain symmetries, then it makes sense to encode those symmetries directly into our deep learning model. That way we are able to give the model a helping hard so that is does not have to learn the symmetry but in some sense already knows it. Harnessing symmetry in deep learning is elaborated further in our previous article on *What Einstein Can Teach Us About Machine Learning*.

As an example of encoding symmetries and invariances, traditional convolutional neural networks (CNNs) exhibit what is called translational equivariance, as illustrated in the diagram below of a cat's face. Consider the model's feature space (on the right). If the camera or cat moves, i.e. is translated in the image, content in

the feature space should more similarly, i.e. is also translated. This property is called translational equivariance and in a sense ensures a pattern (cat's face) need only be learnt once. Rather than having to learn the pattern in all possible locations, by encoding translational equivariance in the model itself we ensure the pattern can then be recognised in all locations.

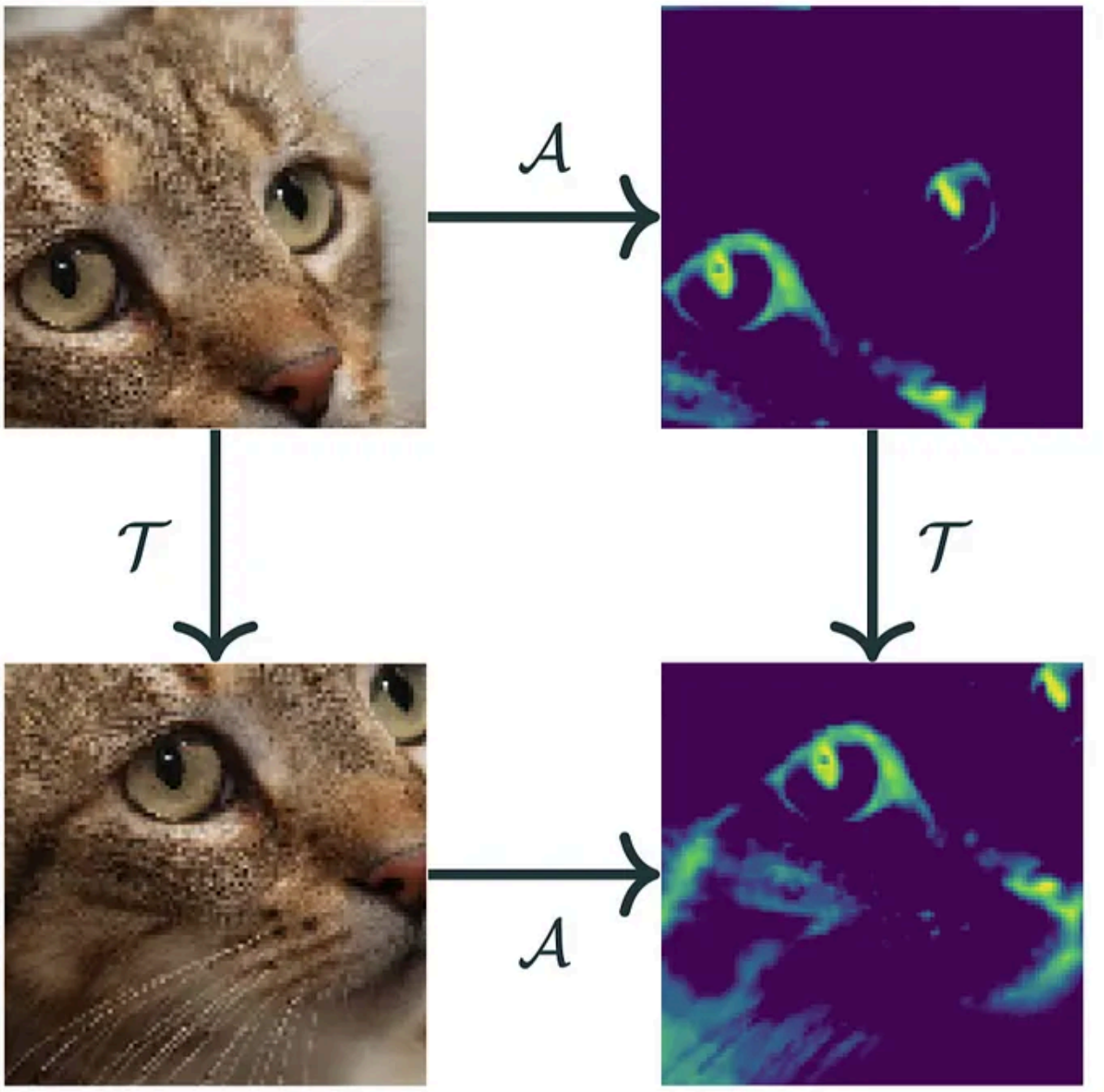


Illustration of translational equivariance. Given an image (top left), computing a feature map (by $\mathcal{A}$) (top right) and then translating ($\mathcal{T}$) the feature map (bottom right) is equivalent to first translating the image (bottom left) and then computing the feature map (bottom right). [Diagram created by authors, first presented here.]

Another common geometric prior is to ensure **stability** of the representation space. We can consider differences between data instances as due to some *distortion* that would map one data instance into another. For a classification problem, for

example, small distortions are responsible for variations within a class, whereas larger distortions can map data instances from one class to another. The size of the distortion between two data instances then captures how “close”, or similar, one data instance is to another. For a representation space to be well-behavied and support effective deep learning, we should preserve measures of similarlity between data instances. To preserve similarity in the representation space, feature mappings must exhibit a form a stability.

As a representative example consider the classification of hand-written digits. The original image space and its representation space are illustrated in the following diagram. Small distortions map one 6 into another, capturing intra-class variations between different instances of a hand drawn 6. In the representational space, these data instances should remain close. Larger distortions, however, can map a 6 into an 8, capturing inter-class variations. Again, in the representation space the measure of similarity should be preserved and so there should be a larger separation between 6s and 8s in the representation space. Stability of the feature mapping is required to ensure such distances are preserved to facilitate effective learning.

