Machine Learning on Graphs and Sets

G. Nikolentzos and M. Vazirgiannis

LIX, École Polytechnique

ALTEGRAD 2020-2021

Random Walk Graph Neural Network (RWNN)

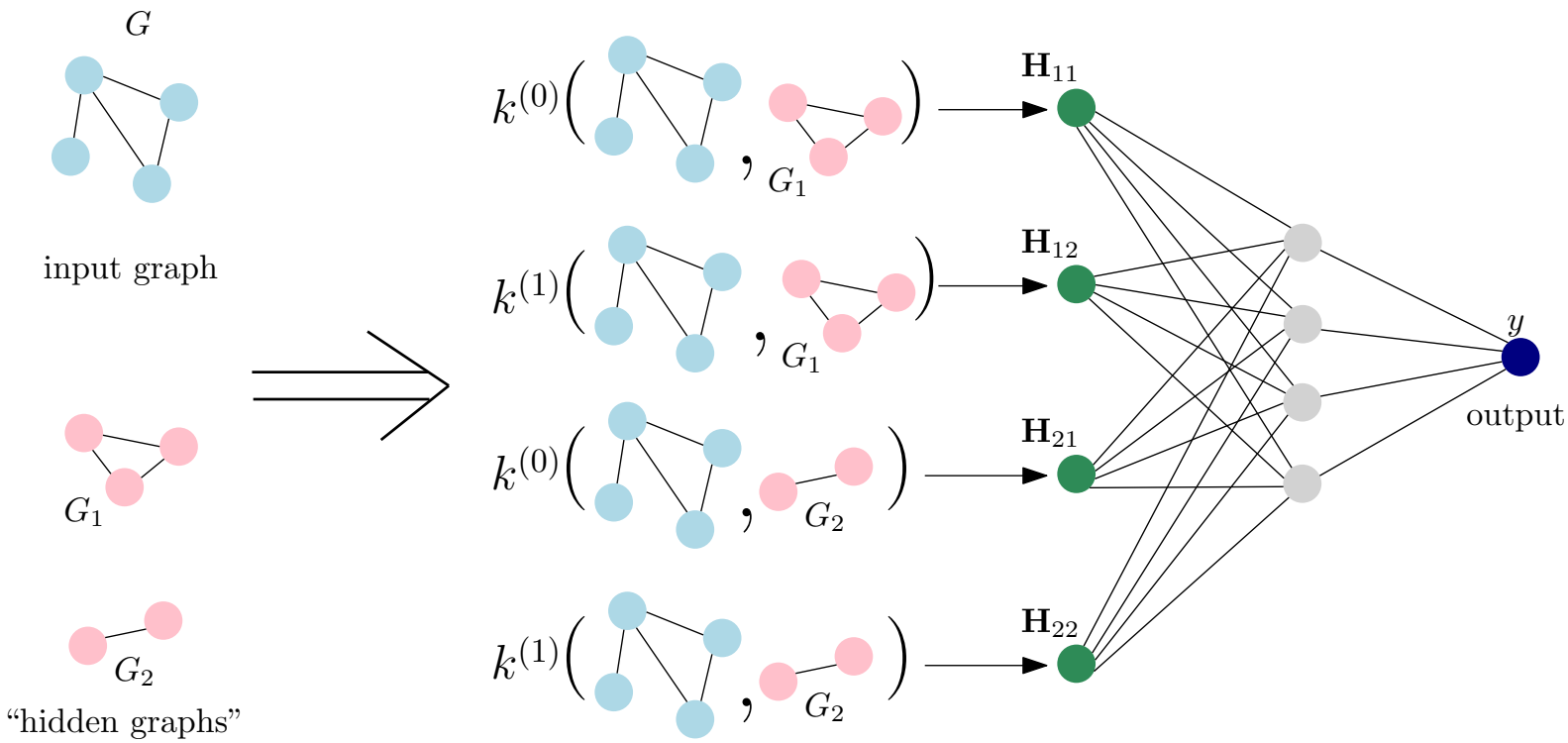
Idea: Existing graph neural networks are counter-intuitive
$\hookrightarrow$ features they learn are in the form of vectors

The Random Walk Graph Neural Network is a model

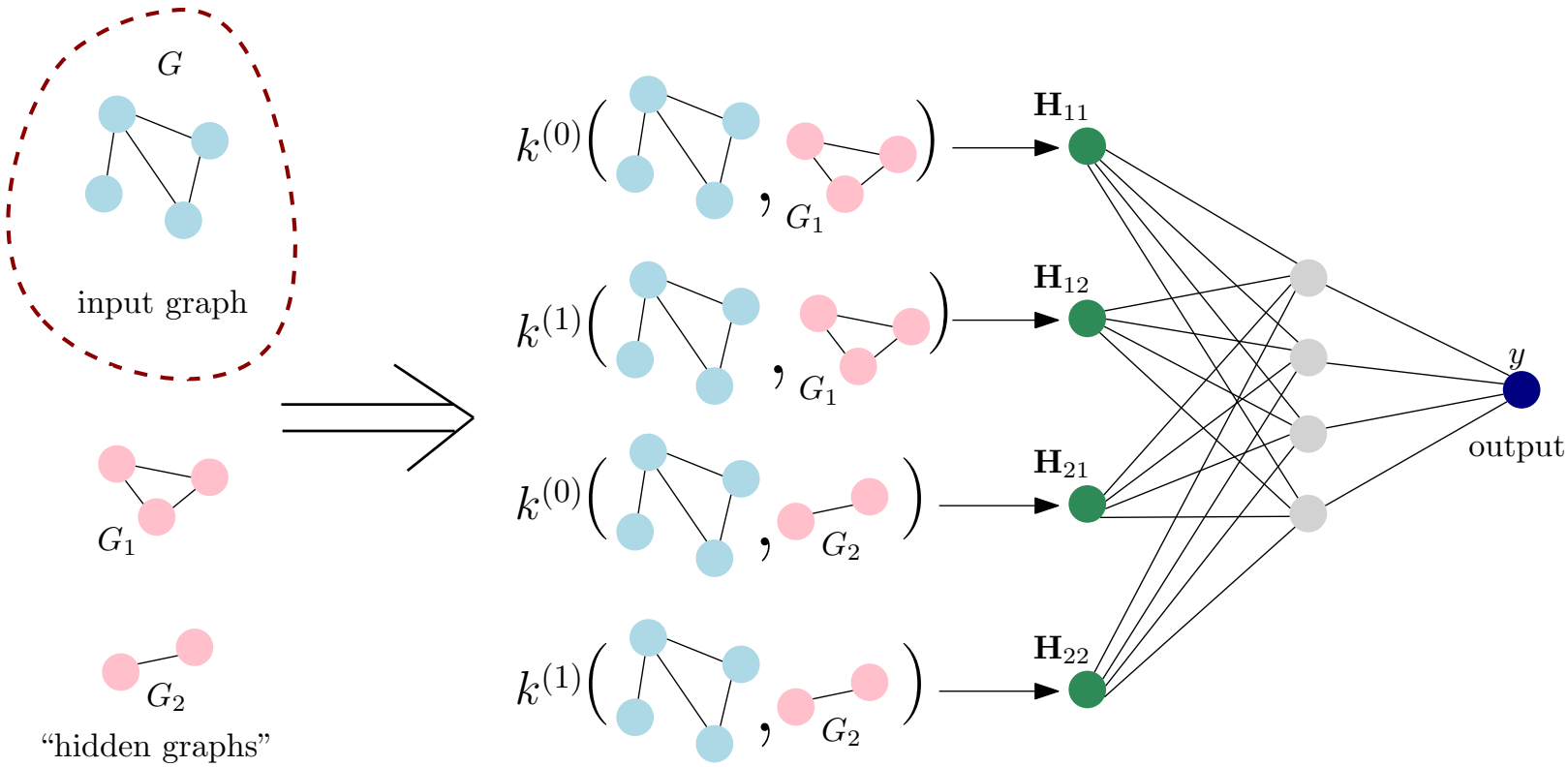
- employs a random walk kernel to learn *graph features* (in the form of fixed-size graphlets) that contribute to interpretable/intuitive graph representations
- An efficient computation scheme to reduce the time and space complexity of the proposed model

