Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Philip, S. Y. (2020). A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems.

<https://jonathan-hui.medium.com/graph-convolutional-networks-gcn-pooling-839184205692>

Graph Convolutional Networks(GCN) & Pooling

What is the challenge?

Even though it is overkill to represent an image this way, a large percentage of machine learning(ML) problems will be much natural and effective to be modelled by a graph. In particular, when the relationships between neighbouring nodes are irregular and high dimensional, we need to define them explicitly in order to solve them effectively. In CNN, we work in a Euclidean space. How weights are associated with the input features(pixels) is well defined.

But this is not the case for a graph. For example, the graphs below are the same even though it looks different spatially.

In general, neural networks(NNs) takes an input x to predict z.

Diagram, schematic

Description automatically generated This leads us to challenge of how a NN can process a graph directly.

A picture containing clock, watch

Description automatically generatedIn GCN(graph convolutional network), the input to the NN will be a graph. Also, instead of interring a single z, it infers the value zi for each node in the graph. And to make prediction for Z, GCN utilizes both X and its neighbouring nodes in the calculation.

Graph Convolutional Networks(GCN)

Instead of having a 2-D array as input, GCN takes a graph as an input. The first diagram( the first row) below is the NN we know and the second diagram is the GCN with a graph containing four nods as the input.

Bubble chart

Description automatically generated

In the first NN, it contains multiple dense layers (fully connected layers). x is the input for the first layer and z\_i is the output of layer i. For each layer, we multiple z (or x for the first layer) with the weight matrix W and the pass the output to activation function ReLU. GCN is very similar, but the input to ReLU is AHW instead of Wz. Z\_i , H\_i are the output vectors from the last hidden layer for NN and GCN respectively. But please note that Wi and Wi are different and have different dimensions. And for the first layer in GCN, X contains an array of nodes instead of a single node x. X will be encoded as matrix with each row contains the features of a node. GCN introduces an adjacency matrix A + I.(diagonal matrix)

That comes to the output of hidden layer to be σ(ÂHⁱWⁱ). AH sums up features on each node with its neighbours. However, we may face the diminishing or exploding problem in a NN if we don’t have certain control over the range of hidden layer output. In specific, GCN wants A to be normalized to maintain the scale of the output feature vectors. One possibility is to multiple A with D where D is the diagonal node degree matrix of A in measuring the degree of each node. At a high level, instead of summing up itself with its neighbour, multiplying the sum with the inverse D sort of averages them. Specifically D is diagonal matrix with the each diagonal element D counts the number of edges for the corresponding node . i. and the output for each hidden layer becomes σ(D̂⁻¹ÂHⁱWⁱ). The equation used to compute a hidden layer output from the last layer output is called upon the propagation rule. 

Besides using , there are other choices. Indeed the propagation rule can be generalized as:

Text, letter

Description automatically generated

FOR CLASSIFICATION OR CLUSTERING

Zachary’s karate club

To find the club membership, we apply two hidden layers below with Hⁱ⁺¹ = σ(D̂⁻¹ÂHⁱWⁱ). the input will be the social graph above (without the club labels) and the las hidden layer will be the output.

The last hidden layer outputs 2 scalar latent features per node and we use these values to represent a node. (Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907)

All propagation rules discussed so far are differentiable. For semi-supervised or unsupervised problems, we can use backpropagation to train the weights using the labelled data if needed.

Chart, bubble chart

Description automatically generated

In short, we just apply clustering on the input social graph. Intuitively, our example produces latent features similar to their neighbours as the hidden layer is averaging the latent features with its neighbours. The Algorithm manages to cluster neighbours together by generating latent factors related to their neighbours.

Lets consider a semi-supervised problem for classification or clustering in which only one data point is labelled for each class or cluster. Once the latent features for each node are calculated, we can compute the distance between non-labelled data and the labelled data. Then we find the nearest neighbours to a known labelled data to classify or cluster the unlabelled data.