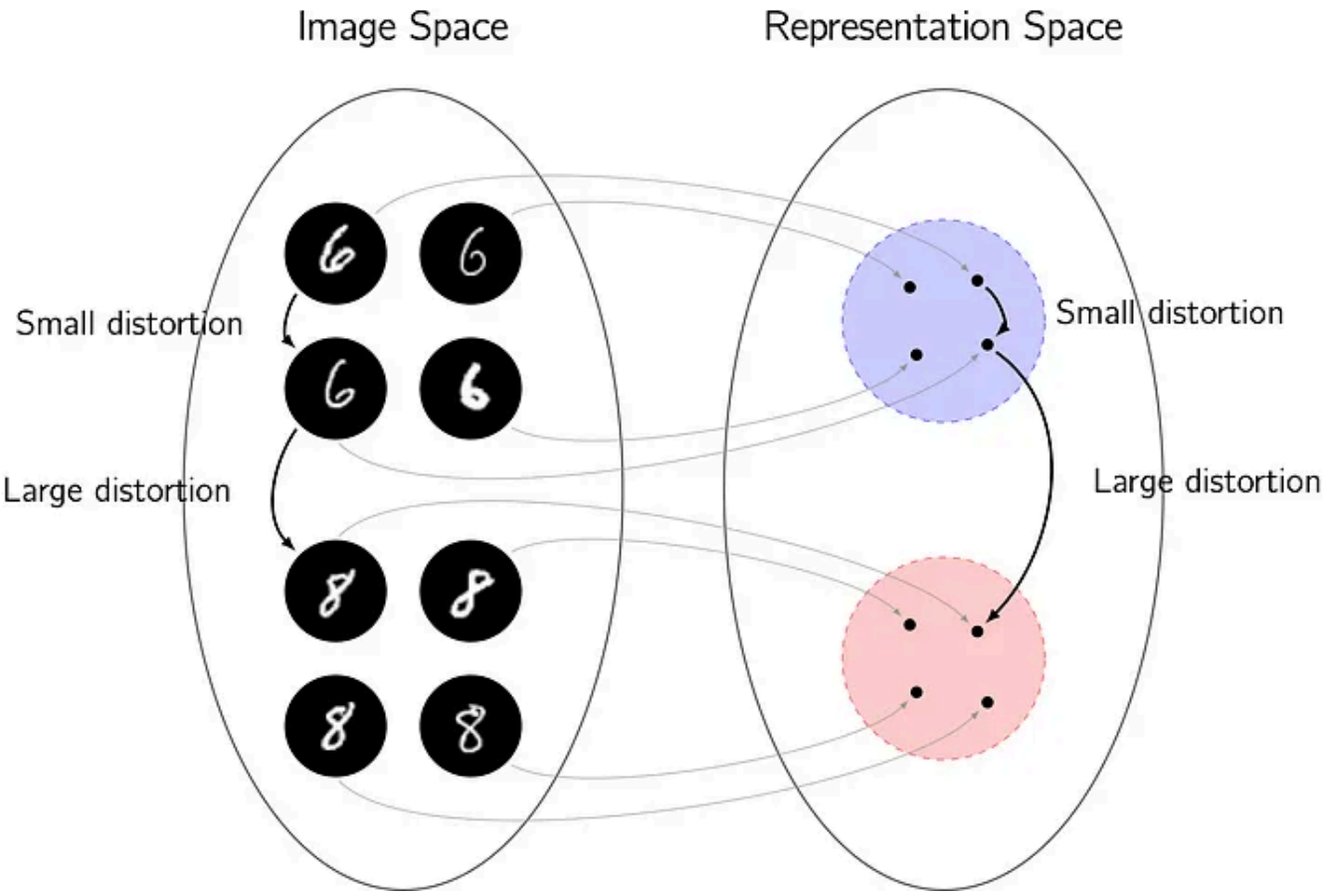
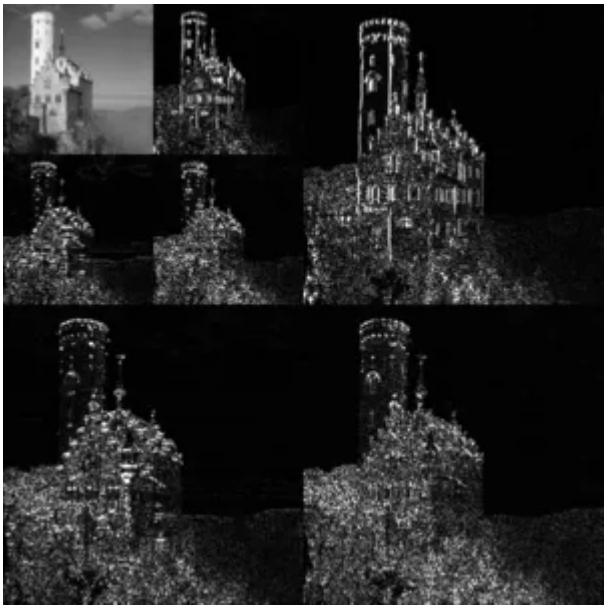


Illustration of stability of mapping to representation space. Small distortions are responsible for intra-class variations, whereas large distortions are responsible for inter-class variations. Stability of the mapping is required to ensure measures of similarity between data instances, i.e. the size of the distortion between

them, is preserved in the representation space in order to facilitate effective learning. [Diagram created by author for [2].]

A third common geometric prior is to encode a **multiscale, hierarchical** representation of data. In a data instance, many of the datum are *not* independent but are correlated in complex ways. Consider an image for example. Each image pixel is not independent but rather nearby pixels are often related and very similar. Different notions of "nearby" are also possible depending on content structure. Effective representational spaces can therefore be constructed by capturing the multiscale, hierarchical nature of much data.

Consider a standard 2D image as an example, such as the image of a castle shown below. The illustration below shows a multiscale, hierarchical representation of the image, with a low-resolution version of the original image in the top-left corner and then remaining image content at different resolutions captured in the other panels of the diagram. This provides a much more efficient representation of the underlying image and, in fact, is the technology powering JPEG-2000 image compression. Similar multiscale, hierarchical representations can be exploited to provide effective representational spaces for learning.

