VISIUM

Cutting-edge AI for the world's best brands

Spinoff ETH zürich

Shedding light on graph neural networks

AMLD Workshop - 29.09.2021

Arnaud Dhaene & William Cappelletti

VISIUM

Your Visium team for today

William Cappelletti

Machine Learning Engineer

Arnaud Dhaene

AI Solution Specialist

Camilla Casamento

Machine Learning Engineer

Matteo Togninalli, PhD

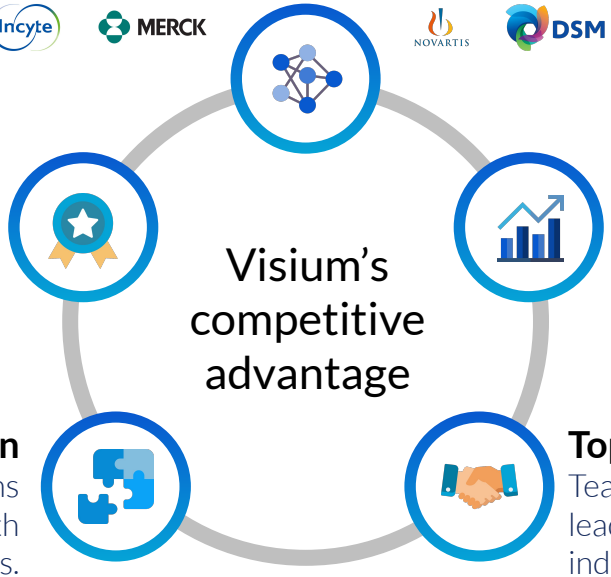
Chief Operating Officer

Visium | Leading AI-consultancy with close connection to academia

Broad experience in AI/ML projects

Successfully completed 90+ projects with 40+ happy clients of combined revenues over CHF 400 BN (Healthcare: 20+ projects).

Roche · Incyte · MERCK · NOVARTIS · DSM · Firmenich



One-stop-shop for AI

Expertise in providing E2E AI-service from use-case ideation, model development to scaling within organization.

Exponential growth

Advising leading companies on AI since 2018 and grown to 40 team members without external funding.

Academia collaboration

Visium is an ETH spin-off and retains close collaboration with leading AI-universities.

Top notch engineers

Team members hold MSc and PhDs from leading AI-universities and possess broad industry AI-project experience.

ETH zürich · EPFL · Stanford University · HARVARD UNIVERSITY · Imperial College London

VISIUM

Table of Contents

VISIUM

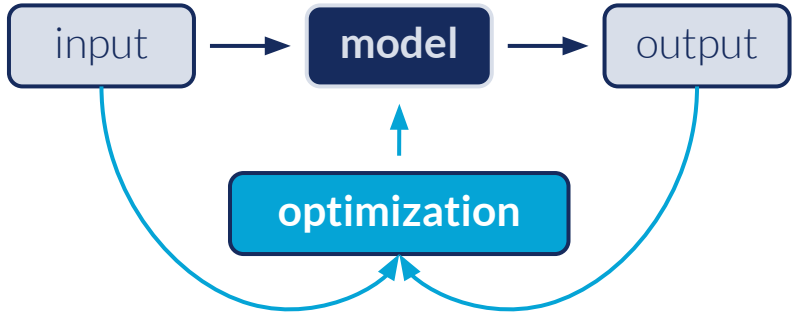
Introduction

Deep Learning on Graphs?

VISIUM

Introduction to Deep Learning

Machine Learning studies algorithms that improve by **observing data**



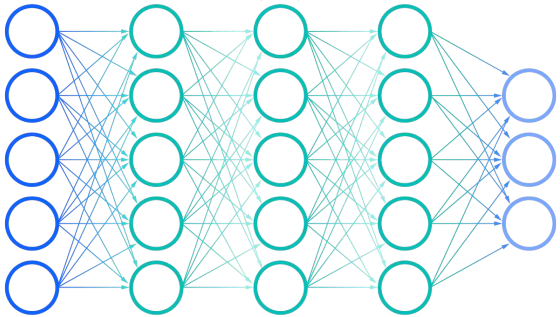


Fig. from ibm.com/cloud/learn/neural-networks

Deep Learning is a family of ML methods based on **Artificial Neural Networks.**

It has been very successful thanks to its **flexibility** to adapt **architectures** and various **data types** to the specific **task**.

Data contains structure that can be leveraged

Data have inherent **structure**, which should be taken into account by the models:

- **Images** have spatial relationships
  - ➜ **CNNs** (convolutions)

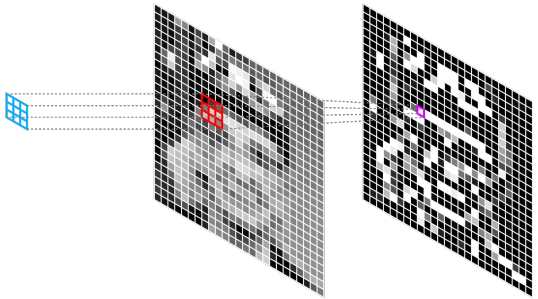


Fig. from Gregory Gundersen blog

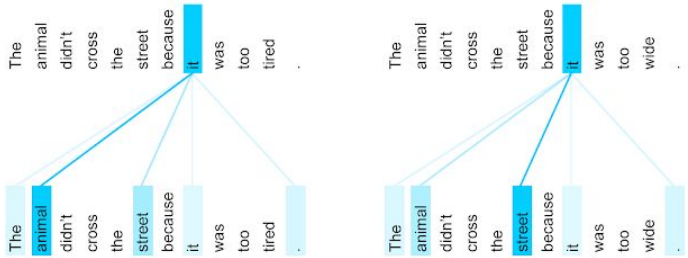


Fig. from Google AI blog

- **Language**: sequential and syntactic structures
  - ➜ **Transformers** (attention and recurrence)

What if our data has even more structure?

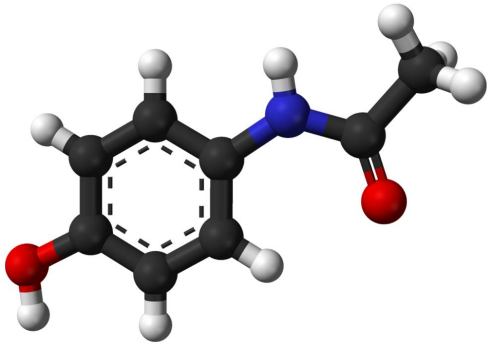
VISIUM

Network representations contain a wealth of information

Many real world data can be naturally modeled as **graphs:**

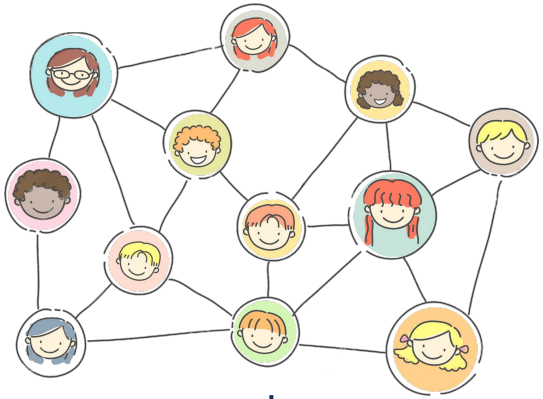
Sets of entities, or **nodes**, with pairwise relations, i.e. a set of **edges**