<https://www.youtube.com/watch?v=Iiv9R6BjxHM> Graph Convolutional Network (GCNs)

Part 1. In this section, we discuss the architecture and convolution of traditional convolutional neural networks. Then we extend to the graph domain. We understand the characteristics of graph and define the graph convolution. Finally we introduce spectrum graph convolutional neural network.

Architecture of Traditional ConvNets

**Convnet – computer vision, speech, NLP**

**Powerful architecture to solve high dimensional learning problems**

**Assumption – data(images, videos, speech) is compositional , it is formed of patterns – local, stationary, hierarchical**

**Convnets leverage the compositionality structure l they extract compositional features and feed them to classifier, recommender, etc (end-to-end system)**

**(data domain)**

**Images, volumes, video s lie on 2D, 3D, 2D+1, Euclidean domains(grids)**

**Sentences, words, speech lie on 1D Euclidean domain(sequence)**

**These domain have strong regular spatial structures.**

**All convnet operations are mathematically well defined and fast (convolution, pooling)**

(graph domain) social network

**Drug, transportation, knowledge graph, gene, recommendation(Netflix, amazon), energy**

**G graph V vertices E edge A adjacent matrix**

**Graph features**

(vector) Node features Hi, Hj

Edge Eji

g

Diagram

Description automatically generated

Convolution of ConvNets

How do we extend convolution to graph

Convolutional layer (for grids)

Graphical user interface, Word

Description automatically generated

N1\*n2\*d = 3 \*3\*d ~ n1\*n2\*d

Diagram

Description automatically generated with medium confidence

Kernel -> graph convolution . template matching for graph?

No node ordering,

Node has no specific position. It is jus an index.

No position

Heterogeneous neighbourhood

How to define convolution

Fourier transform of the convolution of two functions is the pointwise product of their Fourier transforms



Fast spectral convolution for compact kernels?