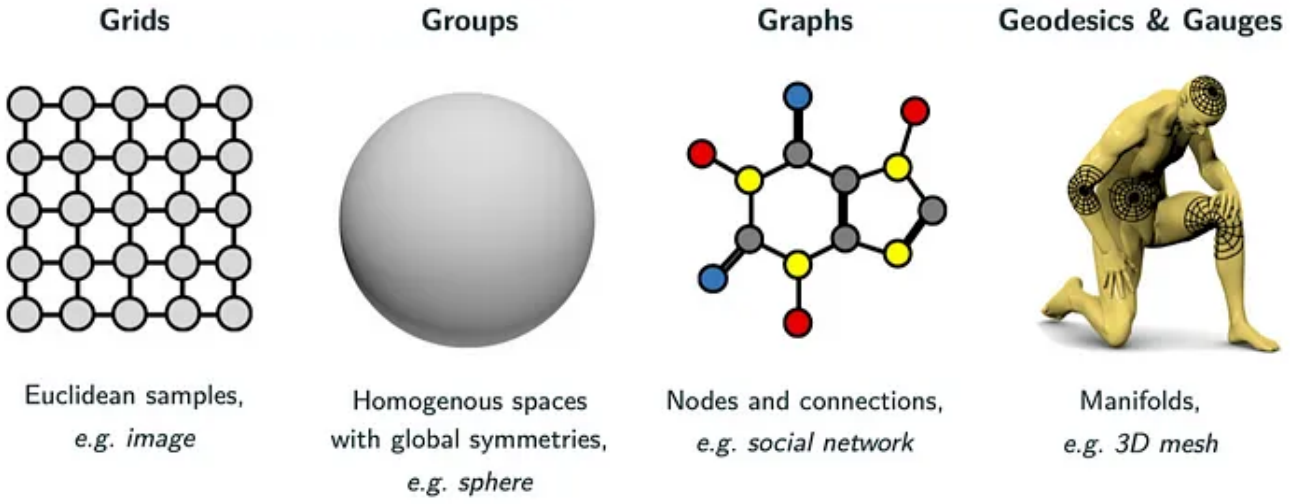


A multiscale, hierarchical representation of an image. A low-resolution version of the original image is shown in the top-left corner and then remaining image content at different resolutions is captured in the other panels of the diagram. Similar representations can be exploited to provide effective representational spaces for learning. [Source wikipedia.]

We have covered the three main types of geometric priors leveraged in geometric deep learning. While these provide the fundamental underlying concepts of geometric learning, they can be applied in a number of different settings.

Categories of Geometric Deep Learning

In Bronstein's recent book [1], geometric deep learning is classified into four fundamental categories, as illustrated in the diagram below.



Categories of geometric deep learning. [Image sourced from article [1], with permission, with annotated overview and examples added.]

Bronstein talks of the 5Gs (extending the 4G categorisation first introduced by Max Welling [1]): grids; groups; graphs; and geodesics and gauges. Since these final two Gs are closely related we consider just four different categories, i.e. 4Gs.

The **grid** category captures regularly sampled, or gridded, data such as 2D images. These data would perhaps typically be the purveyance of classical deep learning. However, it is also possible to interpret many of the classical deep learning models in a geometric perspective (such as CNNs an their translational equivariance, as discussed above).

The **group** category covers homogenous spaces with global symmetries. The canonical example of this category is the sphere (covered in greater detail in our previous article [3]). Spherical data arise in myrad applications, not only when data is acquired directly on the sphere (such as over the Earth or by 360° cameras that capture panoramic photos and videos), but also when considering spherical symmetries (such as in molecular chemistry or magnetic resonance imaging). While the sphere is the most common group setting, other groups and their corresponding symmetries can also be considered.

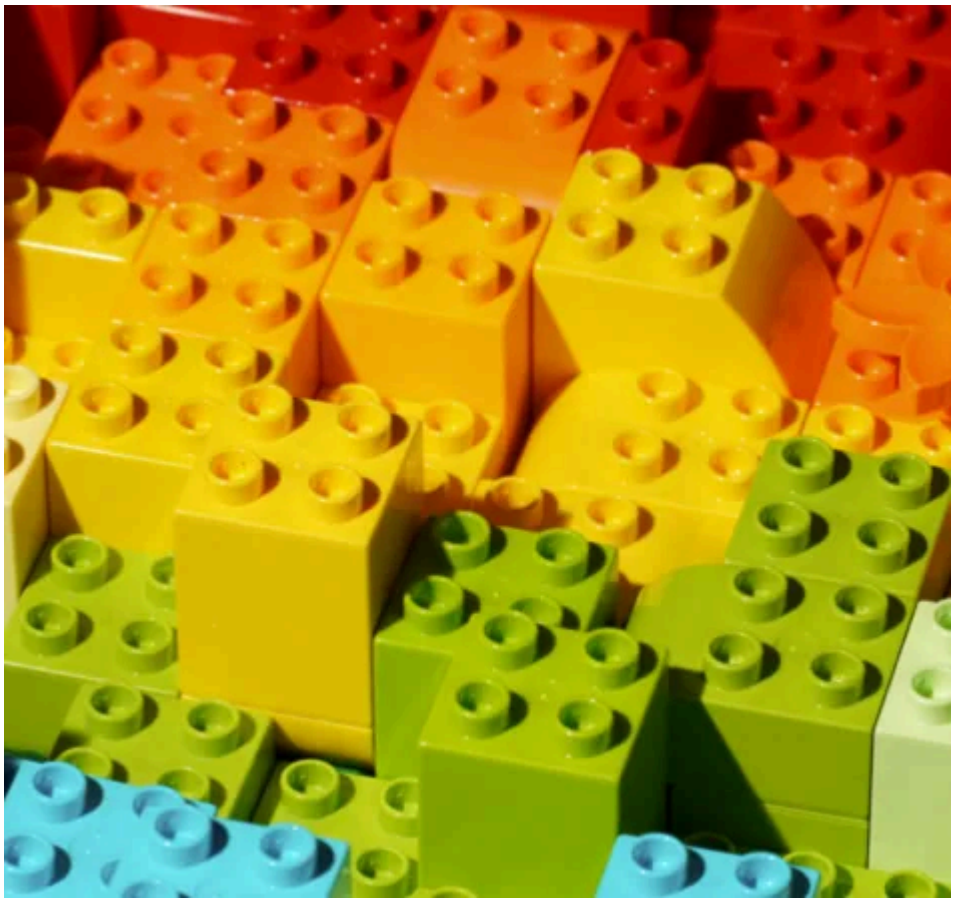
The **graph** category covers data that may be represented by a computational graph, with nodes and edges. Networks are well-suited to such representations, hence

graph deep learning has found wide application in the study of social networks. The graph approach to geometric deep learning provides great flexibility since much data can be represented by a graph. However, this flexibility can come with a loss in specificity and the advantages that affords. For example, the group setting can often be considered with a graph approach but in this case one loses the underlying knowledge of the group, which can otherwise be leveraged.

The final **geodesics and gauges** category involves deep learning on more complex shapes, such as more general maniolds and 3D meshes. Such approaches can be of great use in computer vision and graphics, for example, where one can perform deep learning with 3D models and their deformations.

Building Blocks

While there are a number of different categories of geometric deep learning, as described above, and different types of geometric priors than can be exploited, all approaches to geometric deep learning essentially adopt different incarnations of the following fundamental underlying building blocks.



All approaches to geometric deep learning leverage a core set of fundamental underlying building blocks. [Photo by Sen on Unsplash.]

Deep learning architectures are typically composed of a number of layers, that are combined together to form the overall model architecture. Often combinations of layers are then repeated. Geometric deep learning models typically include the following types of layers.