Molecules



Atoms sharing **bonds**

Social Networks



People with **friendship** relations

Transportation Maps

Intersections connected by **roads**

VISIUM

Fig.s from Petar Veličković CST Seminar

Network representations are used to solve problems

There are **many useful problems** that can be formulated over graphs. The structure's flexibility allows for a wide problem setting. The most common ones are:

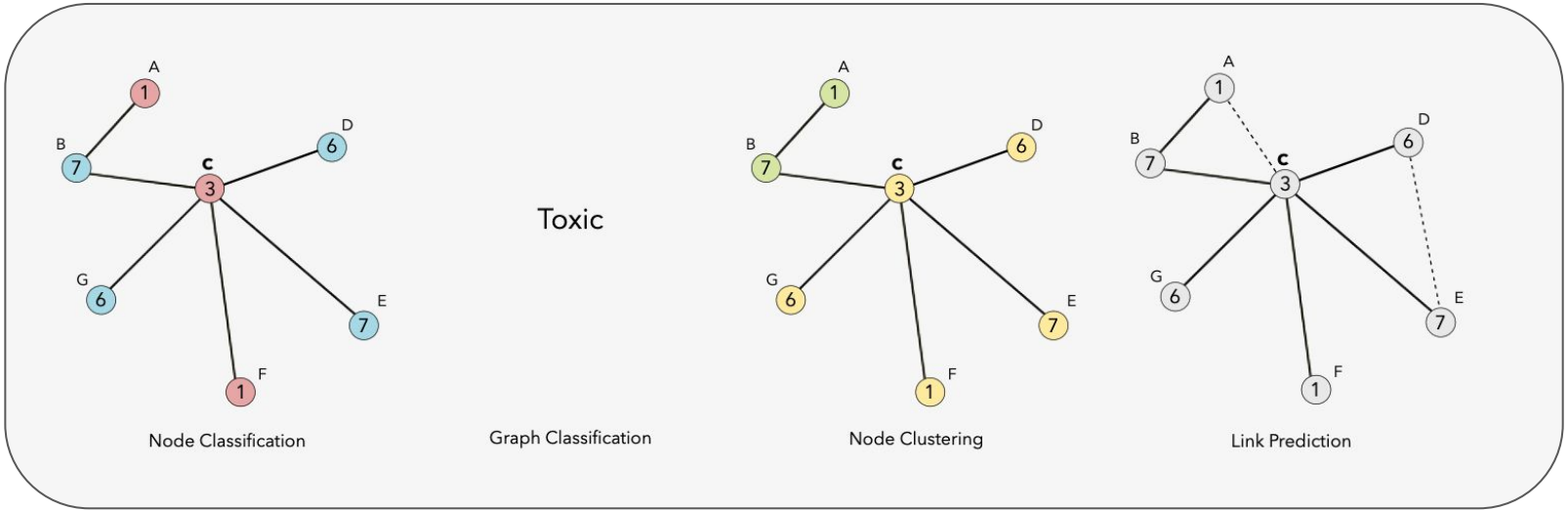


Figure from Understanding Convolutions on Graphs by Google Research

Graph Neural Networks are the method of choice for Graphs

Previous Deep Learning architectures (*CNNs* or *Transformers*) **DO NOT** work on graphs

➜ We have to introduce specific **Graph Neural Networks (GNN)**

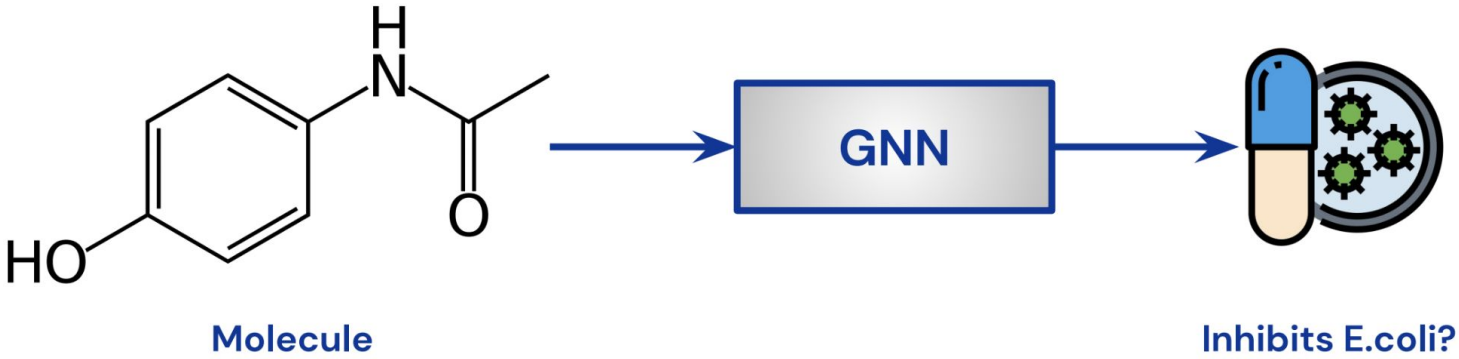
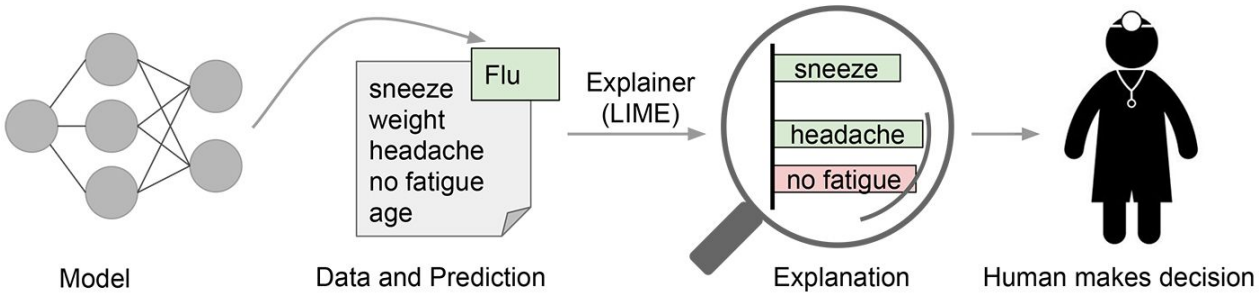


Fig. from Petar Veličković CST Seminar

GNNs, like other Neural Networks, are *Black Box Models*

More complex models make it harder to see what goes on under the hood.
This holds for GNNs as well.



In many applications, for instance *medical* or *decisional* problems, the **motivation** of an answer is as important as the answer itself, if not more.

We need **accountability** and **transparency**.

