Title: Graph neural network

URL: https://en.wikipedia.org/wiki/Graph_neural_network

PageID: 68162942

Categories: Category:2009 in artificial intelligence, Category:Artificial neural networks, Category:Graph algorithms, Category:Neural network architectures, Category:Semisupervised learning, Category:Supervised learning, Category:Unsupervised learning

-----

Supervised learning

Unsupervised learning

Semi-supervised learning

Self-supervised learning

Reinforcement learning

Meta-learning

Online learning

Batch learning

Curriculum learning

Rule-based learning

Neuro-symbolic AI

Neuromorphic engineering

Quantum machine learning

Classification

Generative modeling

Regression

Clustering

Dimensionality reduction

Density estimation

Anomaly detection

Data cleaning

AutoML

Association rules

Semantic analysis

Structured prediction

Feature engineering

Feature learning

Learning to rank

Grammar induction

Ontology learning

Multimodal learning

Apprenticeship learning

Decision trees

Ensembles Bagging Boosting Random forest

Bagging

Boosting

Random forest

k -NN

Linear regression

Naive Bayes

Artificial neural networks

Logistic regression

Perceptron

Relevance vector machine (RVM)

Support vector machine (SVM)

BIRCH

CURE

Hierarchical

k -means

Fuzzy

Expectation–maximization (EM)

DBSCAN

OPTICS

Mean shift

Factor analysis

CCA

ICA

LDA

NMF

PCA

PGD

t-SNE

SDL

Graphical models Bayes net Conditional random field Hidden Markov

Bayes net

Conditional random field

Hidden Markov

RANSAC

k -NN

Local outlier factor

Isolation forest

Autoencoder

Deep learning

Feedforward neural network

Recurrent neural network LSTM GRU ESN reservoir computing

LSTM

GRU

ESN

reservoir computing

Boltzmann machine Restricted

Restricted

GAN

Diffusion model

SOM

Convolutional neural network U-Net LeNet AlexNet DeepDream

U-Net

LeNet

AlexNet

DeepDream

Neural field Neural radiance field Physics-informed neural networks

Neural radiance field

Physics-informed neural networks

Transformer Vision

Vision

Mamba

Spiking neural network

Memtransistor

Electrochemical RAM (ECRAM)

Q-learning

Policy gradient

SARSA

Temporal difference (TD)

Multi-agent Self-play

Self-play

Active learning

Crowdsourcing

Human-in-the-loop

Mechanistic interpretability

RLHF

Coefficient of determination

Confusion matrix

Learning curve

ROC curve

Kernel machines

Bias–variance tradeoff

Computational learning theory

Empirical risk minimization

Occam learning

PAC learning

Statistical learning

VC theory

Topological deep learning

AAAI

ECML PKDD

NeurIPS

ICML

ICLR

IJCAI

ML

JMLR

Glossary of artificial intelligence

List of datasets for machine-learning research List of datasets in computer vision and image processing

List of datasets in computer vision and image processing

Outline of machine learning

v

t

e

Graph neural networks (GNN) are specialized artificial neural networks that are designed for tasks whose inputs are graphs . [1] [2] [3] [4] [5]

One prominent example is molecular drug design. [6] [7] [8] Each input sample is a graph representation of a molecule, where atoms form the nodes and chemical bonds between atoms form the edges. In addition to the graph representation, the input also includes known chemical properties for each of the atoms. Dataset samples may thus differ in length, reflecting the varying numbers of atoms in molecules, and the varying number of bonds between them. The task is to predict the efficacy of a given molecule for a specific medical application, like eliminating E. coli bacteria.