Spectral Convolution

1 graph Laplacian

2 Fourier function

3 Fourier transform

4. convolution theorem

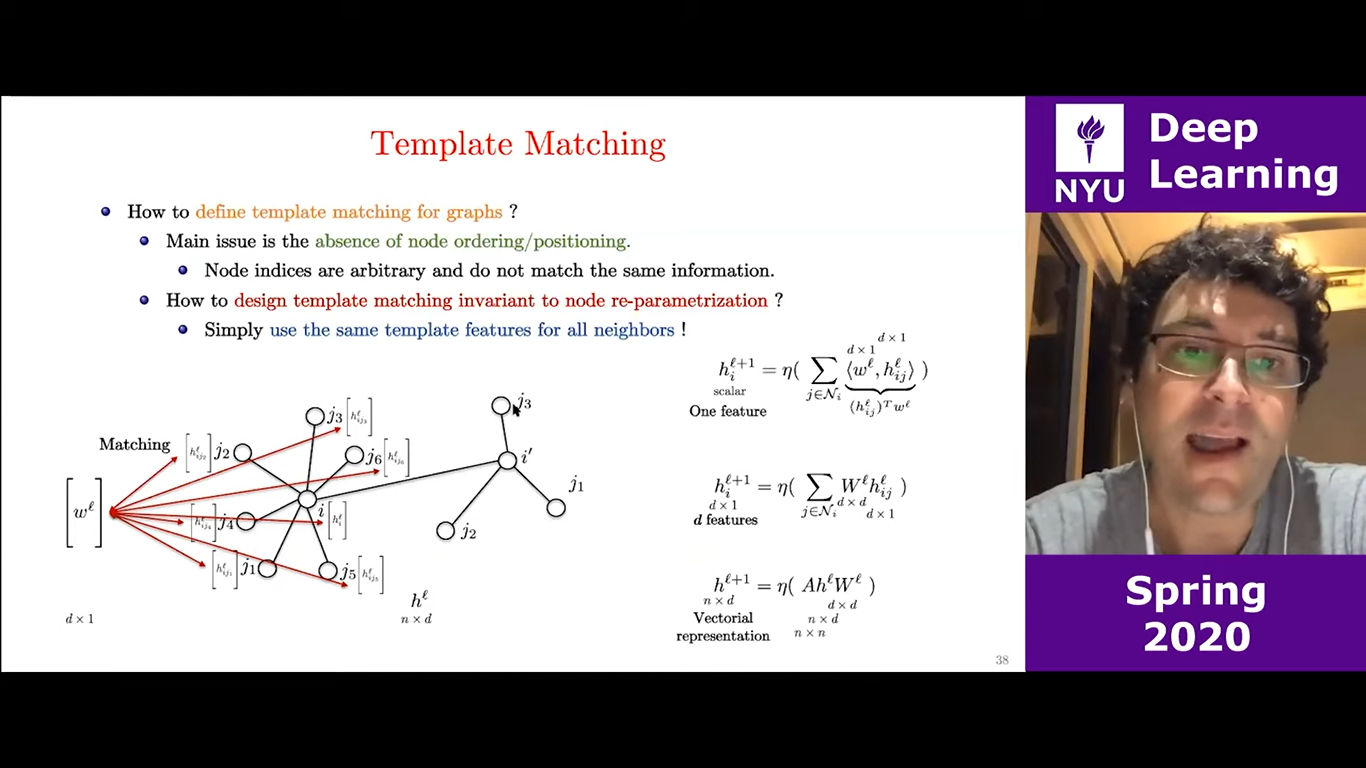
Graph Laplacian

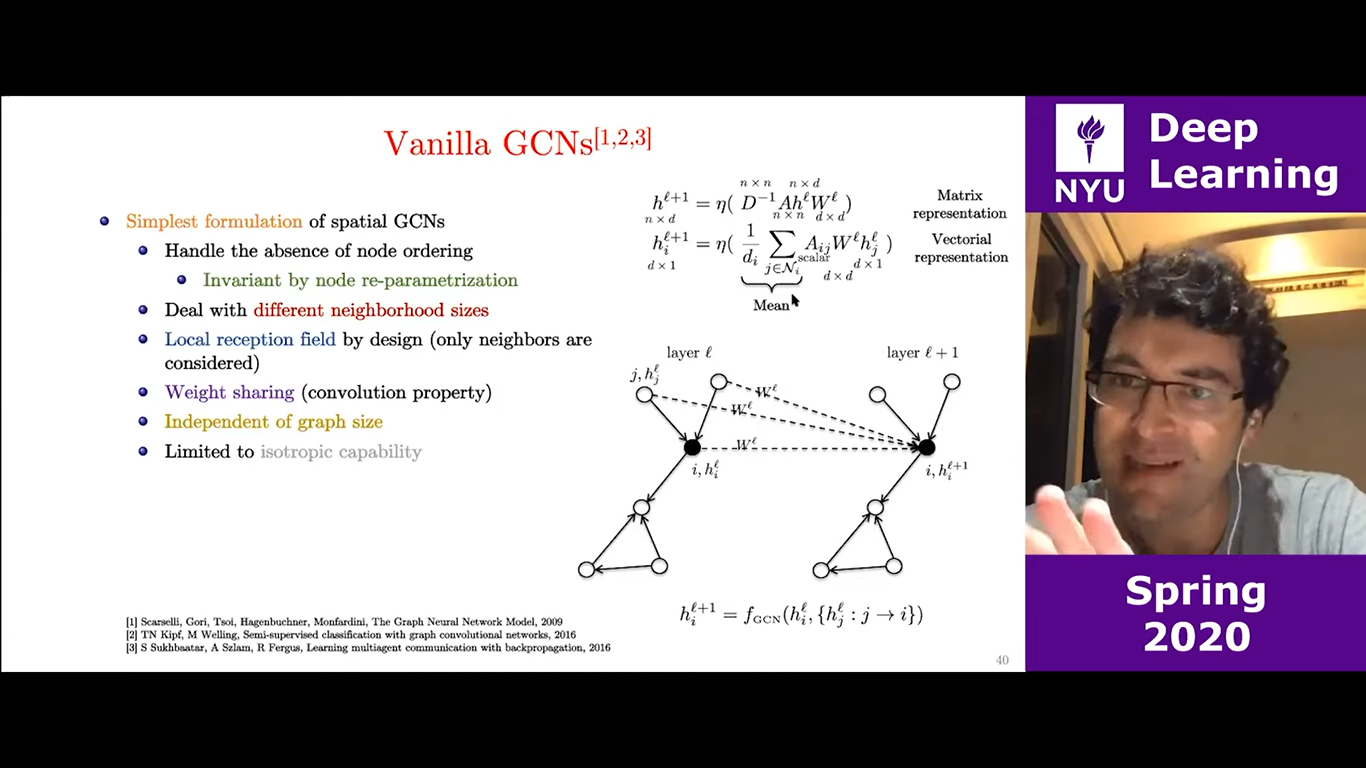
Chart, scatter chart

Description automatically generated

Part2. This section covers the complete spectrum of Graph convolutional networks, starting with the implementation of Spectral Convolution through Spectral Networks. It then provides insights on applicability of the other convolutional definition of Template Matching to graphs, leading to Spatial networks. Various architecture employing the two approaches are detailed out with their corresponding pros & cons, experiments, benchmarks and application.

Spatial GCN





Template Matching, Isotropic GCNs and Benchmarking GNNs

Anisotropic GCN ggcn, gan

Standard ConvNets produce anisotropic filters because Euclidean grids have directional structures( up, down, left, right)

GCN such as ChebNets, Cayleynets, Vanilla GCn, GraphSage, GIN compute isotropic filters as there is no notion of directions on arbitrary graphs.

How to get anisotropy back in GNN?

Natural edge features

Independent of the node parametrization

Edge degrees, edge gates, attention mechanism - gat, gated gcn

Github/graphdeeplearning/benchmarking-gnns

Same templates

Monets – Bayesian Gaussian Mixture model (gmm)

Diagram

Description automatically generated

Anisotropic GCNs

aggregation 에 따라Gated GCN, Attention GCN 등

Diagram

Description automatically generated with medium confidence

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Relational Graph neural network

RGCN – modelling relational data with graph convolutional networks Knowledge graph

Knowledge graph (KG)

Entity 와 relation – node, edge

Head, relation, tail (triplet – 이공냉면, located\_in, 이공계캠퍼스) 이공냉면, is\_a, 음식점