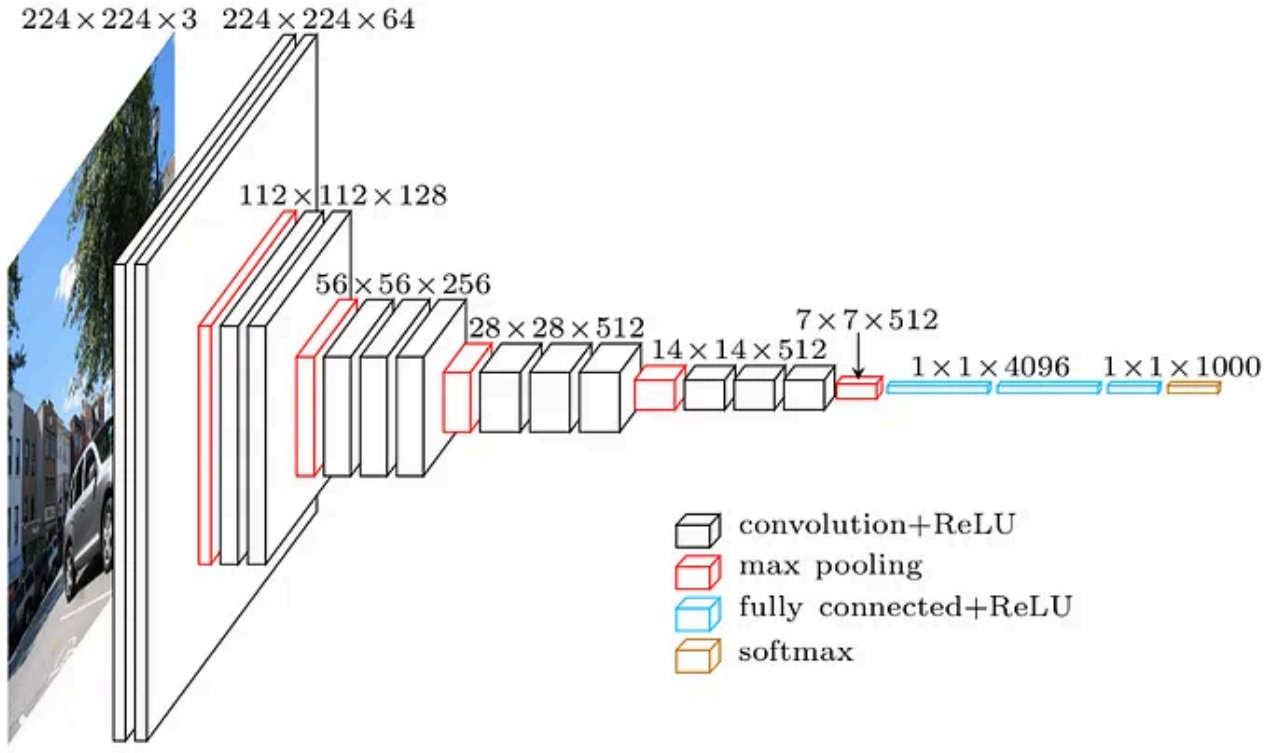
1. **Linear equivariant layers:** The core component of geometric deep learning models is linear layers, such as convolutions, that are equivariant to some symmetry transformation. The linear transform itself needs to be constructed for the geometric category considered, e.g. a convolution on the sphere and graph are difficult, although there are often many analogies.

2. **Non-linear equivariant layers:** To ensure deep learning models have sufficient representational power, they must exhibit non-linearity (otherwise they could only represent simple linear mappings). Non-linear layers must be introduced to achieve this, while also preserving equivariance. The canonical way to introduce non-linearity in an equivariant manner it to do so via pointwise non-linear activation functions (e.g. ReLUs), although other forms of non-linearity tailored specifically to the underlying geometry are sometimes considered [3].

3. **Local averaging:** Most geometric deep learning models also include a form of local averaging, such as max pooling layers in CNNs. Such operations impose local invariances at certain scales, ensuring stability and leading to multi-scale, hierarchical representations by stacking multiple blocks of layers.

4. **Global averaging:** To impose global invariances in geometric deep learning models, global averging layers are often employed, such as global pooling layers in CNNs.

The canonical example of a geometric deep learning model is a tranditional CNN for 2D planar images. While many may consider this as a classical deep learning model, it can be interpreted in a geometric perspective. Indeed, one of the key reasons CNNs have been so successful is due to the geometric properties encoded in their architecture. The following diagram outlines a typical CNN architecture, where it is clear many of the geometric deep learning layers discussed above are included, with blocks of layers repeated to provide a hierarchical, multiscale representational space.



VGG-16 convolutional neural network (CNN) architecture. Although CNNs are typically considered as classifical deep learning models, they can be interpreted in a geometric perspective leveraging the core types of layers of geometric deep learning models. [Image source.]

Future perspectives

Deep learning is now commonplace for standard types of data, such as structured, sequential and image data. However, to exend the application of deep learning to other more complex — geometric — datasets, the geometry of such data must be encoded in deep learning models, giving rise to the field of geometric deep learning.

Geometric deep learning is a topical and rapidly evolving field, where much progress has been made. However, many unsolved questions remain, not only in models themselves but also around scalability and practical application. We will address these issuses in upcoming articles, showing how solving such issues is critical to unlocking the remarkable potential of deep learning for a host of new applications.

References

[1] Bronstein, Bruna, Cohen, Velickovic, *Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges* (2021), arXix:2104.13478

[2] McEwen, Wallis, Mavor-Parker, *Scattering Networks on the Sphere for Scalable and Rotationally Equivariant Spherical CNNs*, ICLR (2022), arXiv:2102.02828

[3] Cobb, Wallis, Mavor-Parker, Marignier, Price, d'Avezac, McEwen, *Efficient Generalised Spherical CNNs*, ICLR (2021), arXiv:2010.11661

Thoughts And Theory | Geometric Deep Learning | Deep Learning | Artificial Intelligence | Machine Learning

Data Science

Following

Published in TDS Archive

822K followers · Last published Feb 4, 2025

An archive of data science, data analytics, data engineering, machine learning, and artificial intelligence writing from the former Towards Data Science Medium publication.

Following

Written by Jason McEwen

295 followers · 14 following

Professor of Astrostatistics, UCL | Founder & CEO, CopernicAI

Responses (2)

Yogesh Haribhau Kulkarni (PhD)

What are your thoughts?

Ev Yemini
Jul 30, 2022

Very interesting article on an exciting new topic in DL. Thank you for taking the time to write it.