The key design element of GNNs is the use of pairwise message passing , such that graph nodes iteratively update their representations by exchanging information with their neighbors. Several GNN architectures have been proposed, [2] [3] [9] [10] [11] which implement different flavors of message passing, [12] [13] started by recursive [2] or convolutional constructive [3] approaches. As of 2022 [update] , it is an open question whether it is possible to define GNN architectures "going beyond" message passing, or instead every GNN can be built on message passing over suitably defined graphs. [14]

In the more general subject of "geometric deep learning ", certain existing neural network architectures can be interpreted as GNNs operating on suitably defined graphs. [12] A convolutional neural network layer, in the context of computer vision , can be considered a GNN applied to graphs whose nodes are pixels and only adjacent pixels are connected by edges in the graph. A transformer layer, in natural language processing , can be considered a GNN applied to complete graphs whose nodes are words or tokens in a passage of natural language text.

Relevant application domains for GNNs include natural language processing , [15] social networks , [16] citation networks , [17] molecular biology , [18] chemistry, [19] [20] physics [21] and NP-hard combinatorial optimization problems. [22]

Open source libraries implementing GNNs include PyTorch Geometric [23] (PyTorch), TensorFlow GNN [24] (TensorFlow), Deep Graph Library [25] (framework agnostic), jraph [26] (Google JAX), and GraphNeuralNetworks.jl [27] /GeometricFlux.jl [28] (Julia , Flux).

Architecture

The architecture of a generic GNN implements the following fundamental layers : [12]

Permutation equivariant : a permutation equivariant layer maps a representation of a graph into an updated representation of the same graph. In the literature, permutation equivariant layers are implemented via pairwise message passing between graph nodes. [12] [14] Intuitively, in a message passing layer, nodes update their representations by aggregating the messages received from their immediate neighbours. As such, each message passing layer increases the receptive field of the GNN by one hop.

Local pooling : a local pooling layer coarsens the graph via downsampling . Local pooling is used to increase the receptive field of a GNN, in a similar fashion to pooling layers in convolutional neural networks . Examples include k-nearest neighbours pooling , top-k pooling, [29] and self-attention pooling. [30]

Global pooling : a global pooling layer, also known as readout layer, provides fixed-size representation of the whole graph. The global pooling layer must be permutation invariant, such that permutations in the ordering of graph nodes and edges do not alter the final output. [31] Examples include element-wise sum, mean or maximum.

It has been demonstrated that GNNs cannot be more expressive than the Weisfeiler–Leman Graph Isomorphism Test . [32] [33] In practice, this means that there exist different graph structures (e.g., molecules with the same atoms but different bonds) that cannot be distinguished by GNNs. More powerful GNNs operating on higher-dimension geometries such as simplicial complexes can be designed. [34] [35] [13] As of 2022 [update] , whether or not future architectures will overcome the message passing primitive is an open research question. [14]

Message passing layers

Message passing layers are permutation-equivariant layers mapping a graph into an updated representation of the same graph. Formally, they can be expressed as message passing neural networks (MPNNs). [12]

Let G = (V , E) {\displaystyle G=(V,E)} be a graph , where V {\displaystyle V} is the node set and E {\displaystyle E} is the edge set. Let N u {\displaystyle N_{u}} be the neighbourhood of some node u ∊ V {\displaystyle u\in V} . Additionally, let x u {\displaystyle \mathbf {x} _{u}} be the features of node u ∊ V {\displaystyle u\in V} , and e u v {\displaystyle \mathbf {e} _{uv}} be the features of edge (u , v) ∊ E {\displaystyle (u,v)\in E} . An MPNN layer can be expressed as follows: [12]

where φ {\displaystyle \phi } and ψ {\displaystyle \psi } are differentiable functions (e.g., artificial neural networks), and ■ {\displaystyle \bigoplus } is a permutation invariant aggregation operator that can accept an arbitrary number of inputs (e.g., element-wise sum, mean, or max). In particular, φ {\displaystyle \phi } and ψ {\displaystyle \psi } are referred to as update and message functions, respectively. Intuitively, in an MPNN computational block, graph nodes update their representations by aggregating the messages received from their neighbours.

The outputs of one or more MPNN layers are node representations h u {\displaystyle \mathbf {h} _{u}} for each node u ∈ V {\displaystyle u\in V} in the graph. Node representations can be employed for any downstream task, such as node/graph classification or edge prediction.

