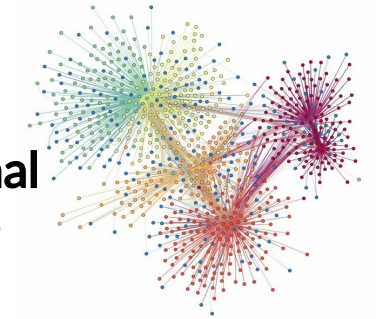
Redes Convolucionais de Grafos

Fundamentos, Aspectos de Pesquisa e Aplicações

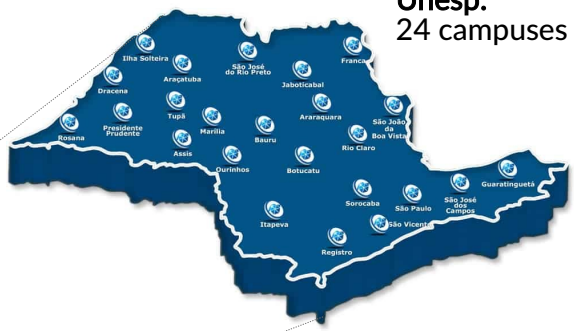
Prof. Daniel Carlos Guimarães Pedronette
Universidade Estadual Paulista (UNESP)

Graph Convolutional Networks



Escola Avançada USP
Big Data Analysis

About UNESP:



São Paulo
State University

unesp

THE WORLD UNIVERSITY RANKINGS

Young University Rankings 2024: 124
Brazil: 1st

About Me:

Daniel Carlos Guimarães Pedronette

Year	Degree	Institution
2012	⋆ PhD in Computer Science	Institute of Computing (IC) - UNICAMP
2008	Master's in Computer Science	Institute of Computing (IC) - UNICAMP
2005	Graduation in Computer Science	Inst. of Geo. and Exact Sciences (IGCE) - UNESP
2000	Technical Course - Informatics	Colégio Técnico de Limeira (COTIL) - UNICAMP

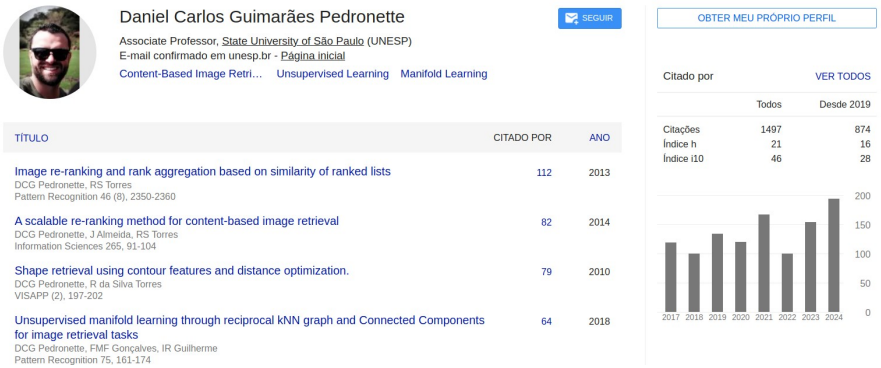
⋆ PhD Research:

- Information Retrieval
- Unsupervised Learning

About Me:

Daniel Carlos Guimarães Pedronette

- Assistant Professor - UNESP (2013)
- Associate Professor - UNESP (2019)
- Associate Editor - Pattern Reconition - Elsevier (since 2019)



Daniel Carlos Guimarães Pedronette

SEGUIR

Associate Professor, State University of São Paulo (UNESP)
E-mail confirmado em unesp.br - Página inicial
Content-Based Image Retri… Unsupervised Learning Manifold Learning

TÍTULO	CITADO POR	ANO
Image re-ranking and rank aggregation based on similarity of ranked lists DCG Pedronette, RS Torres Pattern Recognition 46 (8), 2350-2360	112	2013
A scalable re-ranking method for content-based image retrieval DCG Pedronette, J Almeida, RS Torres Information Sciences 265, 91-104	82	2014
Shape retrieval using contour features and distance optimization. DCG Pedronette, R da Silva Torres VISAPP (2), 197-202	79	2010
Unsupervised manifold learning through reciprocal kNN graph and Connected Components for image retrieval tasks DCG Pedronette, FMF Gonçalves, IR Guilherme Pattern Recognition 75, 161-174	64	2018

OBTER MEU PRÓPRIO PERFIL

Citado por	Todos	Desde 2019
Citações	1497	874
Índice h	21	16
Índice i10	46	28

Research Interests:

Main Research Areas:

- Information Retrieval
- Rank-based Models
- Content-Based Image Retrieval
- Unsupervised Learning
  - Affinity Learning
  - Clustering
- **Semi-Supervised Learning**
- **Graph Embedding and Representation Learning**
- **Graph Convolutional Networks**
- Natural Language Processing

Research Projects

CNPq - Brazilian National Council for Scientific and Technological Development:

- Research Projects
  - **Graph Convolutional Networks for Self-Supervised and Semi-Supervised Learning**
  - Coordinator: Prof. Daniel Pedronette
  - Period: 2022-2025

CNPq
Conselho Nacional de Desenvolvimento Científico e Tecnológico

Graph Neural Networks (GNN) - Taxonomy

Graph Neural Networks (GNN) - Taxonomy

TABLE II: Taxonomy and representative publications of Graph Neural Networks (GNNs)

Category		Publications
Recurrent Graph Neural Networks (RecGNNs)		[15], [16], [17], [18]
Convolutional Graph Neural Networks (ConvGNNs)	Spectral methods	[19], [20], [21], [22], [23], [40], [41]
	Spatial methods	[24], [25], [26], [27], [42], [43], [44] [45], [46], [47], [48], [49], [50], [51] [52], [53], [54], [55], [56], [57], [58]
Graph Autoencoders (GAEs)	Network Embedding	[59], [60], [61], [62], [63], [64]
	Graph Generation	[65], [66], [67], [68], [69], [70]
Spatial-temporal Graph Neural Networks (STGNNs)		[71], [72], [73], [74], [75], [76], [77]

A Comprehensive Survey on Graph Neural Networks. `https://arxiv.org/pdf/1901.00596`

GCNs: Ideias e Fundamentos

- **Graph Convolutional Networks: Ideias e Fundamentos**
  - Convolução e Filtragem de Imagens
  - Convolutional Neural Networks (CNNs)
  - Grafos e Domínios Não Euclidianos
  - **Graph Convolutional Networks (GCNs)**
  - GCNs for Semi-Supervised Classification

Convolução e Filtragem de Imagens

- O conceito de **Convolução** está associado à **Filtragem de Imagens**
- Um filtro **passa-baixas** atenua as altas frequências que estão relacionadas com a informação de detalhes da imagem.
- Um filtro **passa-altas** realça as altas frequências e são normalmente usados para realçar os detalhes na imagem.
- Um filtro **passa-faixa** seleciona um intervalo de frequências do sinal para ser realçado.

Filtragem no Domínio Espacial

- O **domínio espacial** refere-se ao próprio plano da imagem, ou seja, ao conjunto de pixels que compõe uma imagem.
- No **domínio espacial**, o nível de cinza de um ponto $f(x, y)$ após a **transformação** depende do valor do nível de cinza original do ponto e de outros **pontos da vizinhança** de $f(x, y)$.
- Em geral, **os pontos mais próximos contribuem mais** significativamente para o novo valor de nível de cinza do **que os pontos mais afastados**.

Filtragem no Domínio Espacial

- Os operadores de filtragem são geralmente classificados em filtros **lineares** e **não-lineares**.
- **Filtros lineares** calculam o valor resultante do pixel $f'(x, y)$ como uma **combinação linear** dos níveis de cinza em uma vizinhança local do pixel $f(x, y)$ na imagem original.

Filtragem no Domínio Espacial

- No **domínio espacial**, o processo de **filtragem** normalmente é realizado por meio de matrizes denominadas **máscaras**, as quais são aplicadas sobre a imagem.
- A cada posição da máscara está associado um **valor numérico**, chamado de **peso** ou **coeficiente**.
- A aplicação da máscara com centro na coordenada (x, y), sendo x a posição da coluna e y a posição de uma dada linha da imagem, consiste na substituição do valor do pixel na posição (x, y) por um novo valor, o qual depende dos valores dos pixels vizinhos e dos pesos da máscara.
- Os coeficientes do filtro são multiplicados pelos níveis de cinza dos pixels correspondentes e então somados, substituindo o nível de cinza do pixel central.