3 Reply

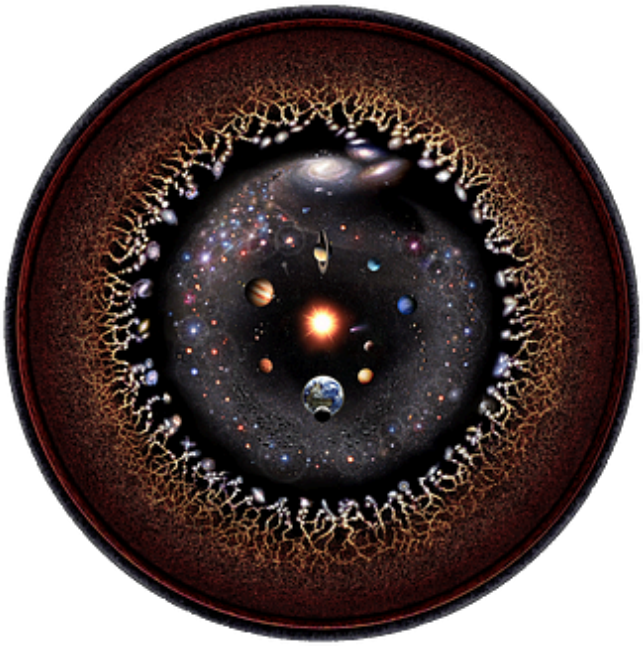
Noam C
Jun 12, 2023

learned a lot!

It's a pity this post did not go through proof-reading. There are a dozen errors I could spot

Reply

More from Jason McEwen and TDS Archive

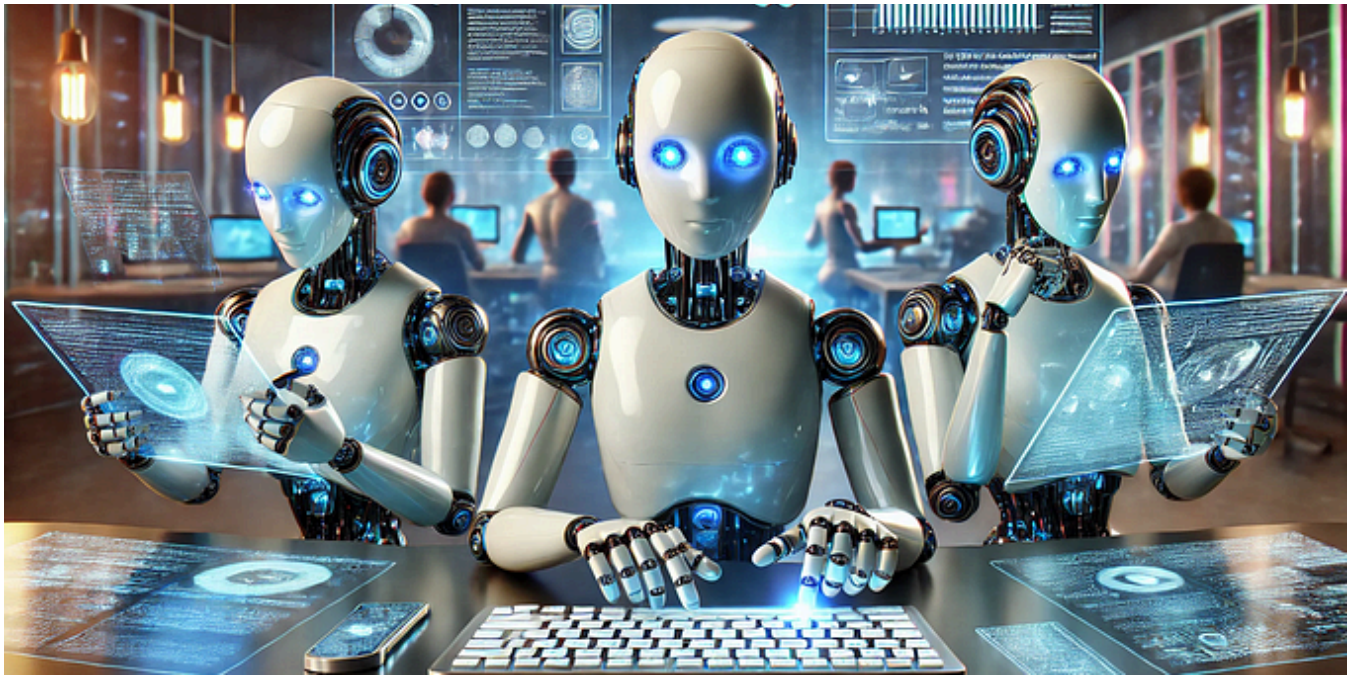


In TDS Archive by Jason McEwen

Differentiable and Accelerated Spherical Harmonic Transforms

In JAX and PyTorch

Mar 14, 2024 89 1

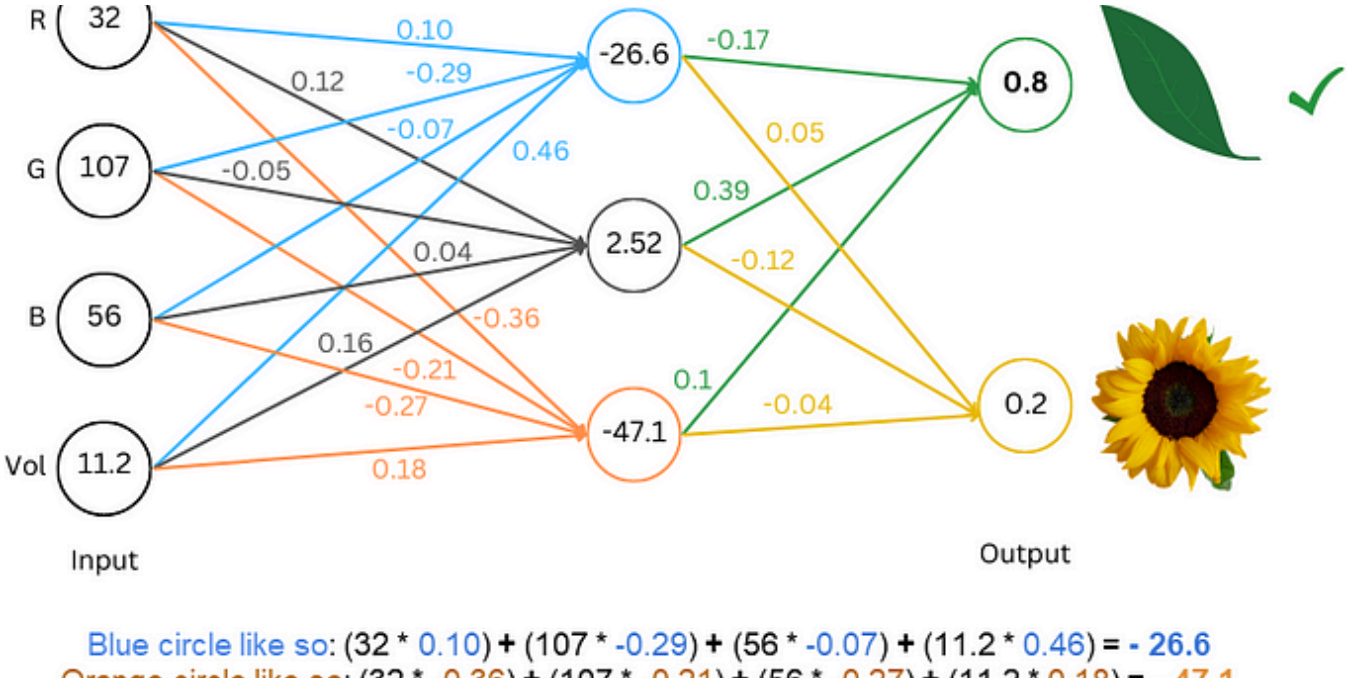


In TDS Archive by Luís Roque

Agentic AI: Building Autonomous Systems from Scratch

A Step-by-Step Guide to Creating Multi-Agent Frameworks in the Age of Generative AI