[Nikolentzos and Vazirgiannis, NeurIPS'20]

Random Walk Graph Neural Network (RWNN)

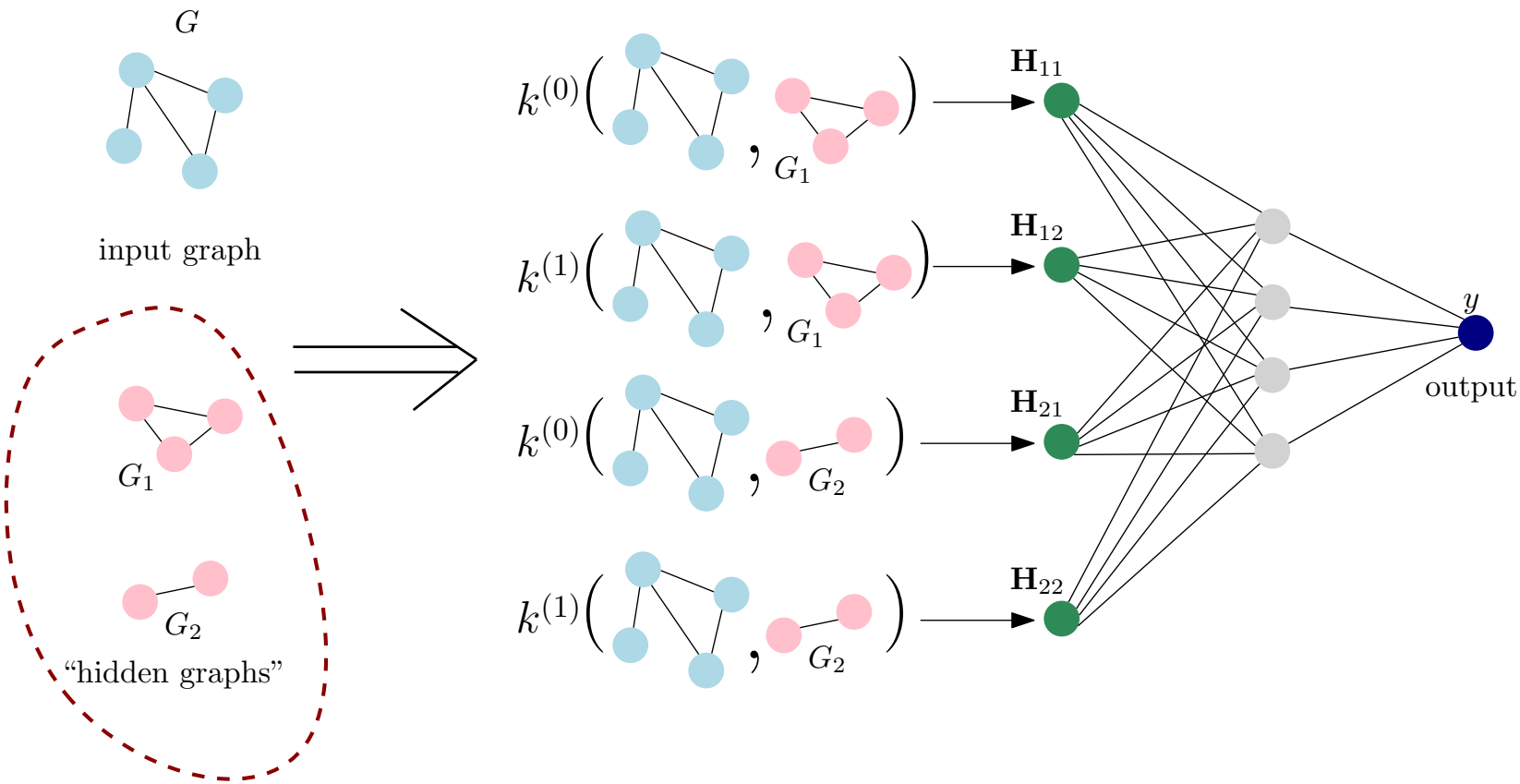


Random Walk Graph Neural Network (RWNN)