Graph nodes in an MPNN update their representation aggregating information from their immediate neighbours. As such, stacking n {\displaystyle n} MPNN layers means that one node will be able to communicate with nodes that are at most n {\displaystyle n} "hops" away. In principle, to ensure that every node receives information from every other node, one would need to stack a number of MPNN layers equal to the graph diameter . However, stacking many MPNN layers may cause issues such as oversmoothing [36] and oversquashing. [37] Oversmoothing refers to the issue of node representations becoming indistinguishable. Oversquashing refers to the bottleneck that is created by squeezing long-range dependencies into fixed-size representations. Countermeasures such as skip connections [10] [38] (as in residual neural networks), gated update rules [39] and jumping knowledge [40] can mitigate oversmoothing. Modifying the final layer to be a fully-adjacent layer, i.e., by considering the graph as a complete graph , can mitigate oversquashing in problems where long-range dependencies are required. [37]

Other "flavours" of MPNN have been developed in the literature, [12] such as graph convolutional networks [9] and graph attention networks, [11] whose definitions can be expressed in terms of the MPNN formalism.

Graph convolutional network

The graph convolutional network (GCN) was first introduced by Thomas Kipf and Max Welling in 2017. [9]

A GCN layer defines a first-order approximation of a localized spectral filter on graphs. GCNs can be understood as a generalization of convolutional neural networks to graph-structured data.

The formal expression of a GCN layer reads as follows:

where H {\displaystyle \mathbf {H} } is the matrix of node representations h u {\displaystyle \mathbf {h} _{u}} , X {\displaystyle \mathbf {X} } is the matrix of node features x u {\displaystyle \mathbf {x} _{u}} , σ (·) {\displaystyle \sigma (\cdot)} is an activation function (e.g., ReLU), A ~ {\displaystyle {\tilde {\mathbf {A} }}} is the graph adjacency matrix with the addition of self-loops, D ~ {\displaystyle {\tilde {\mathbf {D} }}} is the graph degree matrix with the addition of self-loops, and Θ {\displaystyle \mathbf {\Theta } } is a matrix of trainable parameters.

In particular, let A {\displaystyle \mathbf {A} } be the graph adjacency matrix: then, one can define A ~ = A + I {\displaystyle {\tilde {\mathbf {A} }}=\mathbf {A} +\mathbf {I} } and D ~ i i = ∑ j ∈ V A ~ i j {\displaystyle {\tilde {\mathbf {D} }}_{ii}=\sum _{j\in V}{\tilde {A}}_{ij}} , where I {\displaystyle \mathbf {I} } denotes the identity matrix . This normalization ensures that the eigenvalues of D ~ – 1 2 A ~ D ~ – 1 2 {\displaystyle {\tilde {\mathbf {D} }}^{-{\frac {1}{2}}}{\tilde {\mathbf {A} }}{\tilde {\mathbf {D} }}^{-{\frac {1}{2}}}} are bounded in the range [0 , 1] {\displaystyle [0,1]} , avoiding numerical instabilities and exploding/vanishing gradients .

A limitation of GCNs is that they do not allow multidimensional edge features e u v {\displaystyle \mathbf {e} _{uv}} . [9] It is however possible to associate scalar weights w u v {\displaystyle w_{uv}} to each edge by imposing A u v = w u v {\displaystyle A_{uv}=w_{uv}} , i.e., by setting each nonzero entry in the adjacency matrix equal to the weight of the corresponding edge.

Graph attention network

The graph attention network (GAT) was introduced by Petar Veli■kovi■ et al. in 2018. [11]

Graph attention network is a combination of a GNN and an attention layer.

The implementation of attention layer in graphical neural networks helps provide attention or focus to the important information from the data instead of focusing on the whole data.