Dec 13, 2024 1.1K 25



In TDS Archive by Rohit Patel

Understanding LLMs from Scratch Using Middle School Math

In this article, we talk about how LLMs work, from scratch—assuming only that you know how to add and multiply two numbers. The article...

Oct 20, 2024 7.3K 90

In TDS Archive by Jason McEwen

Geometric Deep Learning for Spherical Data

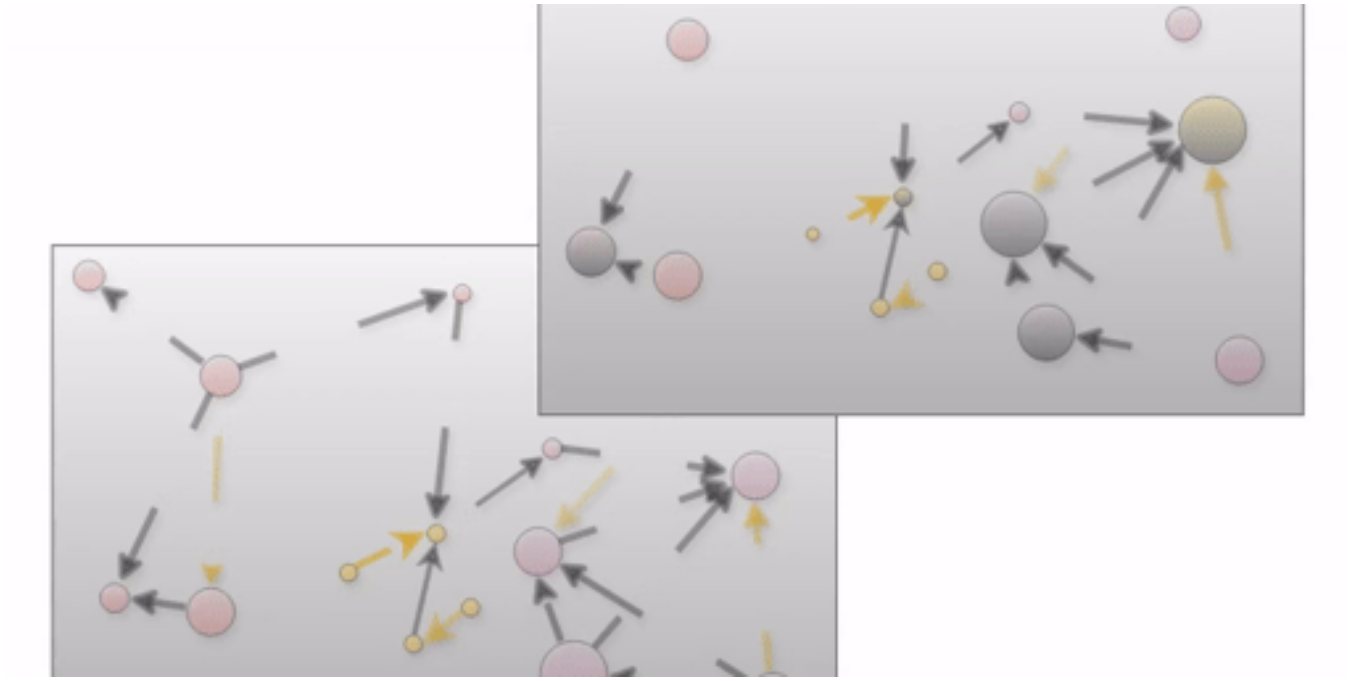
Spherical CNNs

Apr 26, 2022 35 1

See all from Jason McEwen

See all from TDS Archive