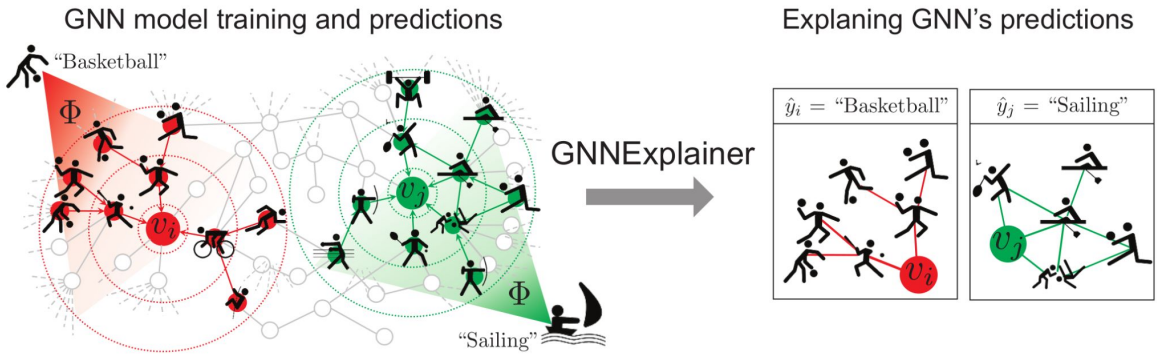
VISIUM

Fig. from *"Why should i trust you?" Explaining the predictions of any classifier.* Ribeiro et al. 2016, arxiv:1602.04938

Explaining the outcome of GNNs is essential to their use

We should try to **explain the predictions** made by the complex GNN models:

- **Why** does the GNN predict a certain output?
- **How** do *features* influence the prediction?
- **What** *structures* are semantically relevant for a certain *sample*, or a given *class*?



➜ Part of these questions are answered by **GNNExplainer**

VISIUM

Fig. from *GNNExplainer: Generating explanations for graph neural networks.* Ying, Rex, et al. 2019, arxiv:1903.03894

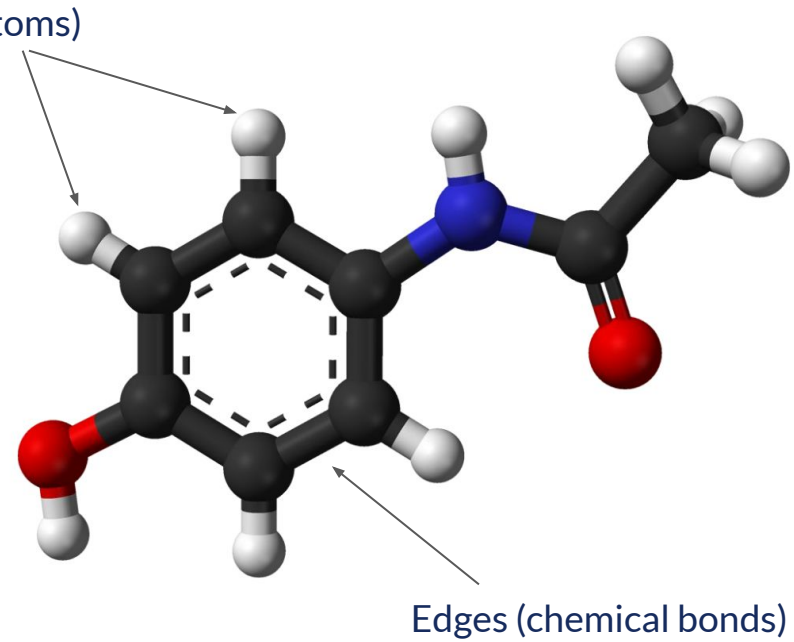
Graph data

Instilling natural structure in data

VISIUM

Network representations contain a wealth of information

A graph **G = (V, E)** can carry information both in its **nodes** and **edges**

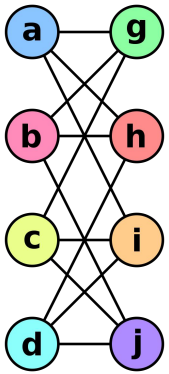


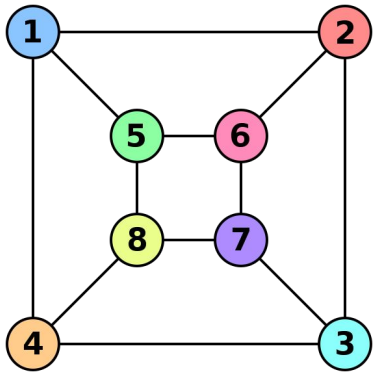
- **Node features**
  e.g. atomic number
- **Edge features**
  e.g. type of bond
  (simple, double)

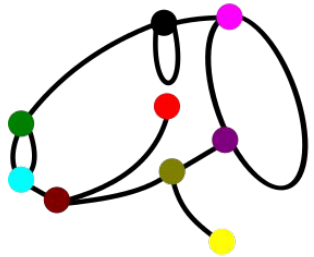
Graphs have very specific properties

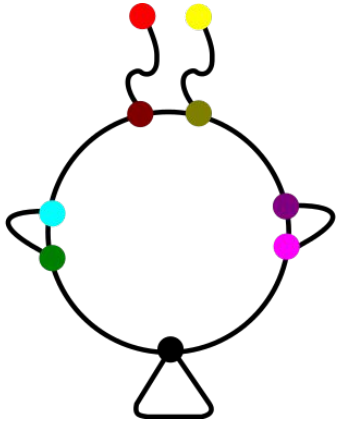
The main property of graphs is in **how the nodes are connected**

Graphs **structure** is **independent** from the labelling of the nodes,
or from how we choose to draw them



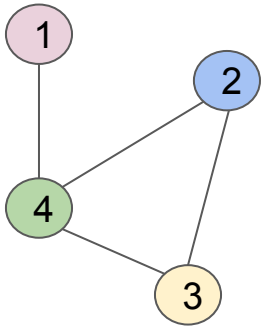




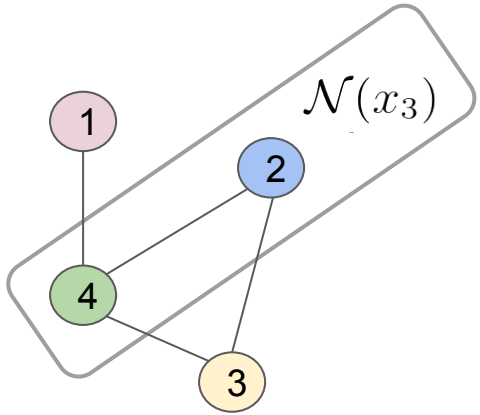


Graph representations

Adjacency matrix: $A_{i,j}$ = 1 iff there is an edge from node *i* to node *j*, otherwise $A_{i,j}$ = 0. Traditional network analysis techniques heavily rely on the use of the adjacency matrix.



$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$



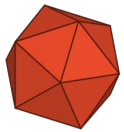
Neighbourhood: $\mathcal{N}(\mathbf{x}_v)$ represents the set of neighbors of node ***v***

Edge list: a list *E* of edges, each represented as a pair *(i, j)*, meaning that node *i* is connected to node *j*.

$$E = [[1,4],[4,1],[2,4],[4,2],[2,3],[3,2],[3,4],[4,3]]$$

There are sophisticated libraries to work with graph data

There are many libraries to integrate Graphs in ML frameworks



PyTorch geometric

We will use **PyTorch Geometric**

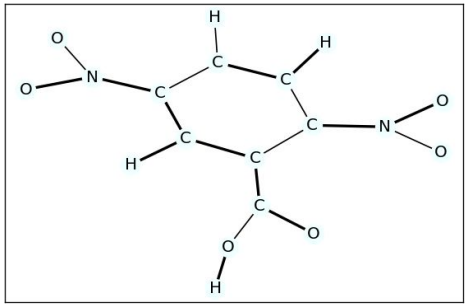
+ **Adjacency** and **features** matrices as (sparse) Tensors
+ Includes many **datasets**
+ Supports **batching** and **data loaders**

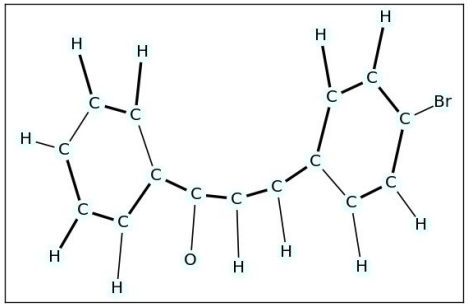
TUDatasets

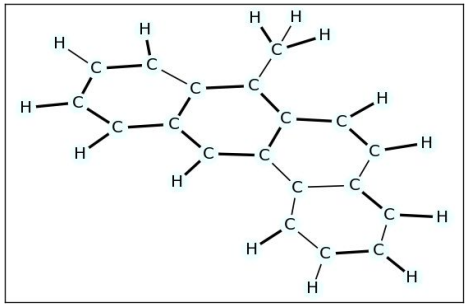
Let's get our hands dirty with some graphs!