A multi-head GAT layer can be expressed as follows:

where K {\displaystyle K} is the number of attention heads, ■ {\displaystyle {\Big \Vert }} denotes vector concatenation , σ (·) {\displaystyle \sigma (\cdot)} is an activation function (e.g., ReLU), α i j {\displaystyle \alpha _{ij}} are attention coefficients, and W k {\displaystyle W^{k}} is a matrix of trainable parameters for the k {\displaystyle k} -th attention head.

For the final GAT layer, the outputs from each attention head are averaged before the application of the activation function. Formally, the final GAT layer can be written as:

Attention in Machine Learning is a technique that mimics cognitive attention . In the context of learning on graphs, the attention coefficient α u v {\displaystyle \alpha _{uv}} measures how important is node u ∈ V {\displaystyle u\in V} to node v ∈ V {\displaystyle v\in V} .

Normalized attention coefficients are computed as follows:

where a {\displaystyle \mathbf {a} } is a vector of learnable weights, · T {\displaystyle \cdot ^{T}} indicates transposition , e u v {\displaystyle \mathbf {e} _{uv}} are the edge features (if present), and LeakyReLU {\displaystyle {\text{LeakyReLU}}} is a modified ReLU activation function. Attention coefficients are normalized to make them easily comparable across different nodes. [11]

A GCN can be seen as a special case of a GAT where attention coefficients are not learnable, but fixed and equal to the edge weights w u v {\displaystyle w_{uv}} .

Gated graph sequence neural network

The gated graph sequence neural network (GGS-NN) was introduced by Yujia Li et al. in 2015. [39] The GGS-NN extends the GNN formulation by Scarselli et al. [2] to output sequences. The message passing framework is implemented as an update rule to a gated recurrent unit (GRU) cell.

A GGS-NN can be expressed as follows:

where ■ {\displaystyle \Vert } denotes vector concatenation , 0 {\displaystyle \mathbf {0} } is a vector of zeros, Θ {\displaystyle \mathbf {\Theta } } is a matrix of learnable parameters, GRU {\displaystyle {\text{GRU}}} is a GRU cell, and l {\displaystyle l} denotes the sequence index. In a GGS-NN, the node representations are regarded as the hidden states of a GRU cell. The initial node features x u (0) {\displaystyle \mathbf {x} _{u}^{(0)}} are zero-padded up to the hidden state dimension of the GRU cell. The same GRU cell is used for updating representations for each node.

Local pooling layers

Local pooling layers coarsen the graph via downsampling. Subsequently, several learnable local pooling strategies that have been proposed are presented. [31] For each case, the input is the initial graph represented by a matrix X {\displaystyle \mathbf {X} } of node features, and the graph adjacency matrix A {\displaystyle \mathbf {A} } . The output is the new matrix X ′ {\displaystyle \mathbf {X} '} of node features, and the new graph adjacency matrix A ′ {\displaystyle \mathbf {A} '} .

Top-k pooling

We first set

y = X p ■ p ■ {\displaystyle \mathbf {y} ={\frac {\mathbf {X} \mathbf {p} }{\Vert \mathbf {p} \Vert }}}

where p {\displaystyle \mathbf {p} } is a learnable projection vector. The projection vector p {\displaystyle \mathbf {p} } computes a scalar projection value for each graph node.

The top-k pooling layer [29] can then be formalised as follows:

where i = top k (y) {\displaystyle \mathbf {i} ={\text{top}}_{k}(\mathbf {y})} is the subset of nodes with the top-k highest projection scores, ■ {\displaystyle \odot } denotes element-wise matrix multiplication , and sigmoid (·) {\displaystyle {\text{sigmoid}}(\cdot)} is the sigmoid function . In other words, the nodes with the top-k highest projection scores are retained in the new adjacency matrix A ′ {\displaystyle \mathbf {A} '} . The sigmoid (·) {\displaystyle {\text{sigmoid}}(\cdot)} operation makes the projection vector p {\displaystyle \mathbf {p} } trainable by backpropagation , which otherwise would produce discrete outputs. [29]

Self-attention pooling

We first set

where GNN {\displaystyle {\text{GNN}}} is a generic permutation equivariant GNN layer (e.g., GCN, GAT, MPNN).