Recommended from Medium

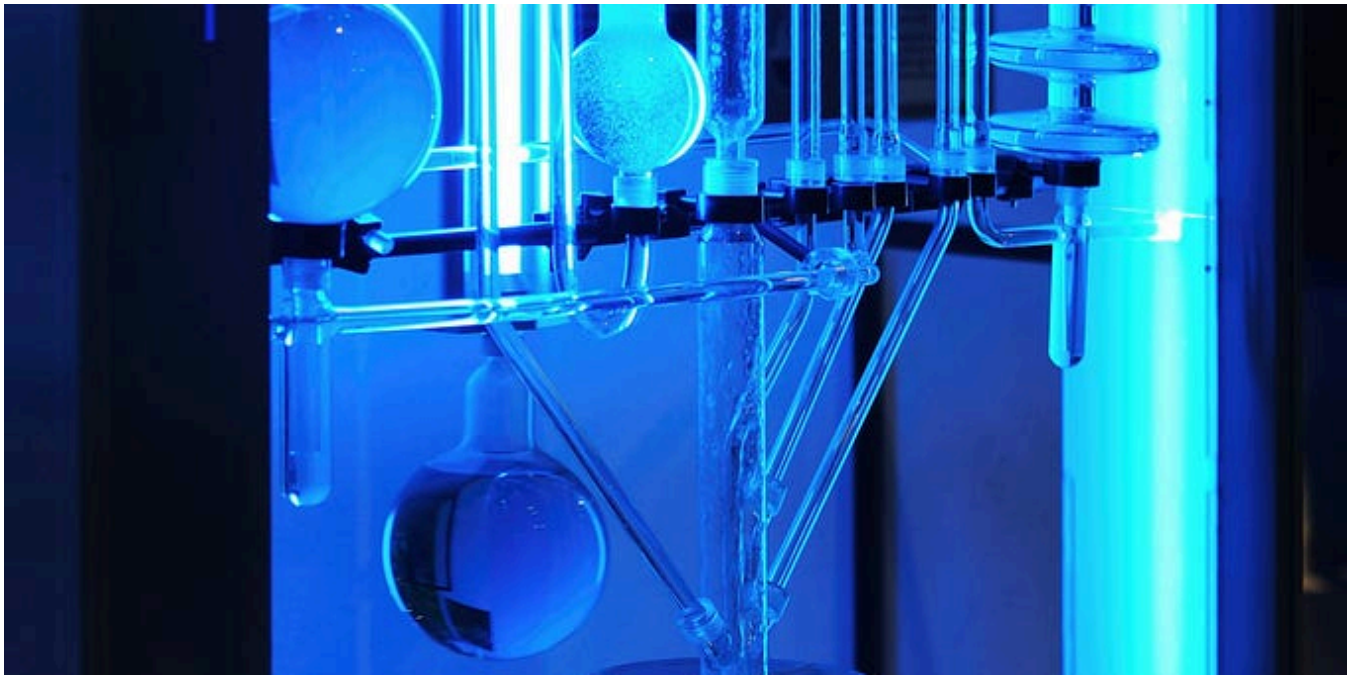


Rjnclarke

Build a Graph Neural Network Classifier in PyTorch

A Brief Diversion

Dec 19, 2024 1

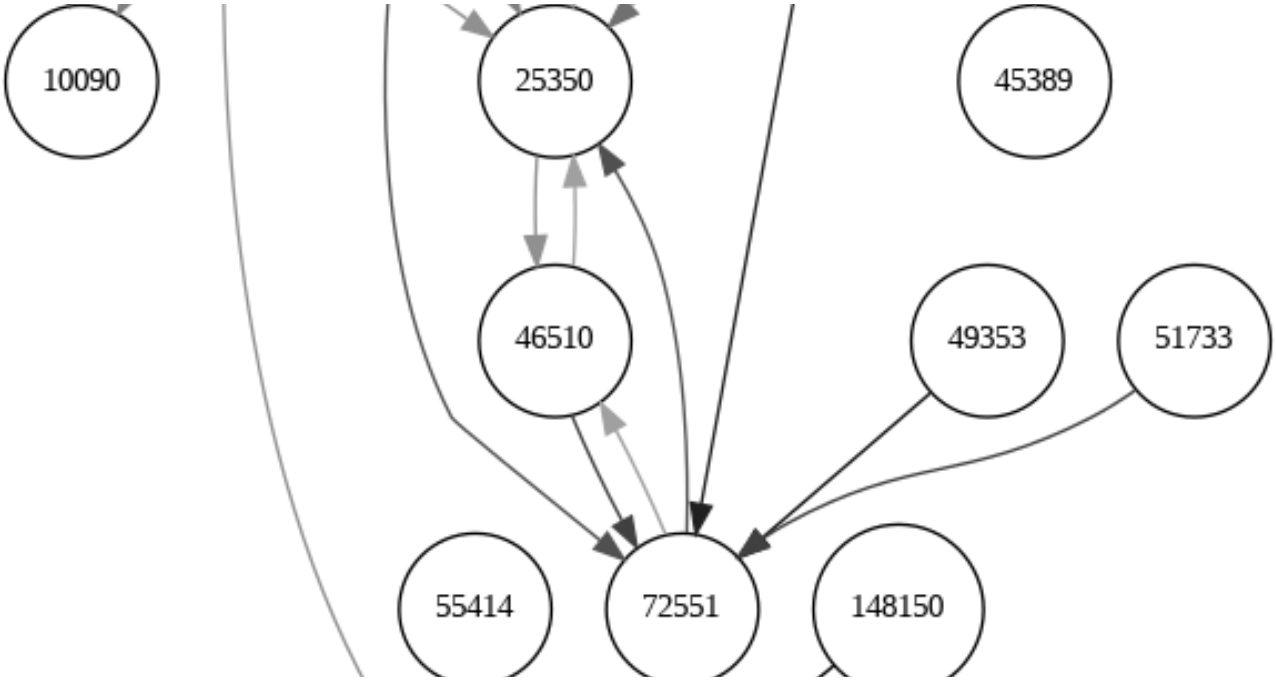


Anna Alexandra Grigoryan

Advancing Physics-Informed Neural Networks (PINNs): Their Role, Extensions, and Challenges—Part 3

Continuing my exploration of Physics-Informed Neural Networks (PINNs), I'm excited to share the third post in this series.