Colab Notebook

1. Data loading and EDA
2. Graph visualization
3. Batching

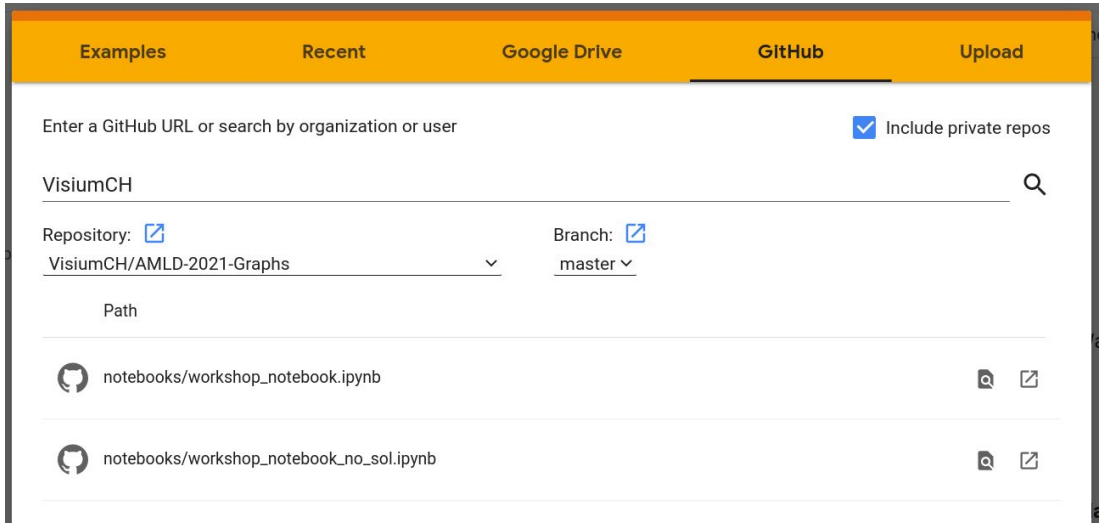






Open the notebook on Colab

1. Go to https://colab.research.google.com/
2. Under **GitHub** search for **VisiumCH**
3. Choose the **AMLD-2021-Graphs** repo
4. Open **notebooks/workshop_notebook.ipynb**



Direct link: https://colab.research.google.com/github/VisiumCH/AMLD-2021-Graphs/blob/master/notebooks/workshop_notebook.ipynb