그러나 not complete

강의수강 knowledge graph

Graph type heterogenous, undirected 강의 , 수강, 전공 edge

R1: 희망, 수강 학생 강의

R2: 같은 1 전공, 같은 2전공 학생 학생

R3: 같은 이수구분, 같은 개설학과 강의-강의

Relationship 이 여러개라면? This is more like real world, multi-relational graph convolutional network

Propagation – convolutional, spectral convolution은 높은 비용이 소모되므로, spatial convolution

Isotropic 한 모델들은 가중치를 다르게 두지 않기 때문에 Anisotropic 모델 선택 (monets, GGCN, GAT)

Graph type l rgcn, multi relational graph

Multi relational graph convolutional network

Sampling 등

Modelling Relational Data with Graph Convolution network

Abstract

Knowledge graph enable a wide variety of applications, including question answering and information retrieval. Despite the great effort invested in their creation and maintenance, even the largest remain incomplete. We introduce Relational Graph Convolutional Networks (r-gcn) and apply them to two standard knowledge base completion task, link prediction ( recovery of missing fact, - subject -predicate- object triples) and entity classification ( recovery of missing entity attributes). R-GCN are related to a recent class of neural networks operating on graphs, and are developed specifically to deal with the highly multi-relational data characteristics of realistic knowledge bases. We demonstrate the effective ness of R gcn as a stand alone model for entity classification. We further show that factorization models for link prediction such as DistMult can be significantly improved by enriching them with an encode model to accumulation evidence over multiple inference steps in the relational graph, demonstrating a large improvement of 29.8% over decode-only baseline.

Knowledge bases organize and store factual knowledge, enabling a multitude of applications including question answering and information retrieval.