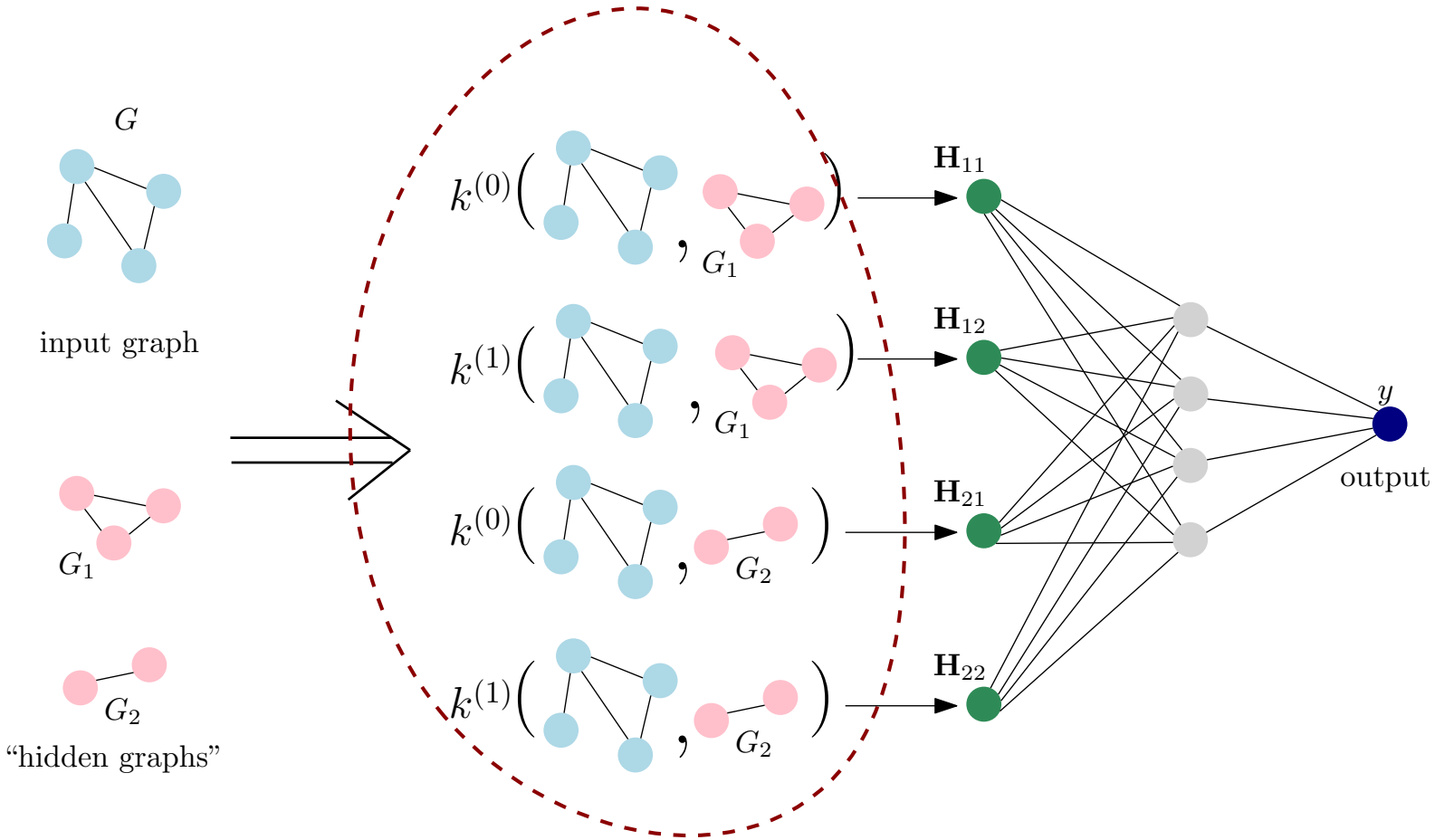
Given an input graph G

Random Walk Graph Neural Network (RWNN)



and a set of trainable "hidden graphs" $G_1, G_2, \ldots$

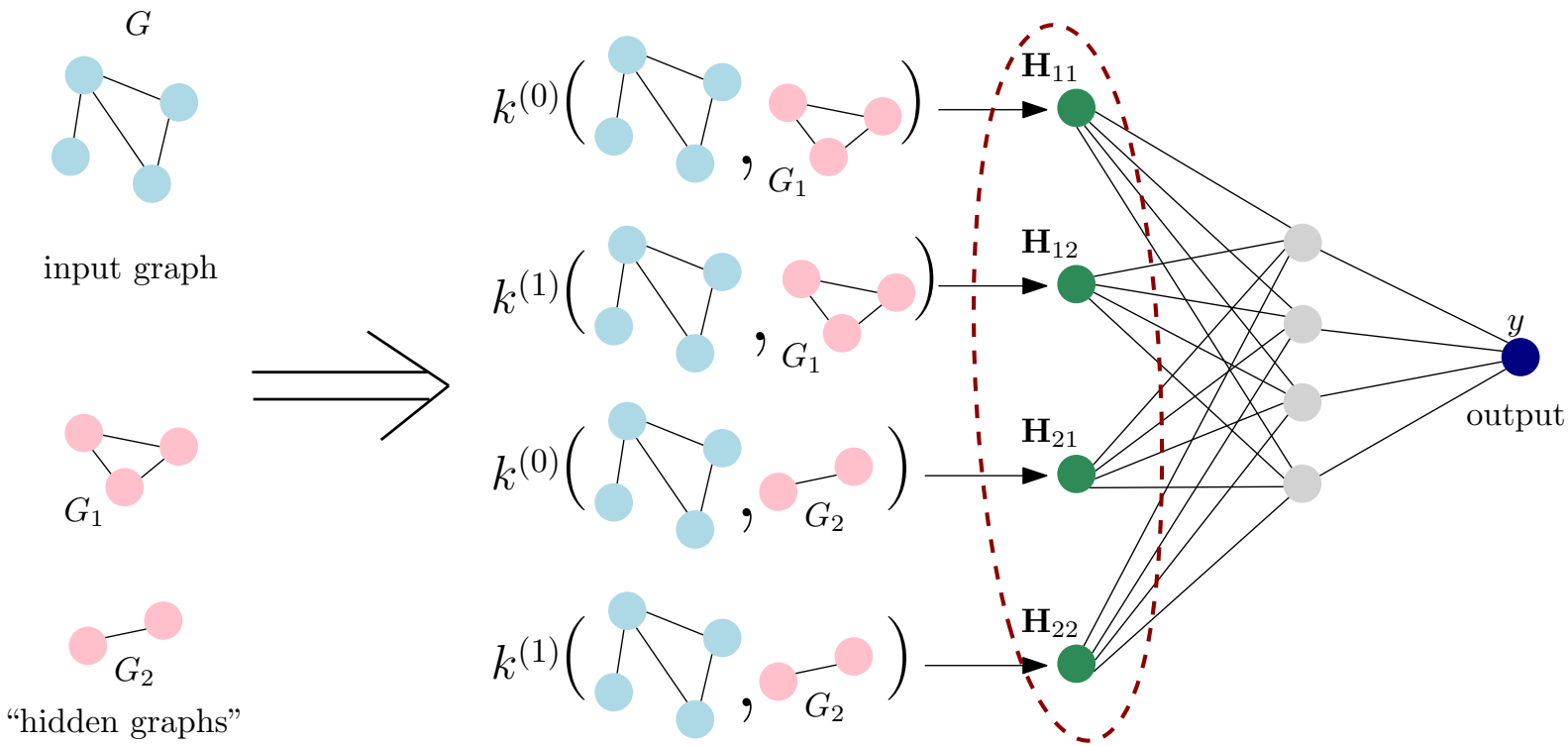
Random Walk Graph Neural Network (RWNN)



The model computes the following kernel between the input graph G and each "hidden graph" $G_{\mathtt{i}}$:

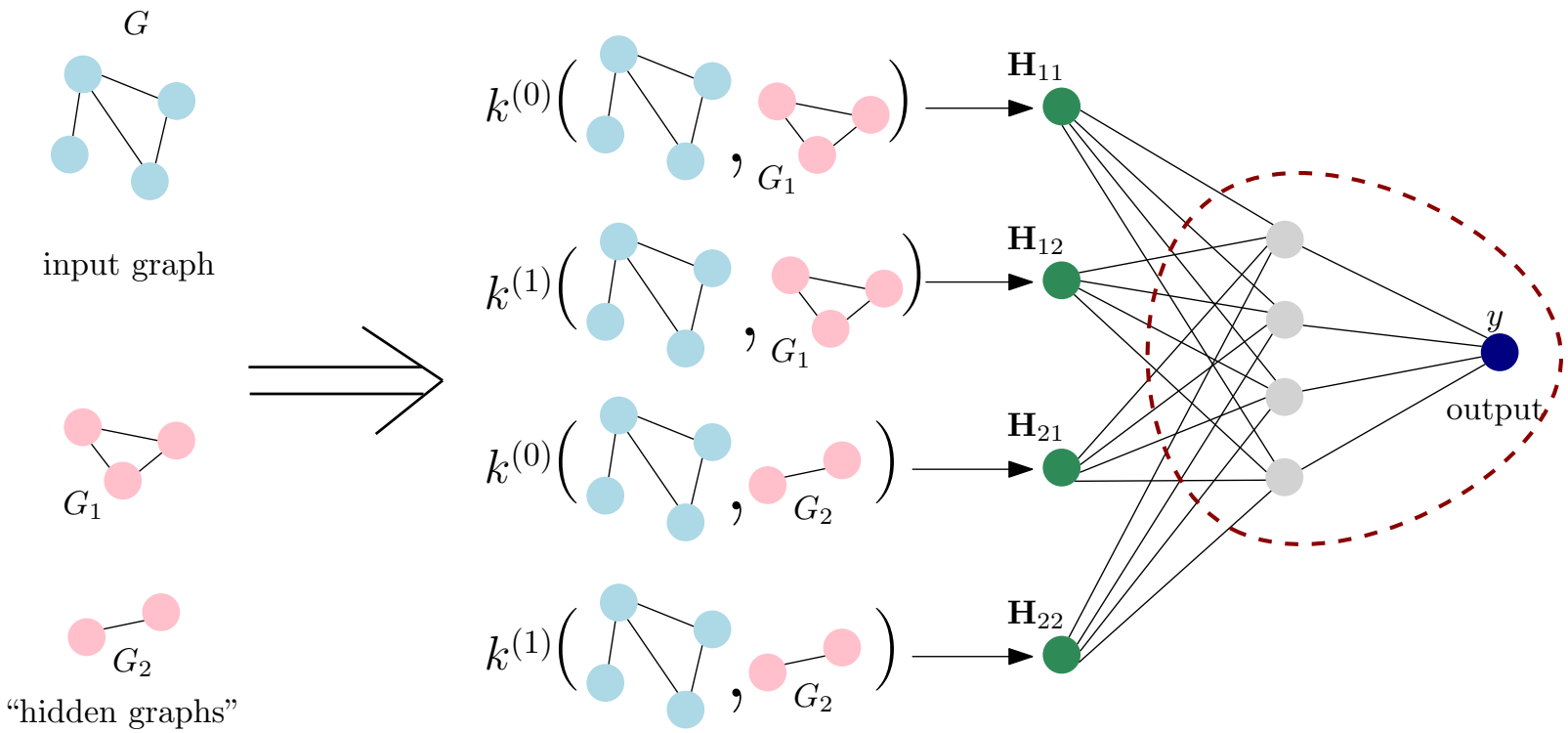
$$k^{(p)}(G, G_{\mathtt{i}}) = \sum_{i=1}^{|V_\times|} \sum_{j=1}^{|V_\times|} \mathsf{s}_i \mathsf{s}_j \left[\mathsf{A}_\times^p\right]_{ij}$$

Random Walk Graph Neural Network (RWNN)



For each input graph G, we build a matrix $\mathsf{H} \in \mathbb{R}^{N \times P+1}$ where $\mathsf{H}_{ij} = k^{(j-1)}(G, G_i)$

Random Walk Graph Neural Network (RWNN)



Matrix H is flattened and fed into a fully-connected neural network to produce the output

Experimental Evaluation - Graph Classification

We evaluated RWNN on the following standard graph classification datasets from bio/chemo-informatics and social networks

Dataset	ENZYMES	NCI1	PROTEINS	D&D	IMDB BINARY	IMDB MULTI	REDDIT BINARY	REDDIT MULTI-5K	COLLAB
Max # vertices	126	111	620	5,748	136	89	3,782	3,648	492
Min # vertices	2	3	4	30	12	7	6	22	32
Average # vertices	32.63	29.87	39.05	284.32	19.77	13.00	429.61	508.50	74.49
Max # edges	149	119	1,049	14,267	1,249	1,467	4,071	4,783	40,119
Min # edges	1	2	5	63	26	12	4	21	60
Average # edges	62.14	32.30	72.81	715.66	96.53	65.93	497.75	594.87	2,457.34
# labels	3	37	3	82	-	-	-	-	-
# graphs	600	4,110	1,113	1,178	1,000	1,500	2,000	4,999	5,000
# classes	6	2	2	2	2	3	2	5	3

Experimental Setup

Published as a conference paper at ICLR 2020

A Fair Comparison of Graph Neural Networks for Graph Classification

Federico Errica*
Department of Computer Science
University of Pisa
federico.errica@phd.unipi.it

Marco Podda*
Department of Computer Science
University of Pisa
marco.podda@di.unipi.it

Davide Bacciu*
Department of Computer Science
University of Pisa
bacciu@di.unipi.it

Alessio Micheli*
Department of Computer Science
University of Pisa
micheli@di.unipi.it

- 10-fold CV for model assessment and an inner holdout technique with a 90%/10% training/validation split for model selection
- After each model selection $\rightarrow$ train 3 times on the whole training fold, holding out a random fraction (10%) of the data to perform early stopping
- Final test fold score obtained as the mean of these 3 runs

Baselines

- Graph Kernels
  - Shortest path kernel (SP) **[Borgwardt and Kriegel, ICDM'05]**
  - Graphlet kernel (GR) **[Shervashidze et al., AISTATS'09]**
  - Weisfeiler-Lehman subtree kernel (WL) **[Shervashidze et al., JMLR'11]**

- Graph Neural Networks
  - DGCNN **[Zhang et al., AAAI'18]**
  - DiffPool **[Ying et al., NIPS'18]**
  - ECC **[Simonovsky and Komodakis, CVPR'17]**
  - GIN **[Xu et al., ICLR'19]**
  - GraphSAGE **[Hamilton et al., NIPS'17]**

Graph Classification - Real World Datasets

	MUTAG	D&D	NCI1	PROTEINS	ENZYMES
SP	80.2 (± 6.5)	**78.1** (± 4.1)	72.7 (± 1.4)	**75.3** (± 3.8)	38.3 (± 8.0)
GR	80.8 (± 6.4)	75.4 (± 3.4)	61.8 (± 1.7)	71.6 (± 3.1)	25.1 (± 4.4)
WL	84.6 (± 8.3)	**78.1** (± 2.4)	**84.8** (± 2.5)	73.8 (± 4.4)	50.3 (± 5.7)
DGCNN	84.0 (± 6.7)	76.6 (± 4.3)	76.4 (± 1.7)	72.9 (± 3.5)	38.9 (± 5.7)
DiffPool	79.8 (± 7.1)	75.0 (± 3.5)	76.9 (± 1.9)	73.7 (± 3.5)	59.5 (± 5.6)
ECC	75.4 (± 6.2)	72.6 (± 4.1)	76.2 (± 1.4)	72.3 (± 3.4)	29.5 (± 8.2)
GIN	84.7 (± 6.7)	75.3 (± 2.9)	80.0 (± 1.4)	73.3 (± 4.0)	**59.6** (± 4.5)
GraphSAGE	83.6 (± 9.6)	72.9 (± 2.0)	76.0 (± 1.8)	73.0 (± 4.5)	58.2 (± 6.0)
1-step RWNN	**89.2** (± 4.3)	77.6 (± 4.7)	71.4 (± 1.8)	74.7 (± 3.3)	56.7 (± 5.2)
2-step RWNN	88.1 (± 4.8)	76.9 (± 4.6)	73.0 (± 2.0)	74.1 (± 2.8)	57.4 (± 4.9)
3-step RWNN	88.6 (± 4.1)	77.4 (± 4.9)	73.9 (± 1.3)	74.3 (± 3.3)	57.6 (± 6.3)