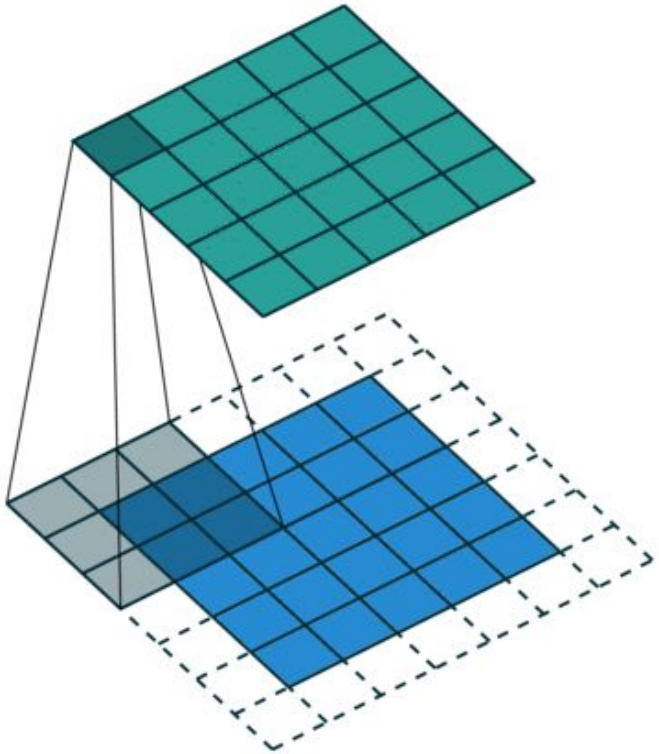
VISIUM

Deep Learning on Graphs

The implementation of Graph Neural Networks

VISIUM

Leveraging spatial information with convolutions



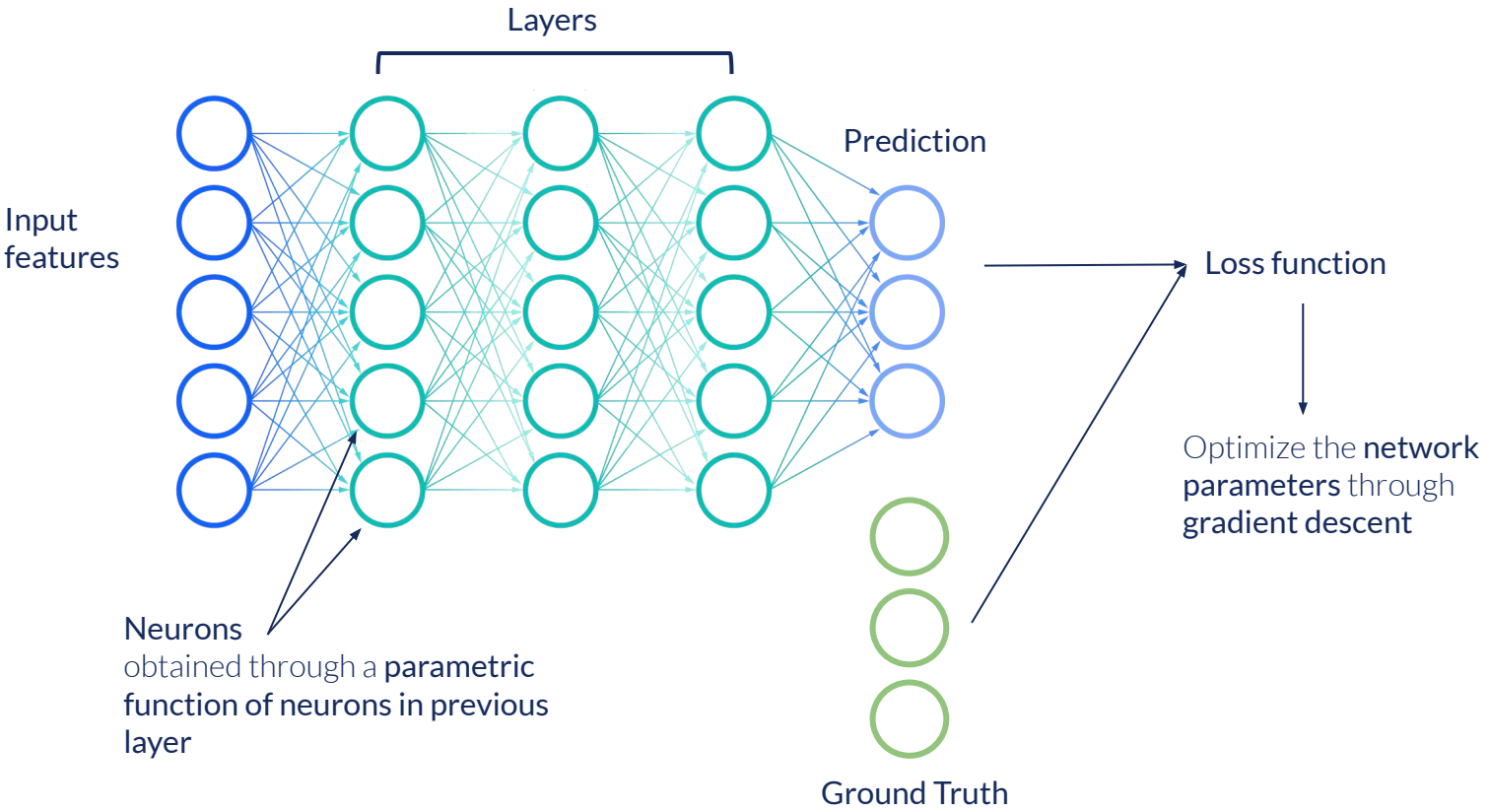
Properties of image convolution

- **Locality**
  Exploit position information
- **Translation equivariance**
  A translation of the input is reflected in the output (e.g. image segmentation)

Gif. from Theano Tutorial

VISIUM

The general structure of artificial neural networks

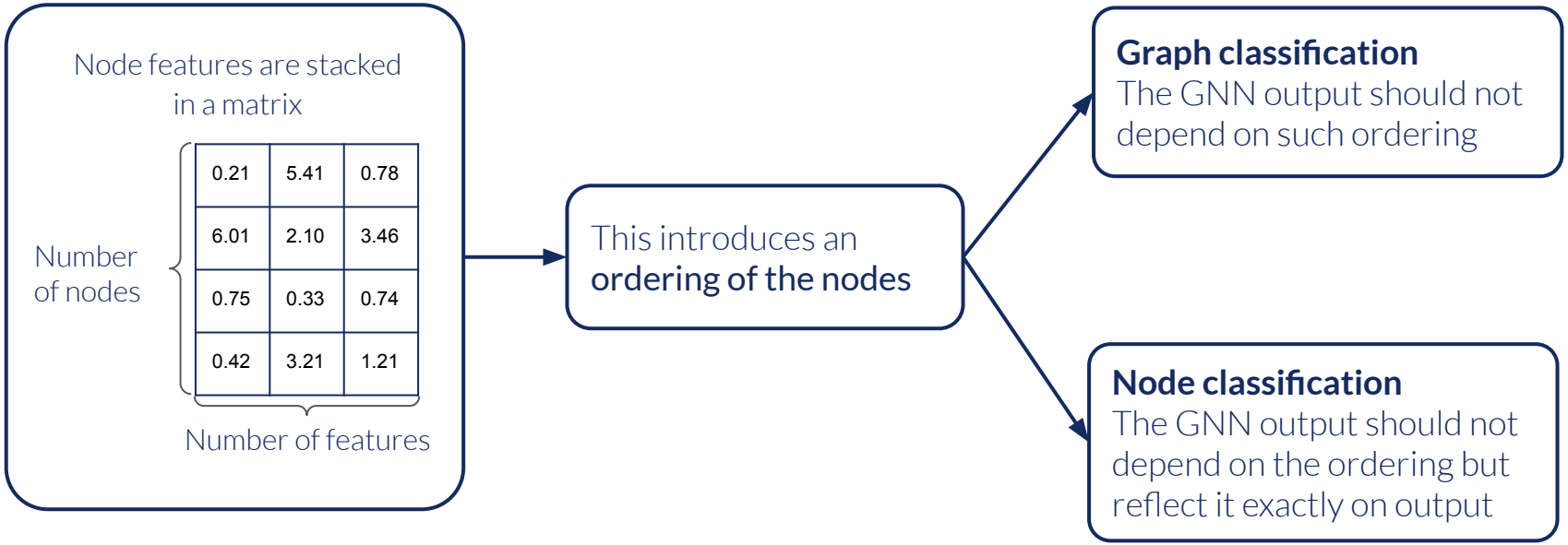


VISIUM

Fig. from ibm.com/cloud/learn/neural-networks

Towards Graph Neural Networks

What are the **desired properties** that **graph convolution** should satisfy?



Permutation invariance and equivariance

Let X be our feature matrix, P a permutation matrix and A the graph adjacency matrix.

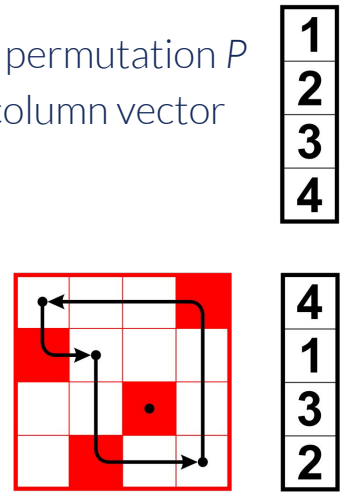
- A function f is said to be **permutation invariant** if

$$f(PX, PAP^T) = f(X, A)$$

- A function f is said to be **permutation equivariant** if

$$f(PX, PAP^T) = P(f(X, A))$$

Schema of a permutation P acting on a column vector



The desired properties for Graph Neural Networks

- **Locality**: the features of the neighbors of the considered node *i*, $N(x_i)$, should influence the computation of the latent representation of i

- **Permutation equivariance**: if we permute the order of the nodes in the input matrix, such permutation will also be applied to the nodes latent representations

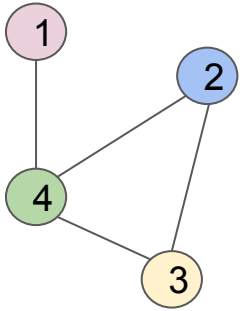
$$f(X, A) = \begin{pmatrix} g(x_1, \mathcal{N}(x_1)) \\ g(x_2, \mathcal{N}(x_2)) \\ \dots \\ g(x_N, \mathcal{N}(x_N)) \end{pmatrix}$$

graph convolution function *f*, permutation equivariant

g is permutation invariant over $N(x_i)$

the ordering of the neighbors of a node should not influence the node's latent representation

The inner workings of Graph Neural Networks | Example



Adjacency matrix

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

Feature matrix

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \in \mathbb{R}^{4 \times d}$$

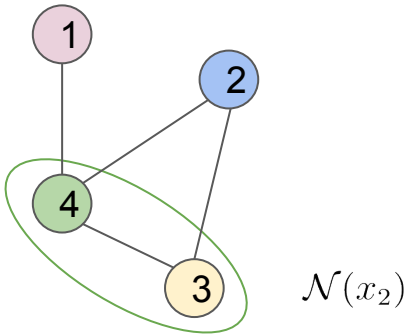
Graph convolution

$$f(X, A) = \begin{pmatrix} g(x_1, \mathcal{N}(x_1)) \\ g(x_2, \mathcal{N}(x_2)) \\ g(x_3, \mathcal{N}(x_3)) \\ g(x_4, \mathcal{N}(x_4)) \end{pmatrix} \rightarrow W_1 x_2 + W_2 (x_3 + x_4)$$