The Self-attention pooling layer [30] can then be formalised as follows:

where i = top k (y) {\displaystyle \mathbf {i} ={\text{top}}_{k}(\mathbf {y})} is the subset of nodes with the top-k highest projection scores, ■ {\displaystyle \odot } denotes element-wise matrix multiplication .

The self-attention pooling layer can be seen as an extension of the top-k pooling layer. Differently from top-k pooling, the self-attention scores computed in self-attention pooling account both for the graph features and the graph topology.

Heterophilic Graph Learning

Homophily principle, i.e., nodes with the same labels or similar attributes are more likely to be connected, has been commonly believed to be the main reason for the superiority of Graph Neural Networks (GNNs) over traditional Neural Networks (NNs) on graph-structured data, especially on node-level tasks. [41] However, recent work has identified a non-trivial set of datasets where GNN’s performance compared to the NN’s is not satisfactory. [42] Heterophily , i.e., low homophily, has been considered the main cause of this empirical observation. [43] People have begun to revisit and re-evaluate most existing graph models in the heterophily scenario across various kinds of graphs, e.g., heterogeneous graphs , temporal graphs and hypergraphs . Moreover, numerous graph-related applications are found to be closely related to the heterophily problem, e.g. graph fraud/anomaly detection , graph adversarial attacks and robustness , privacy, federated learning and point cloud segmentation , graph clustering , recommender systems , generative models , link prediction , graph classification and coloring , etc. In the past few years, considerable effort has been devoted to studying and addressing the heterophily issue in graph learning. [41] [43] [44]

Applications

Protein folding

Graph neural networks are one of the main building blocks of AlphaFold , an artificial intelligence program developed by Google 's DeepMind for solving the protein folding problem in biology . AlphaFold achieved first place in several CASP competitions. [45] [46] [40]

Social networks

Social networks are a major application domain for GNNs due to their natural representation as social graphs . GNNs are used to develop recommender systems based on both social relations and item relations. [47] [16]

Combinatorial optimization

GNNs are used as fundamental building blocks for several combinatorial optimization algorithms. [48] Examples include computing shortest paths or Eulerian circuits for a given graph, [39] deriving chip placements superior or competitive to handcrafted human solutions, [49] and improving expert-designed branching rules in branch and bound . [50]

Cyber security

When viewed as a graph, a network of computers can be analyzed with GNNs for anomaly detection. Anomalies within provenance graphs often correlate to malicious activity within the network. GNNs have been used to identify these anomalies on individual nodes [51] and within paths [52] to detect malicious processes, or on the edge level [53] to detect lateral movement .

Water distribution networks

Water distribution systems can be modelled as graphs, being then a straightforward application of GNN. This kind of algorithm has been applied to water demand forecasting, [54] interconnecting District Measuring Areas to improve the forecasting capacity. Other application of this algorithm on water distribution modelling is the development of metamodels. [55]

Computer Vision

To represent an image as a graph structure, the image is first divided into multiple patches, each of which is treated as a node in the graph. Edges are then formed by connecting each node to its nearest neighbors based on spatial or feature similarity. This graph-based representation enables the application of graph learning models to visual tasks. The relational structure helps to enhance feature extraction and improve performance on image understanding. [56]

Text and NLP

Graph-based representation of text helps to capture deeper semantic relationships between words. Many studies have used graph networks to enhance performance in various text processing tasks such as text classification, question answering, Neural Machine Translation (NMT), event extraction, fact verification, etc. [57]