	IMDB BINARY	IMDB MULTI	REDDIT BINARY	REDDIT MULTI-5K	COLLAB
SP	57.7 (± 4.1)	39.8 (± 3.7)	89.0 (± 1.0)	51.1 (± 2.2)	**79.9** (± 2.7)
GR	63.3 (± 2.7)	39.6 (± 3.0)	76.6 (± 3.3)	38.1 (± 2.3)	71.1 (± 1.4)
WL	**72.8** (± 4.5)	**51.2** (± 6.5)	74.9 (± 1.8)	49.6 (± 2.0)	78.0 (± 2.0)
DGCNN	69.2 (± 3.0)	45.6 (± 3.4)	87.8 (± 2.5)	49.2 (± 1.2)	71.2 (± 1.9)
DiffPool	68.4 (± 3.3)	45.6 (± 3.4)	89.1 (± 1.6)	53.8 (± 1.4)	68.9 (± 2.0)
ECC	67.7 (± 2.8)	43.5 (± 3.1)	OOR	OOR	OOR
GIN	71.2 (± 3.9)	48.5 (± 3.3)	89.9 (± 1.9)	**56.1** (± 1.7)	75.6 (± 2.3)
GraphSAGE	68.8 (± 4.5)	47.6 (± 3.5)	84.3 (± 1.9)	50.0 (± 1.3)	73.9 (± 1.7)
1-step RWNN	70.8 (± 4.8)	47.8 (± 3.8)	**90.4** (± 1.9)	51.7 (± 1.5)	71.7 (± 2.1)
2-step RWNN	70.6 (± 4.4)	48.8 (± 2.9)	90.3 (± 1.8)	51.7 (± 1.4)	71.3 (± 2.1)
3-step RWNN	70.7 (± 3.9)	47.8 (± 3.5)	89.7 (± 1.2)	53.4 (± 1.6)	71.9 (± 2.5)

Outline

Sets

What is a set?

A set is a well-defined collection of distinct objects

Complex data sets decomposed into sets of simpler objects

↪ NLP: documents as sets of word embeddings

↪ Graph Mining: graphs as sets of node embeddings

↪ Computer Vision: images as sets of local features

Machine learning on sets has attracted a lot of attention recently

- Set classification
- Set regression

Set Classification

$S_1 = \{1, 4, 2\}$

$y_1 = -1$

$S_2 = \{5, 0, 8, 10\}$

$y_2 = -1$

$S_6 = \{2, 6, 3, 5\}$

$y_6 = ???$

$S_3 = \{3, 7\}$

$y_3 = 1$

$S_4 = \{3, 5, 6\}$

$y_4 = 1$

$S_7 = \{4, 2, 5\}$

$y_7 = ???$

$S_5 = \{5\}$

$y_5 = -1$

- Let $\mathcal{X}$ be a set
- Input data $S \in 2^{\mathcal{X}}$
- Output $y \in \{-1, 1\}$
- Training set $\{(S_1, y_1), \ldots, (S_n, y_n)\}$
- Goal: estimate a function $f : 2^{\mathcal{X}} \to \in \{-1, 1\}$ to predict y from $f(S)$

Limitations of Standard Machine Learning Models

Conventional machine learning models cannot handle sets:

- expect fixed dimensional data instances
  $\hookrightarrow$ sets allowed to vary in the number of elements

- not invariant to permutations of features
  - a learning algorithm for sets needs to produce identical representations for any permutation of the elements of an input set
  - for instance, a model f needs to satisfy the following for any permutation π of the set's elements:

$$f(\{x_1, \ldots, x_M\}) = f(\{x_{\pi(1)}, \ldots, x_{\pi(M)}\})$$

Outline

Neural Networks for Sets

Recent approaches:

- unordered sets $\rightarrow$ ordered sequences $\rightarrow$ RNN **[Vinyals et al., ICLR'16]**
- DeepSets **[Zaheer et al., NIPS'17]** and PointNet **[Qi et al., CVPR'17]** transform the vectors of the sets into new representations, then apply permutation-invariant functions
- PointNet++ **[Qi et al., NIPS'17]** and SO-Net **[Li et al., CVPR'18]** apply PointNet hierarchically in order to better capture local structures
- Set Transformer **[Lee et al., ICML'19]**, a neural network that uses self-attention to model interactions among the elements of the input set
- RepSet **[Skianis et al., AISTATS'20]**, a neural network that generates representations for sets by comparing them against some trainable sets

SOTA in supervised learning tasks:

- regression: population statistic estimation, sum of digits
- classification: point cloud classification, outlier detection

Theorem (Zaheer et al., NIPS'17)