We consider two fundamental SRL tasks ; link prediction and entity classification

In both cases, many missing pieces of information can be expected to reside within the graph encoded through the neighbourhood structure – knowing that Mikhail was educated at the Vaganova academy implies both that Mikhail should have the label person, and that the triple (Mikhail, lived\_in, Russia) must belong to the knowledge graph. Following this intuition, we develop an encoder model for entities in the relational graph and apply it to both tasks.

Out entity classification model, use softmax classifiers at each node in the graph. The classifiers take node representations supplied by a relational graph convolution network(rgcn) and predict the labels. The mode, including rgcn parameters, is learned by optimizing the cross-entropy loss.

Out link prediction model can be regarded as an autoencoder consisting of an encoder: RGCN producing latent feature representations of entities, and a decoder: a tensor factorization model exploiting theses representations to predict labelled edges. Though in principle the decoder can rely on any type of factorization ( or generally any scoring function ) we use one of the simplest and most effective factorization methods: mistmult

We observe that out method achieved competitive results on standard benchmarks outperforming mammon other baseline direct optimization of the factorization vanilla distmult.

Our main contributions are as follows. To the best of our knowledge, we are the first to show that the GCN frame work can be applied to modelling relational data, specifically to link prediction and entity classification tasks. Secondly. We introduce techniques for parameter sharing and to enforce sparsity constraints and use them to apply rgcns to multigraphs with large numbers of relations lastly, we show that the performance of factorization, models, (distmult) can be significantly improved by enriching them with an encoder model that performs multiple steps of information propagation in the relational graph.

2. neural relational modelling

We introduce the following notation: we denote directed and labelled multi graphs as G with nodes (entities) and labelled edges(relation) (v, r, v) r relation type

2.1 relational graph convolutional networks

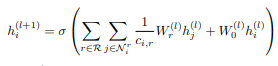
Our model is primarily motivated as an extension of GCNs that operate on local graph neighbourhoods to large-scale relational data. These and related methods such as graph neural networks can be understood as special cases of a simple differentiable message-passing framework, where

Message Passing framework ?

H is the hidden state of node vi in the lth layer of the neural network. With d is dimension

Messages of form are accumulated and passed through an element-wise activation Relu. (message-specific )neural network-like function or simply linear transformation



 this type of transformation has been shown to be very effective at accumulating and encoding features from local, structured neighbourhoods, and has led to significant improvements in areas such as graph classification and graph -based semi supervised learning.

Motivated by these architectures, we define the following simple propagation model for calculating the forward-pass update of an entity or node denoted by vi in a relational multi graph

Where N denotes the set of neighbour indices of node I under relation r . c is a problem specific normalization constant that can eighter be learned or chosed in advance ? (Attention? Gated?)

3.entity classification

Loss function – cross entropy loss

For (semi) supervised classification of nodes(entities) we simply stack rgcn layers of the form with softmax activation per node on the output of the lst layer. We minimize the following cross entropy loss on all labelled nodes(while ignoring unlabelled nodes)

Where y Is the set of node indices that have labels and h is the kth entry of the network output for ith labelled node. T denotes its respective group truth label. In practice we train the model using gradient descent techniques. A schematic depiction of our entity classification model is in figure 3a.

4. link prediction ; input rgcn – distmult(decode) edge loss

Link prediction deals with prediction of new facts(i.e triples(subject, relation , object)

Formally the knowledge vase is represented by a directed, labelled graph G (VER)

Rather than the full set of edge E , we are given only an incomplete set E. the task is to assign **scores f(s,r,o)** to possible dedge in order to determine how likely those edges are to belong to E.

In order to tackle this problem, we introduce a graph auto-encoder model, comprised of an entity encoder and a scoring function (decoder) the encoder maps each entity to real valued vector e . the decoder reconstructs edges of the graph relying on the vertex representation: in order words, it scores (subject, relation , object)-triples through a function

Tensor and nueral factorization methods can be interpreted under this framework. Distmult factorization as the scoring function, which is known to perform well on standard link prediction benchmarks when used on its own. In distmult, every relation r is associated with a diagonal matrix, and a triple ais scores.

Regularization