References

External links

A Gentle Introduction to Graph Neural Networks

v

t

e

History timeline

timeline

Companies

Projects

Parameter Hyperparameter

Hyperparameter

Loss functions

Regression Bias–variance tradeoff Double descent Overfitting

Bias–variance tradeoff

Double descent

Overfitting

Clustering

Gradient descent SGD Quasi-Newton method Conjugate gradient method

SGD

Quasi-Newton method

Conjugate gradient method

Backpropagation

Attention

Convolution

Normalization Batchnorm

Batchnorm

Activation Softmax Sigmoid Rectifier

Softmax

Sigmoid

Rectifier

Gating

Weight initialization

Regularization

Datasets Augmentation

Augmentation

Prompt engineering

Reinforcement learning Q-learning SARSA Imitation Policy gradient

Q-learning

SARSA

Imitation

Policy gradient

Diffusion

Latent diffusion model

Autoregression

Adversary

RAG

Uncanny valley

RLHF

Self-supervised learning

Reflection

Recursive self-improvement

Hallucination

Word embedding

Vibe coding

Machine learning In-context learning

In-context learning

Artificial neural network Deep learning

Deep learning

Language model Large language model NMT

Large language model

NMT

Reasoning language model

Model Context Protocol

Intelligent agent

Artificial human companion

Humanity's Last Exam

Artificial general intelligence (AGI)

AlexNet

WaveNet

Human image synthesis

HWR

OCR

Computer vision

Speech synthesis 15.ai ElevenLabs

15.ai

ElevenLabs

Speech recognition Whisper

Whisper

Facial recognition

AlphaFold

Text-to-image models Aurora DALL-E Firefly Flux Ideogram Imagen Midjourney Recraft Stable Diffusion

Aurora

DALL-E

Firefly

Flux

Ideogram

Imagen

Midjourney

Recraft

Stable Diffusion

Text-to-video models Dream Machine Runway Gen Hailuo AI Kling Sora Veo

Dream Machine

Runway Gen

Hailuo AI

Kling

Sora

Veo

Music generation Riffusion Suno AI Udio

Riffusion

Suno AI

Udio

Word2vec

Seq2seq

GloVe

BERT

T5

Llama

Chinchilla AI

PaLM

GPT 1 2 3 J ChatGPT 4 4o o1 o3 4.5 4.1 o4-mini 5

1

2

3

J

ChatGPT

4

4o

o1

o3

4.5

4.1

o4-mini

5

Claude

Gemini Gemini (language model) Gemma

Gemini (language model)

Gemma

Grok

LaMDA

BLOOM

DBRX

Project Debater

IBM Watson

IBM Watsonx

Granite

PanGu-Σ

DeepSeek

Qwen

AlphaGo

AlphaZero

OpenAI Five

Self-driving car

MuZero

Action selection AutoGPT

AutoGPT

Robot control

Alan Turing

Warren Sturgis McCulloch

Walter Pitts

John von Neumann

Claude Shannon

Shun'ichi Amari

Kunihiko Fukushima

Takeo Kanade

Marvin Minsky

John McCarthy

Nathaniel Rochester

Allen Newell

Cliff Shaw

Herbert A. Simon

Oliver Selfridge

Frank Rosenblatt

Bernard Widrow

Joseph Weizenbaum

Seymour Papert

Seppo Linnainmaa

Paul Werbos

Geoffrey Hinton

John Hopfield

Jürgen Schmidhuber

Yann LeCun

Yoshua Bengio

Lotfi A. Zadeh

Stephen Grossberg

Alex Graves

James Goodnight

Andrew Ng

Fei-Fei Li

Alex Krizhevsky

Ilya Sutskever

Oriol Vinyals

Quoc V. Le

Ian Goodfellow

Demis Hassabis

David Silver

Andrej Karpathy

Ashish Vaswani

Noam Shazeer

Aidan Gomez

John Schulman

Mustafa Suleyman

Jan Leike

Daniel Kokotajlo

François Chollet

Neural Turing machine

Differentiable neural computer

Transformer Vision transformer (ViT)

Vision transformer (ViT)

Recurrent neural network (RNN)

Long short-term memory (LSTM)

Gated recurrent unit (GRU)

Echo state network

Multilayer perceptron (MLP)

Convolutional neural network (CNN)

Residual neural network (RNN)

Highway network

Mamba

Autoencoder

Variational autoencoder (VAE)

Generative adversarial network (GAN)

Graph neural network (GNN)

Category