$\mathcal{N}(x_2)$

g permutation invariant over $N(x_i)$

Network parameters



$\mathcal{N}(x_2)$

VISIUM

The inner workings of Graph Neural Networks

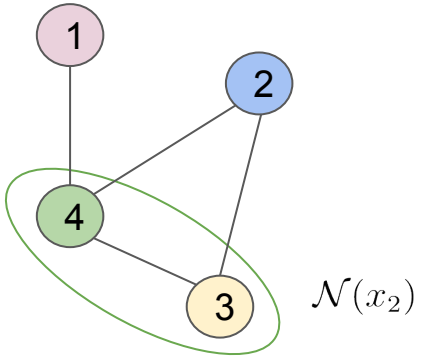
Graph convolution

$$f(X, A) = \begin{pmatrix} g(x_1, \mathcal{N}(x_1)) \\ g(x_2, \mathcal{N}(x_2)) \\ g(x_3, \mathcal{N}(x_3)) \\ g(x_4, \mathcal{N}(x_4)) \end{pmatrix}$$

g permutation invariant over $N(x_i)$

$$W_1 x_2 + W_2(x_3 + x_4)$$

$\mathcal{N}(x_2)$



An example
[Morris et al. (2018)]

$$\mathbf{x}_v^{(\ell+1)} = g(\mathbf{x}_v^{(\ell)}, \mathcal{N}(\mathbf{x}_v)) = \mathbf{W}_1^{(\ell+1)} \mathbf{x}_v^{(\ell)} + \mathbf{W}_2^{(\ell+1)} \sum_{w \in \mathcal{N}(v)} \mathbf{x}_w^{(\ell)}$$

Latent representation of node v in layer l+1

Shared weight matrices

The sum is permutation invariant

VISIUM

Graph Neural Networks solve three different tasks

Depending on the specific learning task, the latent nodes representations are aggregated differently

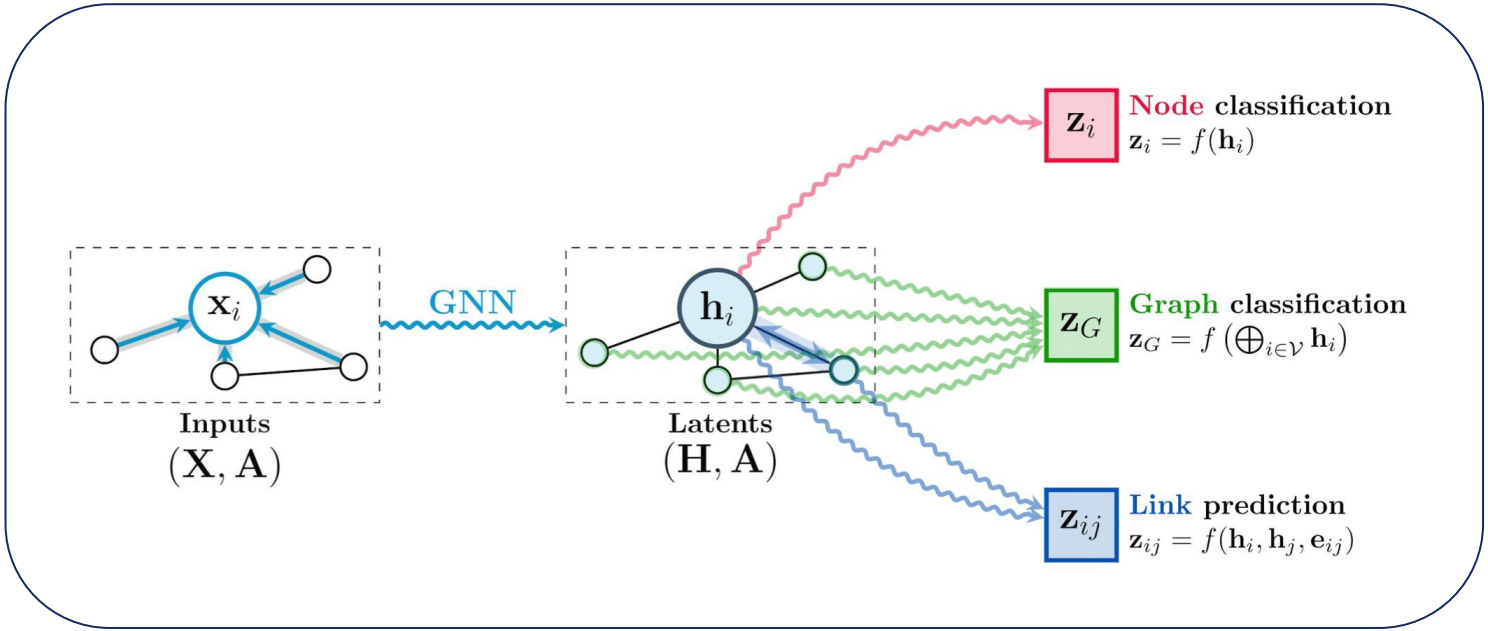


Fig. from Petar Veličković CST Seminar

VISIUM

Let's get our hands dirty with some GNNs!

Colab Notebook

1. Implement a **Graph-convolution** module
2. Build a deep network for **graph classification**
3. **Train** the model
4. **Evaluate** the model

Interpretability

How to evaluate the outcome of your model

VISIUM

The difference between inherent and post hoc interpretability

📘 **Interpretability** is the degree to which a human can **understand the cause of a decision**

- Some models are **inherently explainable**, which means that by definition we have a complete understanding of their mechanics
    - Linear and logistic regression
    - Decision trees
    - Nearest neighbors

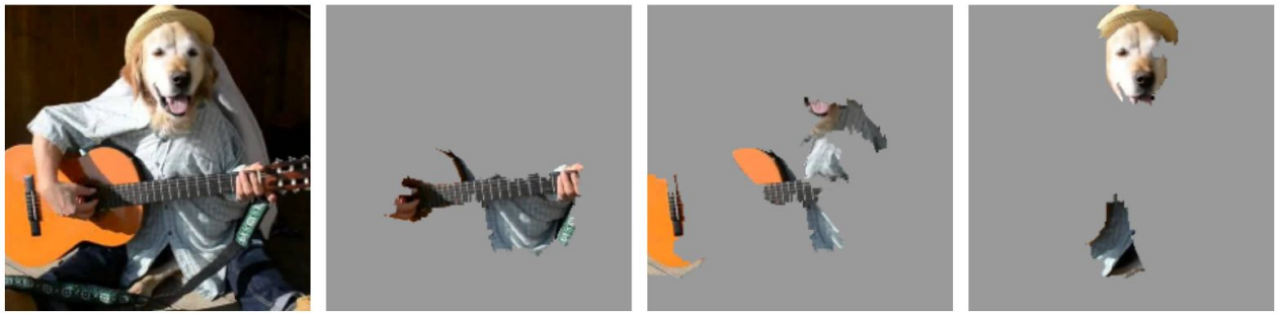
- Increasingly complex models, as *Deep Neural Networks*, are harder to grasp

    ➜ We have to use **post-hoc interpretation**
    ➜ Freedom for developers with **model-agnostic methods**

Current post-hoc Neural Network interpretability techniques

We can group interpretability methods for neural networks into **two main families**

- **Proxy models** formulate *locally faithful approximations* around the prediction, using simple models (linear) or sets of rules
  ➜ LIME (Ribeiro et al., 2016)



(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Fig. from *"Why should i trust you?" Explaining the predictions of any classifier.* Ribeiro et al. 2016, arxiv:1602.04938

What's going on under the hood of my ANN?