Filtragem no Domínio Espacial

- A figura abaixo mostra uma máscara genérica de 3×3 pixels. Denotando os níveis de cinza da imagem sob a máscara por $z_i = f(x, y)$, $1 \leq i \leq 9$, a resposta da máscara é

$$R = w_1 z_1 + w_2 z_2 + \ldots + w_9 z_9 = \sum_{i=1}^{9} w_i z_i \qquad (1)$$

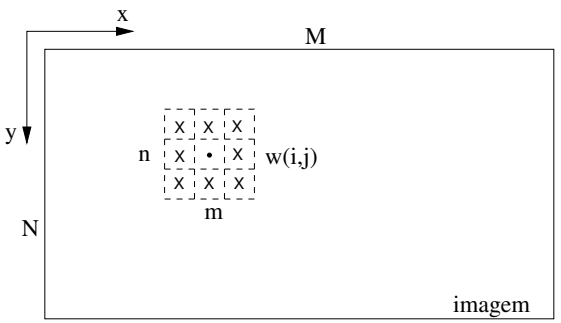
em que w_i representa os coeficientes da máscara.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Figura: Máscara de 3×3 pixels com coeficientes arbitrários.

Filtragem no Domínio Espacial

- Se o **centro da máscara** estiver em uma posição (x, y) na imagem, **o nível de cinza do pixel posicionado em** (x, y) **será substituído por** R, conforme equação 1.
- A **máscara é então movida para a próxima posição de pixel na imagem** e o **processo se repete** até que todas as posições de pixels tenham sido cobertas.
- A figura a seguir ilustra essa operação, em que a imagem e a máscara possuem dimensões $M \times N$ e $m \times n$ pixels, respectivamente.



Convolução

- A convolução de uma imagem f unidimensional por um filtro w pode ser expressa como:

$$\mathrm{w} * \mathrm{f}(x) = \sum_{i=\lfloor -m/2 \rfloor}^{\lfloor m/2 \rfloor} w(i)\, f(x-i) \tag{2}$$

- Para o caso da convolução bidimensional, os pesos do filtro devem ser refletidos tanto na horizontal quanto na vertical, ou seja

$$\mathrm{w}(x,y) * \mathrm{f}(x,y) = \sum_{i=\lfloor -m/2 \rfloor}^{\lfloor m/2 \rfloor} \sum_{j=\lfloor -n/2 \rfloor}^{\lfloor n/2 \rfloor} w(i,j)\, f(x-i, y-j) \tag{3}$$

em que m e n são dimensões da máscara.

Filtragem de Imagens

- O efeito visual de um filtro **passa-baixas** é o de **suavização** da imagem, uma vez que as **altas frequências**, que correspondem às **transições abruptas**, são atenuadas. A suavização tende também, pelas mesmas razões, a **minimizar** o efeito do **ruído** em imagens.
- Para filtros **passa-altas**, o efeito obtido é, em geral, o de tornar **mais nítidas** as transições entre regiões diferentes, conhecidas como **bordas**. Um efeito indesejado desses filtros é o de **enfatizar** o **ruído** presente na imagem.

Filtros Passa-Baixas (Lineares)

- O efeito de um **filtro passa-baixa** é o de **suavização** da imagem, uma vez que as **frequências altas** que correspondem às transições abruptas **são atenuadas**.
- A suavização tende também, pelas mesmas razões, a **minimizar o efeito do ruído** em imagens.
- Por outro lado, devido ao **borramento** causado pela filtragem passa-baixa, **detalhes finos** podem ser **removidos** da imagem.

Filtros Passa-Baixas (Lineares)

- Alguns exemplos de filtros passa-baixas são mostrados abaixo.

$$h_1 = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad h_2 = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad h_3 = \frac{1}{49}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

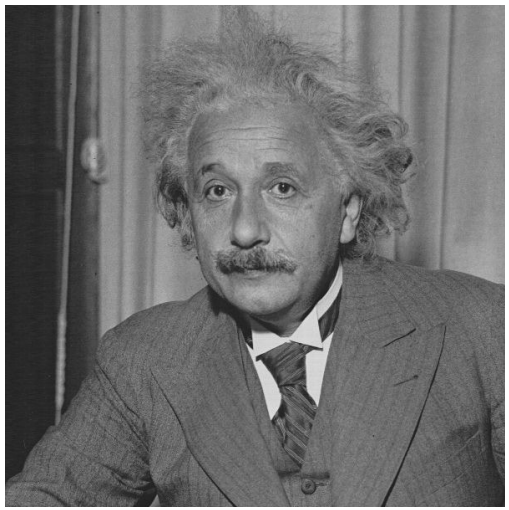
$$h_4 = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad h_5 = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Filtros Passa-Baixas (Lineares)