Dec 27, 2024 4

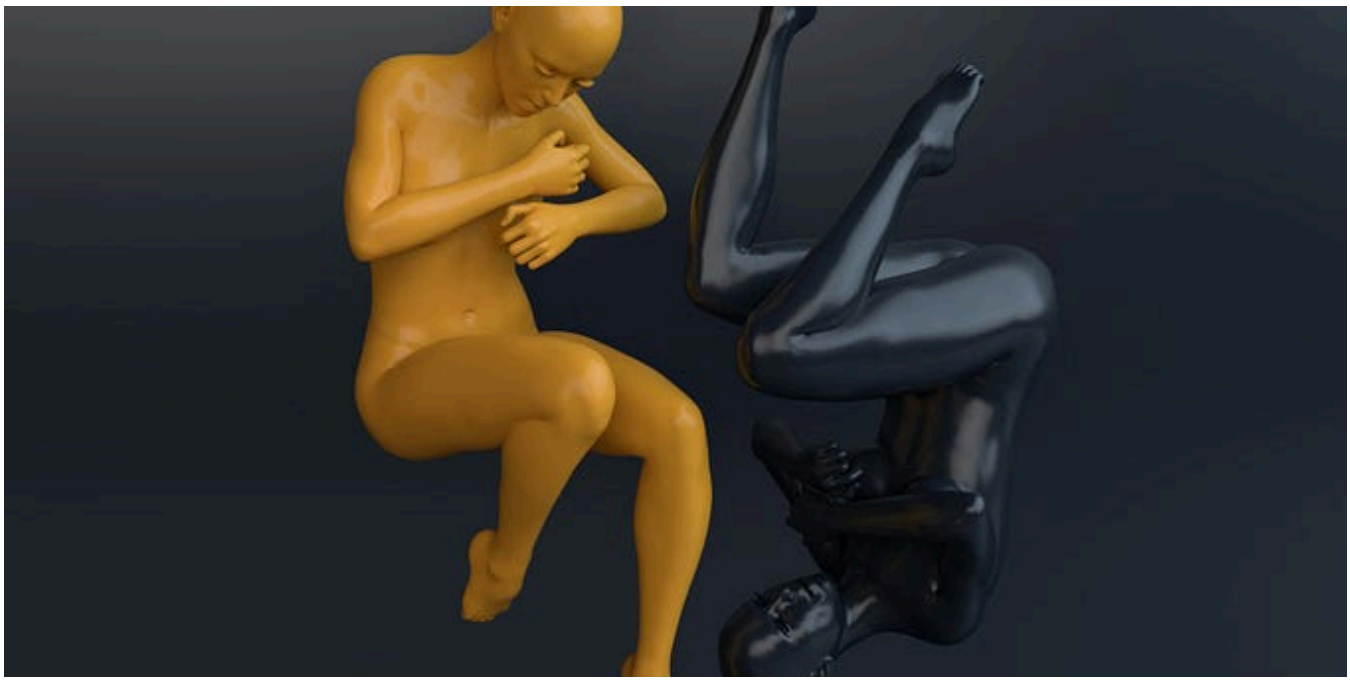


In Stanford CS224W: Machine Learning with Graphs by Ethan Bogle

Neural Networks and Model Explainability for Graphs

Authors: Ethan Bogle and Hollie Zheng

Jan 4 2

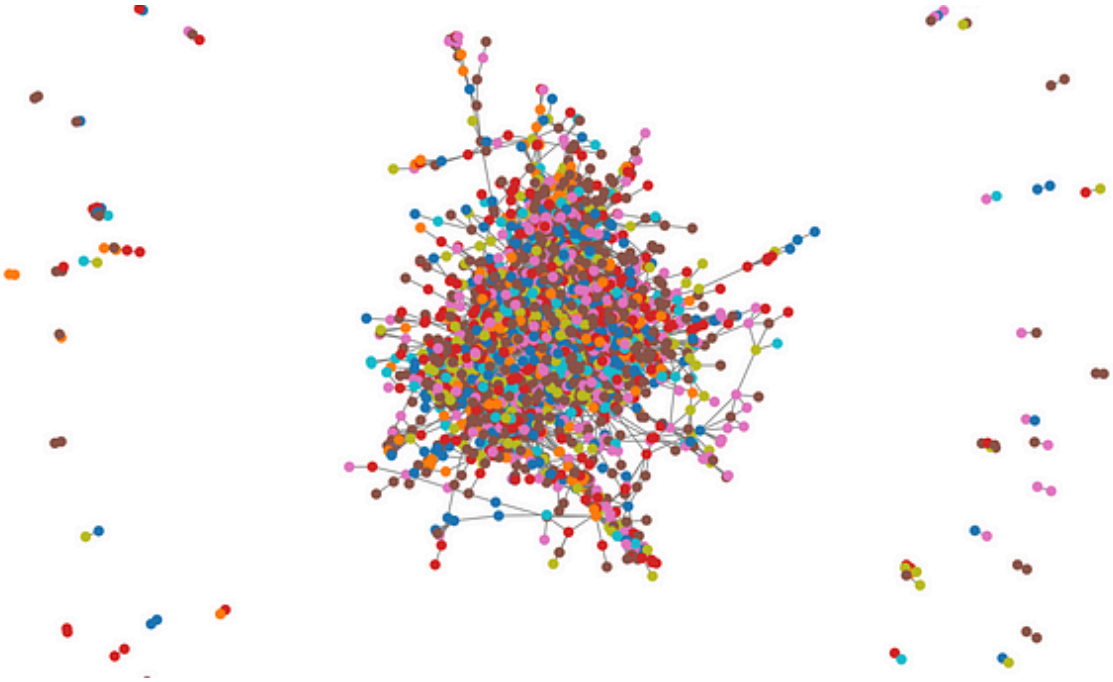


Deeraj Manjaray

Generating 3D Images from 2D Using Open3D Python

A Beginner's Guide to Getting Started with an Open-source Library for Processing 3D Data

Jan 1 76

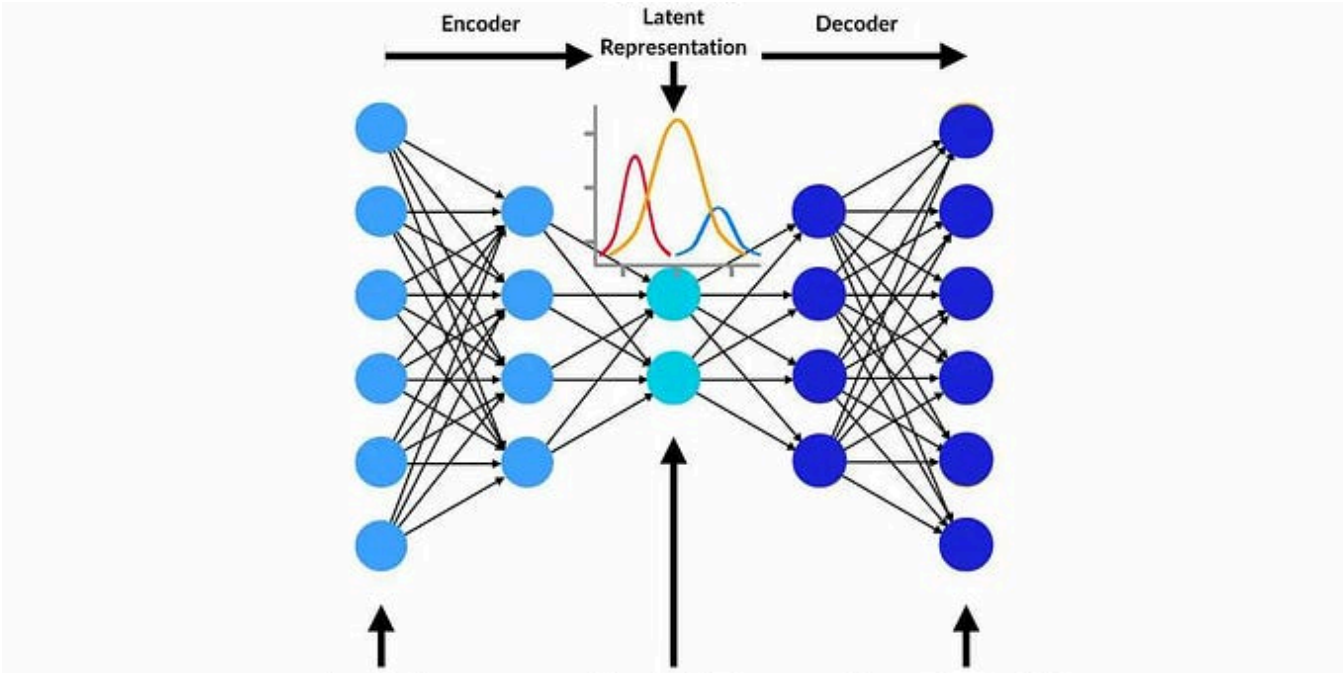


Machine Learning With K

Graph Convolutional Networks (GCN)

Convolutional Neural Networks (CNN) is one of the state-of-the-art approaches in modern machine vision. It was introduced in 1998 by Yann...

Feb 13 1 1



baris_kaplan

An Overview of Variational Autoencoder (VAE), KL Divergence, Evidence Lower Bound (ELBO)...

Introduction:

Jan 31 101

See more recommendations