We can group interpretability methods for neural networks into **two main families**

- Identify important aspects of the **computation**
  - **Learned features** through *activation maximization*
  - Find **input-specific units** through *network dissection*

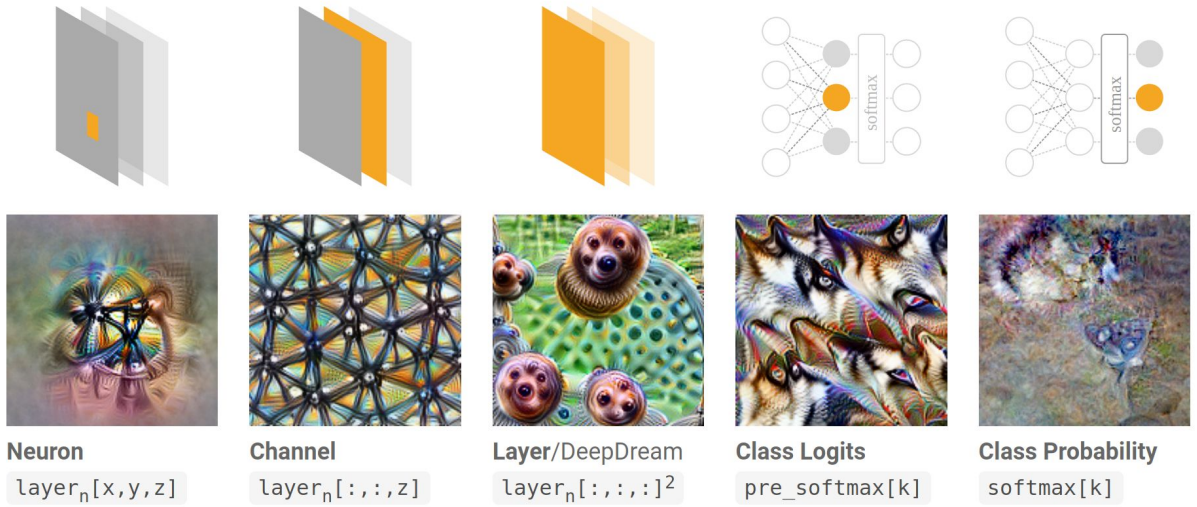


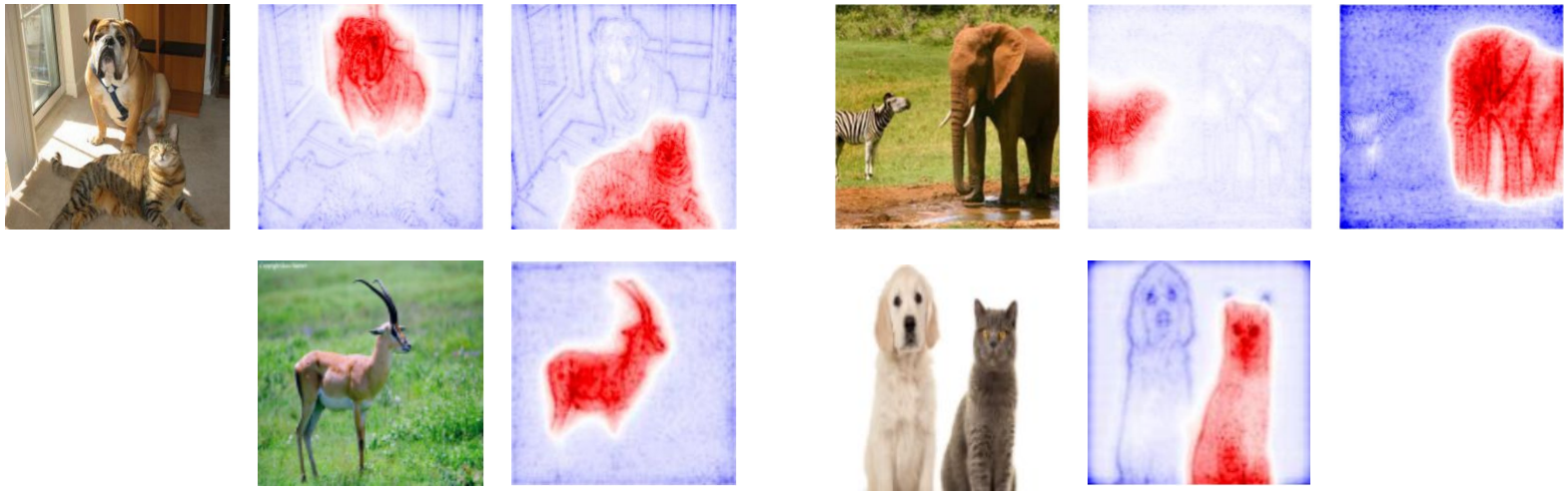
Fig. from *Feature Visualization* Olah et al. 2017, DOI:10.23915/distill.00007

VISIUM

What's going on under the hood of my ANN?

We can group interpretability methods for neural networks into **two main families**

- Identify important aspects of the **computation**
  - Obtain **per-class explanations**, in the form of heatmaps, using gradients and salient features



VISIUM

Fig.s from *Visualization of Supervised and Self-Supervised Neural Networks via Attribution Guided Factorization* Gur et al. 2019, arxiv:2012.02166

Graphs require different explanations for different tasks

Explain a prediction from a GNN as a **small subgraph** of the input graph with a **small subset of node features**

- **Node classification** and **Edge prediction:**
  - ➜ Which **combination of neighbors** most influence the current one?

    E.g. Origin of traffic jam on a specific road (edge) or intersection (node)

- **Graph classification:**
  - ➜ Which **part of the graph**, and which **properties**, determine the output?

    E.g. Which groups in a molecule characterize the physical properties

How to handle interpretability on graphs? → GNN Explainer

Explain a prediction from a GNN as a **small subgraph** of the input graph with a **small subset of node features**