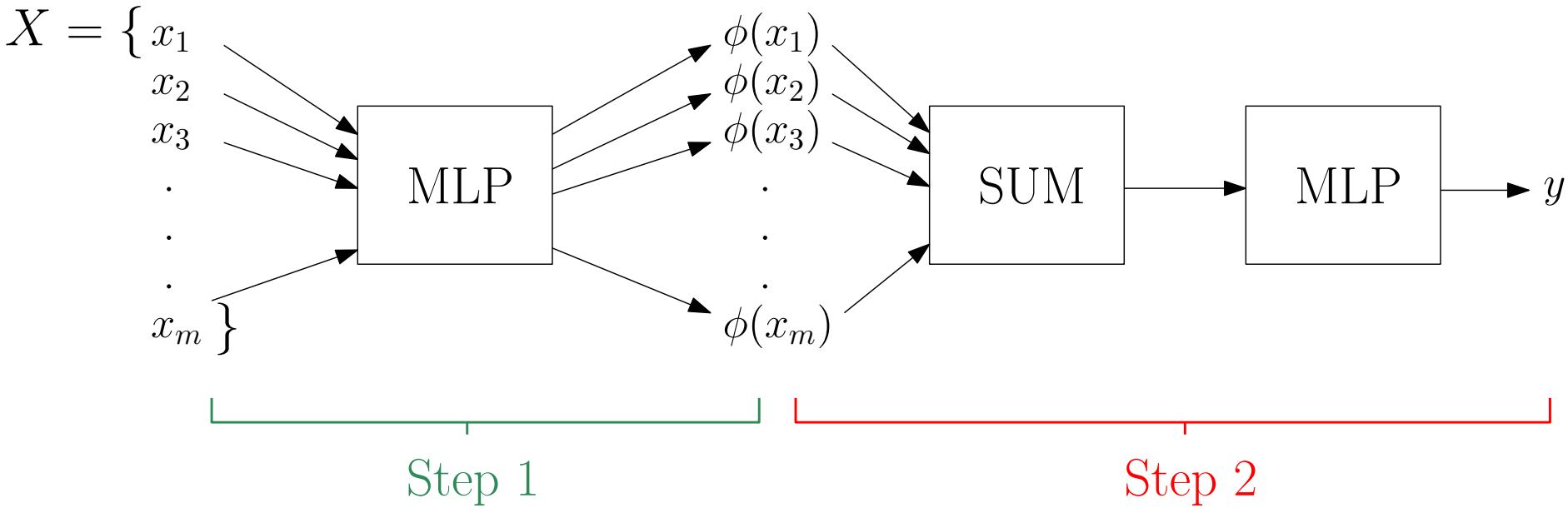
If $\mathfrak{X}$ is a countable set and $\mathcal{Y} = \mathbb{R}$, then a function $f(X)$ operating on a set X having elements from $\mathfrak{X}$ is a valid set function, i.e., invariant to the permutation of instances in X, if and only if it can be decomposed in the form $\rho(\sum_{x \in X} \phi(x))$, for suitable transformations ϕ and ρ.

DeepSets achieves permutation invariance by replacing ϕ and ρ with multi-layer perceptrons (universal approximators)

DeepSets consist of the following two steps:

1. Each element x_i of each set is transformed (possibly by several layers) into some representation $\phi(x_i)$
2. The representations $\phi(x_i)$ are added up and the output is processed using the ρ network in the same manner as in any deep network (e.g., fully connected layers, nonlinearities, etc.)

DeepSets



Step 1: The elements $x_1, \ldots, x_m$ of the input set X are transformed into representations $\phi(x_1), \ldots, \phi(x_m)$

Step 2: A representation for the entire set is produced as $z_X = \phi(x_1) + \ldots + \phi(x_m)$ and is also transformed as follows $y = \rho(z_X)$ to produce the output

Experiments - Point Cloud Classification

Objective: classify point-clouds
$\hookrightarrow$ point-clouds are sets of low-dimensional vectors (typically 3-dimensional vectors representing the x, y, z-coordinates of objects)

Dataset: ModelNet40 $\rightarrow$ consists of 3-dimensional representations of $9,843$ training and $2,468$ test instances belonging to 40 classes of objects

Setup: point-clouds directly passed on to DeepSets

Model	Instance Size	Representation	Accuracy
3DShapeNets [25]	30^3	voxels (using convolutional deep belief net)	77%
VoxNet [26]	32^3	voxels (voxels from point-cloud + 3D CNN)	83.10%
MVCNN [21]	$164\times164\times12$	multi-vew images (2D CNN + view-pooling)	90.1%
VRN Ensemble [27]	32^3	voxels (3D CNN, variational autoencoder)	95.54%
3D GAN [28]	64^3	voxels (3D CNN, generative adversarial training)	83.3%
DeepSets	$\mathbf{5000\times3}$	point-cloud	$90\pm.3\%$
DeepSets	$\mathbf{100\times3}$	point-cloud	$82\pm2\%$

Experiments - Text Concept Retrieval

Objective: retrieve words belonging to a "concept" given few words from the concept

Example: given the set of words $\{tiger, lion, cheetah\}$, retrieve other related words like jaguar and puma, which all belong to the concept of big cats

Setup: query word added to set and new set fed to DeepSet which produces a score

Method	LDA-1k (Vocab = 17k)					LDA-3k (Vocab = 38k)					LDA-5k (Vocab = 61k)				
	Recall (%)			**MRR**	**Med.**	**Recall (%)**			**MRR**	**Med.**	**Recall (%)**			**MRR**	**Med.**
	@10	**@100**	**@1k**			**@10**	**@100**	**@1k**			**@10**	**@100**	**@1k**		
Random	0.06	0.6	5.9	0.001	8520	0.02	0.2	2.6	0.000	28635	0.01	0.2	1.6	0.000	30600
Bayes Set	1.69	11.9	37.2	0.007	2848	2.01	14.5	36.5	0.008	3234	1.75	12.5	34.5	0.007	3590
w2v Near	**6.00**	**28.1**	**54.7**	0.021	**641**	4.80	21.2	43.2	0.016	2054	4.03	16.7	35.2	0.013	6900
NN-max	4.78	22.5	53.1	0.023	779	5.30	24.9	54.8	0.025	672	4.72	21.4	47.0	0.022	1320
NN-sum-con	4.58	19.8	48.5	0.021	1110	5.81	27.2	60.0	**0.027**	453	4.87	23.5	53.9	0.022	731
NN-max-con	3.36	16.9	46.6	0.018	1250	5.61	25.7	57.5	0.026	570	4.72	22.0	51.8	0.022	877
DeepSets	5.53	24.2	54.3	**0.025**	696	**6.04**	**28.5**	**60.7**	**0.027**	**426**	**5.54**	**26.1**	**55.5**	**0.026**	**616**

Experiments - Image Tagging

Objective: retrieve all relevant tags corresponding to an image

Setup: features of the image are concatenated to the embeddings of the tags, and then the whole set is passed on to DeepSets to assign a single score to the set

Method	ESP game				IAPRTC-12.5			
	P	R	F1	N+	P	R	F1	N+
Least Sq.	35	19	25	215	40	19	26	198
MBRM	18	19	18	209	24	23	23	223
JEC	24	19	21	222	29	19	23	211
FastTag	46	22	30	**247**	**47**	26	34	**280**
Least Sq.(D)	**44**	32	**37**	232	46	30	36	218
FastTag(D)	**44**	32	**37**	229	46	**33**	**38**	254
DeepSets	39	**34**	36	246	42	31	36	247

Experiments - Set Anomaly Detection

Objective: find the anomalous face in each set

Architecture: consists of 9 2d-convolution and max-pooling layers followed by the DeepSets model, and a softmax layer that assigns a probability value to each set member

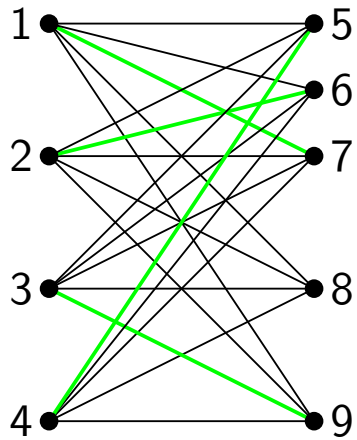
RepSet

- A permutation invariant neural network for sets
- Generates a number of "hidden sets" and it compares the input set with these sets using a network flow algorithm (e.g., bipartite matching)
- The outputs of the network flow algorithm form the penultimate layer and are fed to a fully-connected layer which produces the output
- The model is end-to-end trainable $\rightarrow$ "hidden sets" are updated during training
- For large sets, solving the flow problems can become prohibitive ☹
  $\hookrightarrow$ ApproxRepSet is a relaxed formulation (also permutation invariant) that scales to very large datasets