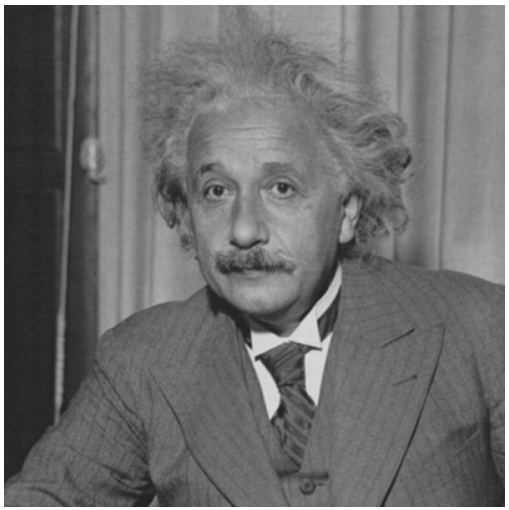
- As máscaras h_1, h_2 e h_3 possuem todos seus coeficientes iguais a 1, e o resultado da convolução é dividido por um fator de normalização.
- Tais filtros são denominados **filtros da média**, em que cada pixel é substituído pelo valor médio de seus vizinhos.
- O fator de normalização é, em geral, igual à soma dos coeficientes da máscara, de modo a preservar o valor médio.
- Dessa forma, a aplicação de filtros da média em uma região homogênea da imagem, ou seja, com níveis de cinza constantes, não sofrerá alteração de seus níveis de cinza.
- Os filtros h_4 e h_5 introduzem uma ponderação conforme a distância e a orientação dos pontos vizinhos.

Filtros Passa-Baixas (Lineares)

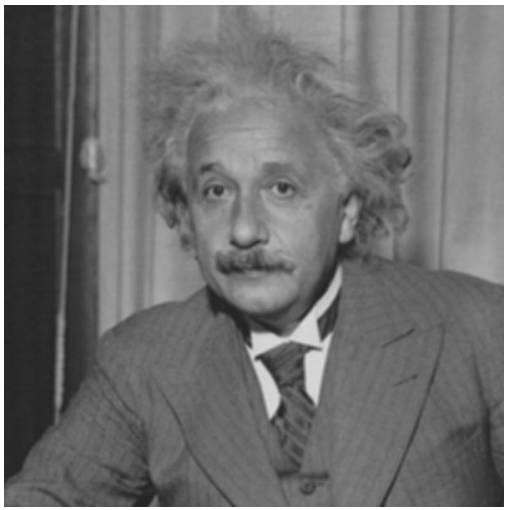
- Filtro da média com diferentes tamanhos de máscara.