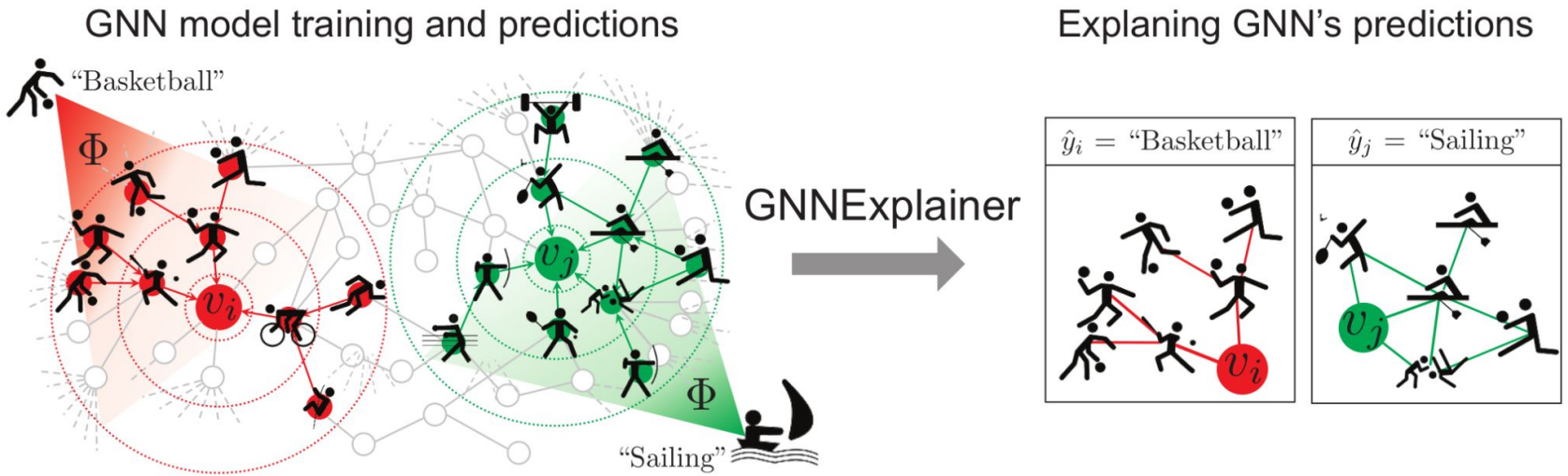


Fig. from *Gnnexplainer: Generating explanations for graph neural networks*. Ying, Rex, et al. 2019, arxiv:1903.03894

Optimizing mutual information to explain predictions

The explainer objective is to **maximize** the **mutual information (MI)** between the network prediction Y over the full computation graph G_C and the prediction conditioned over a *subgraph* G_S and *subsets of features* X_S^F

$$\max_{G_S, X_S^F} MI(Y, (G_S, X_S^F)) = H(Y) - H(y|G = G_S, X = X_S^F)$$

We can reformulate as a **cross entropy minimization** for a *graph mask M* and a *feature mask F*

$$\min_{M,F} - \sum_{c=1}^{C} \mathbf{1}_{y=c} \log P_\Phi(Y = y|G = A_c \odot \sigma(M), X = X_c \odot F)$$

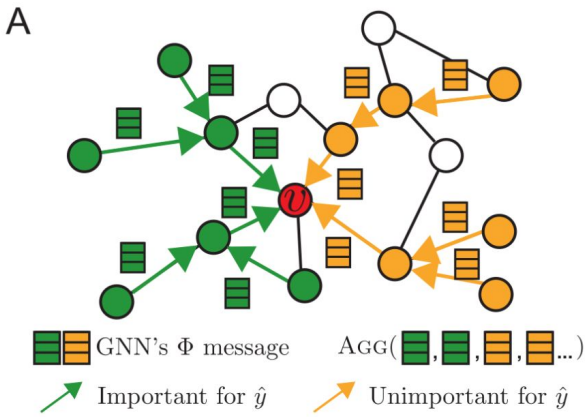
And include **penalization terms**:

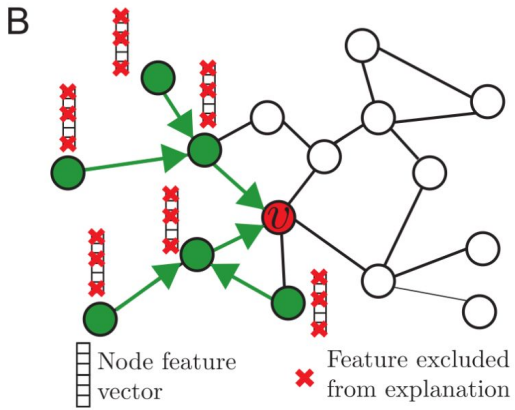
- Constraint on the **graph size** $\lambda_1 \|\sigma(M)\|_1$
- Constraint on the **graph entropy** $\lambda_2 H\big(\sigma(M)\big)$
- Constraint on the **number of features** $\lambda_3 \|\sigma(F)\|_1$

Let's get our hands dirty with GNN Explainer!

Colab Notebook

1. Implement the **Loss function**
2. Understand **GNN Explainer** workflow
3. **Explain** some graphs





VISIUM

Fig. from *Gnnexplainer: Generating explanations for graph neural networks*. Ying, Rex, et al. 2019, arxiv:1903.03894

Summary

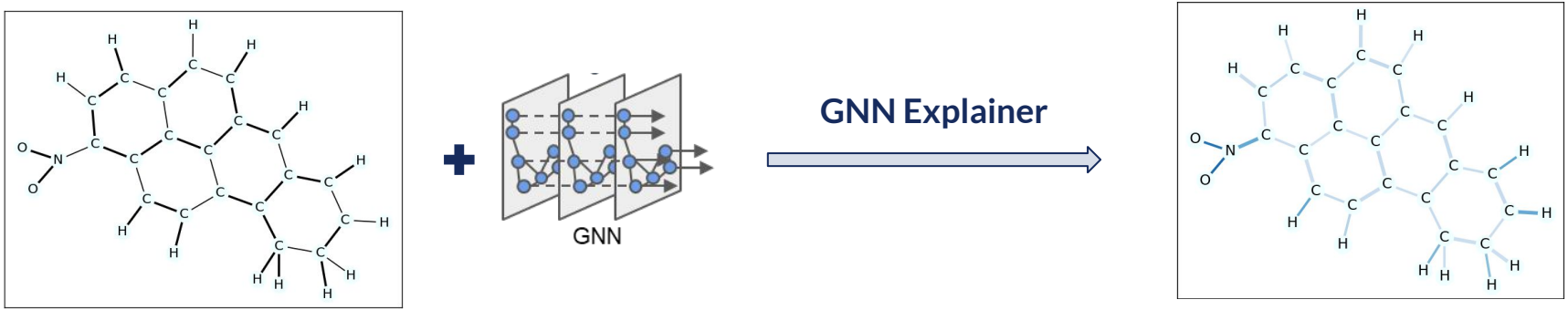
What we learned today

VISIUM

What you learned today

During the course of this workshop, we learned about:

- **Graph data** and how to represent them;
- How to do **deep learning** on graphs;
- The importance of **interpretability** in ML, and
- How to explain a GNN prediction using **GNN Explainer**.



VISIUM

Questions?

Thank you for your participation and attendance!

For more information about **Visium**

- Visium homepage
- Work with us to build something great
- Join our team

You can reach us directly

- Arnaud Dhaene
- William Cappelletti
- Camilla Casamento Tumeo

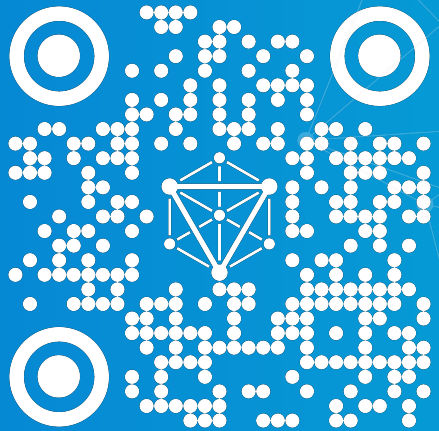
VISIUM

EPFL Innovation Park, 1015 Lausanne
Technopark Zurich, 8005 Zurich

www.visium.ch

+41 21 691 55 30

arnaud.dhaene@visium.ch



VISIUM