Permutation Invariant Layer

- A layer whose output is the same regardless of the ordering of the input's elements
- Contains m "hidden sets" $Y_1, Y_2, \ldots, Y_m$ of d-dimensional vectors
  $\hookrightarrow$ may have different cardinalities and their components are trainable
- Measure similarity between input set and each one of the "hidden sets" by comparing their building blocks, i.e., their elements $\rightarrow$ bipartite matching

1	1.76	0.40	0.97
2	2.24	1.86	−0.97
3	0.95	−0.15	−0.10
4	0.41	0.14	1.45



5	0.52	0.08	1.62
6	2.14	1.72	−1.05
7	1.55	−0.45	0.88
8	−0.34	−1.26	0.24
9	1.08	−0.21	−0.09

Figure: Example of a bipartite graph generated from 2 sets of 3-dimensional vectors, and of its maximum matching M. Green color indicates that an edge belongs to M

RepSet - Bipartite Matching Problem

- Input set $X = \{v_1, v_2, \ldots, v_{|X|}\}$ where $v_1, \ldots, v_{|X|}$ vectors
- "Hidden set" $Y = \{u_1, u_2, \ldots, u_{|Y|}\}$
- Maximum matching between the elements of X and Y by solving the following linear program:

$$\max \sum_{i=1}^{|X|} \sum_{j=1}^{|Y|} z_{ij} f(v_i, u_j) \text{ subject to:}$$

$$\sum_{i=1}^{|X|} z_{ij} \leq 1 \quad \forall j \in \{1, \ldots, |Y|\}$$

$$\sum_{j=1}^{|Y|} z_{ij} \leq 1 \quad \forall i \in \{1, \ldots, |X|\}$$

$$z_{ij} \geq 0 \quad \forall i \in \{1, \ldots, |X|\}, \forall j \in \{1, \ldots, |Y|\}$$

where $f(v_i, u_j)$ is a differentiable function (e.g., inner product), and $z_{ij} = 1$ if component i of X is assigned to component j of Y_i, and 0 otherwise

RepSet - Produced Set Representations

Given an input set X and the m "hidden sets" $Y_1, Y_2, \ldots, Y_m$

1. formulate m different bipartite matching problems
2. by solving all m problems, end up with an m-dimensional vector $\mathbf{x} \rightarrow$ hidden representation of set X
3. this m-dimensional vector can be used as features for different machine learning tasks (e.g., set regression, set classification)
   $\hookrightarrow$ For instance, in the case of a set classification problem with $|\mathcal{C}|$classes, output is computed as

$$\mathrm{p} = \mathrm{softmax}(\mathrm{W}\mathrm{x} + \mathrm{b})$$

   where W is a matrix of trainable parameters and b is the bias term

RepSet - Architecture

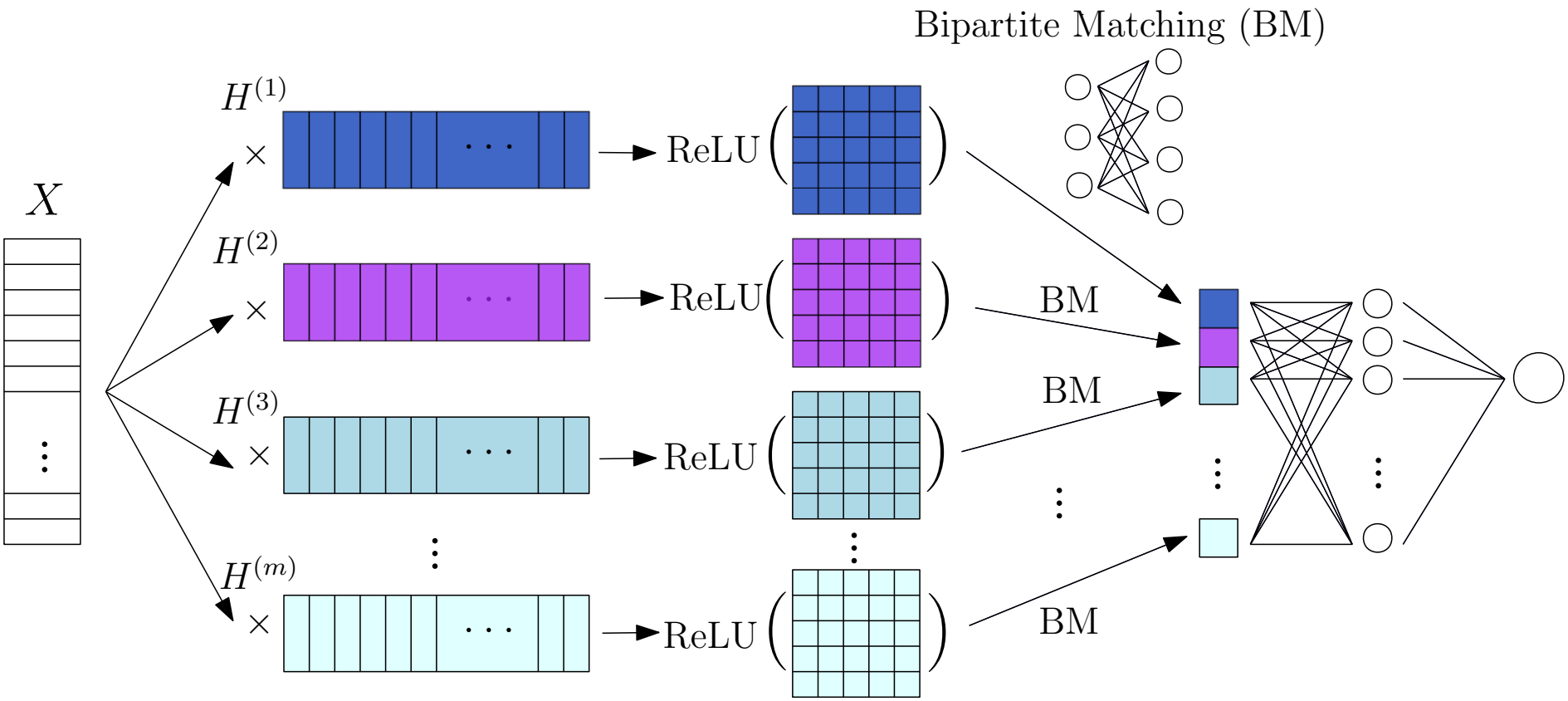


Figure: Each element of the input set is compared with the elements of all "hidden sets", and the emerging matrices serve as the input to bipartite matching. The values of the BM problems correspond to the representation of the input set.

RepSet - Relaxed Variant (ApproxRepSet)

- Input set $X = \{v_1, v_2, \ldots, v_{|X|}\}$ where $v_1, \ldots, v_{|X|}$ vectors
- "Hidden set" $Y = \{u_1, u_2, \ldots, u_{|Y|}\}$
- Identify which of the two sets has the highest cardinality.
- If $|X| \geq |Y|$, we solve the following problem:

$$\max \sum_{i=1}^{|X|} \sum_{j=1}^{|Y|} z_{ij} f(v_i, u_j) \text{ subject to:}$$

$$\sum_{i=1}^{|X|} z_{ij} \leq 1 \quad \forall j \in \{1, \ldots, |Y|\}$$

$$z_{ij} \geq 0 \quad \forall i \in \{1, \ldots, |X|\}, \forall j \in \{1, \ldots, |Y|\}$$

- Multiple elements of X (the bigger set) can be matched with the same element of Y
- Optimal solution matches an element y_j of Y with x_i of X if $f(v_i, u_j)$ is positive and $f(v_i, u_j) = \max_k f(v_k, u_j)$

Experimental Evaluation

- Text categorization
  - given a document, the input to the model is the set of embeddings of its terms
  - standard text categorization datasets (TWITTER, BBCSPORT etc.)

- Graph classification
  - represent each graph as a set of vectors (i.e., the embeddings of its nodes)
  - node embeddings are extracted by struc2vec **[Ribeiro et al., KDD'17]**
  - datasets derived from bioinformatics (MUTAG, PROTEINS) and social networks (IMDB-BINARY, -MULTI, REDDIT-BINARY)

Experiments - Text classification

	BBCSPORT	TWITTER	RECIPE	OHSUMED	CLASSIC	REUTERS	AMAZON	20NG
WMD	4.60 ± 0.70	28.70 ± 0.60	42.60 ± 0.30	44.50	**2.88** ± 0.10	3.50	7.40 ± 0.30	26.80
S-WMD	2.10 ± 0.50	27.50 ± 0.50	39.20 ± 0.30	34.30	3.20 ± 0.20	3.20	5.80 ± 0.10	26.80
DeepSets	25.45 ± 20.1	29.66 ± 1.62	70.25 ± 0.00	71.53	5.95 ± 1.50	10.00	8.58 ± 0.67	38.88
NN-mean	10.09 ± 2.62	31.56 ± 1.53	64.30 ± 7.30	45.37	5.35 ± 0.75	11.37	13.66 ± 3.16	38.40
NN-max	2.18 ± 1.75	30.27 ± 1.26	43.47 ± 1.05	35.88	4.21 ± 0.11	4.33	7.55 ± 0.63	32.15
NN-attention	4.72 ± 0.97	29.09 ± 0.62	43.18 ± 1.22	**31.36**	4.42 ± 0.73	3.97	6.92 ± 0.51	28.73
Set-Transformer	4.18 ± 1.23	27.79 ± 0.47	42.54 ± 1.35	35.68	5.23 ± 0.52	4.52	7.18 ± 0.44	30.01
RepSet	**2.00** ± 0.89	**25.42** ± 1.10	**38.57** ± 0.83	33.88	3.38 ± 0.50	3.15	**5.29** ± 0.28	**22.98**
ApproxRepSet	4.27 ± 1.73	27.40 ± 1.95	40.94 ± 0.40	35.94	3.76 ± 0.45	**2.83**	5.69 ± 0.40	23.82

Table: Classification test error of the proposed architecture and baselines on 8 TC datasets.

Hidden set	Terms similar to elements of hidden sets	Terms similar to centroids of hidden sets
1	chelsea, football, striker, club, champions	footballing
2	qualify, madrid, arsenal, striker, united, france	ARSENAL_Wenger
3	olympic, athlete, olympics, sport, pentathlon	Olympic_Medalist
4	penalty, cup, rugby, coach, goal	rugby
5	match, playing, batsman, batting, striker	batsman

Table: Terms of the employed pre-trained model that are most similar to the elements and centroids of 5 hidden sets.

Experiments - Graph Classification

	MUTAG	PROTEINS	IMDB BINARY	IMDB MULTI	REDDIT BINARY
PSCN $k = 10$	88.95 ($\pm$ 4.37)	75.00 ($\pm$ 2.51)	71.00 ($\pm$ 2.29)	45.23 ($\pm$ 2.84)	86.30 ($\pm$ 1.58)
Deep GR	82.66 ($\pm$ 1.45)	71.68 ($\pm$ 0.50)	66.96 ($\pm$ 0.56)	44.55 ($\pm$ 0.52)	78.04 ($\pm$ 0.39)
EMD	86.11 ($\pm$ 0.84)	-	-	-	-
DGCNN	85.80 ($\pm$ 1.70)	75.50 ($\pm$ 0.90)	70.03 ($\pm$ 0.86)	47.83 ($\pm$ 0.85)	-
SAEN	84.99 ($\pm$ 1.82)	75.31 ($\pm$ 0.70)	71.59 ($\pm$ 1.20)	48.53 ($\pm$ 0.76)	87.22 ($\pm$ 0.80)
RetGK	**90.30** ($\pm$ 1.10)	76.20 ($\pm$ 0.50)	72.30 ($\pm$ 0.60)	48.70 ($\pm$ 0.60)	**92.60** ($\pm$ 0.30)
DiffPool	-	**76.25**	-	-	-
DeepSets	86.26 ($\pm$ 1.09)	60.82 ($\pm$ 0.79)	69.84 ($\pm$ 0.64)	47.62 ($\pm$ 1.18)	52.01 ($\pm$ 1.47)
NN-mean	87.55 ($\pm$ 0.98)	73.00 ($\pm$ 1.21)	71.48 ($\pm$ 0.48)	49.92 ($\pm$ 0.82)	84.57 ($\pm$ 0.84)
NN-max	85.84 ($\pm$ 0.99)	71.05 ($\pm$ 0.54)	69.56 ($\pm$ 0.91)	48.28 ($\pm$ 0.43)	80.98 ($\pm$ 0.79)
NN-attention	85.92 ($\pm$ 1.16)	74.48 ($\pm$ 0.22)	**72.40** ($\pm$ 0.45)	49.56 ($\pm$ 0.47)	88.74 ($\pm$ 0.53)
Set-Transformer	87.71 ($\pm$ 1.14)	59.62 ($\pm$ 1.42)	71.21 ($\pm$ 1.28)	**50.25** ($\pm$ 0.74)	83.79 ($\pm$) 0.83
RepSet	88.63 ($\pm$ 0.86)	73.04 ($\pm$ 0.42)	**72.40** ($\pm$ 0.73)	49.93 ($\pm$ 0.60)	87.45 ($\pm$ 0.86)
ApproxRepSet	86.33 ($\pm$ 1.48)	70.74 ($\pm$ 0.85)	71.46 ($\pm$ 0.91)	48.92 ($\pm$ 0.28)	80.30 ($\pm$ 0.56)

Table: Classification accuracy ($\pm$ standard deviation) of the proposed architecture and the baselines on the 5 graph classification datasets.

THANK YOU !

Acknowledgements
G. Salha

http://www.lix.polytechnique.fr/dascim/