(a) imagem original



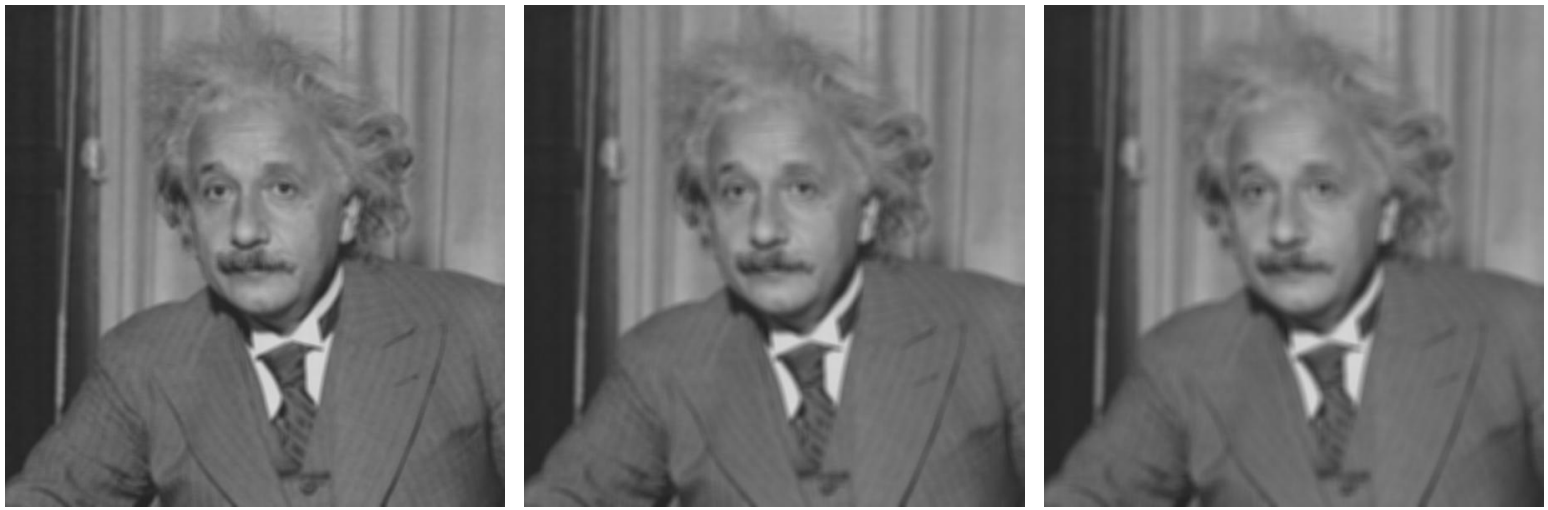
(b) máscara 3×3



(c) máscara 5×5

Filtros Passa-Baixas (Lineares)

- Filtro da média com diferentes tamanhos de máscara.



(a) máscara 7×7 (b) máscara 9×9 (c) máscara 11×11

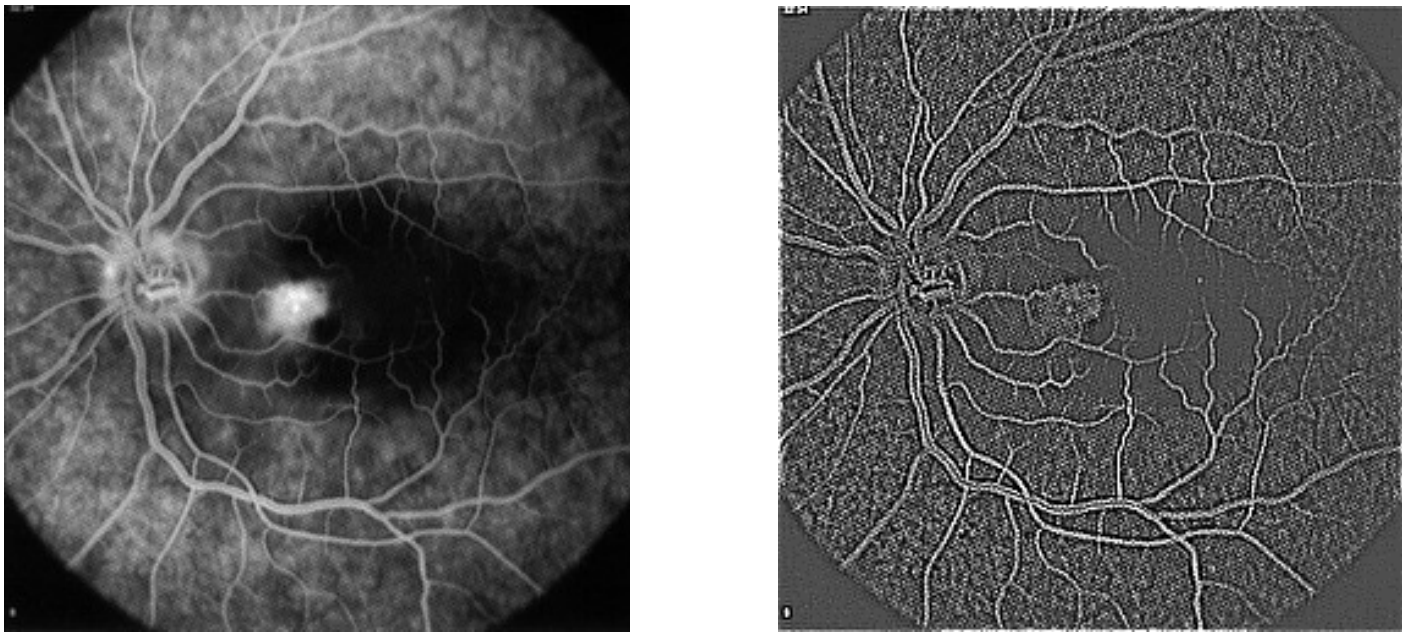
Filtros Passa-Altas

- Os filtros passa-altas podem ser usados para realçar certas características presentes na imagem, tais como bordas, linhas ou regiões de interesse.
- Dois exemplos de filtros passa-altas são mostrados a seguir:

$$h_1 = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array} \qquad h_2 = \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & 8 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$$

Filtros Passa-Altas

- A figura (b) mostra o resultado da aplicação do filtro passa-alta h_2 sobre a imagem da figura (a).



(a) (b)

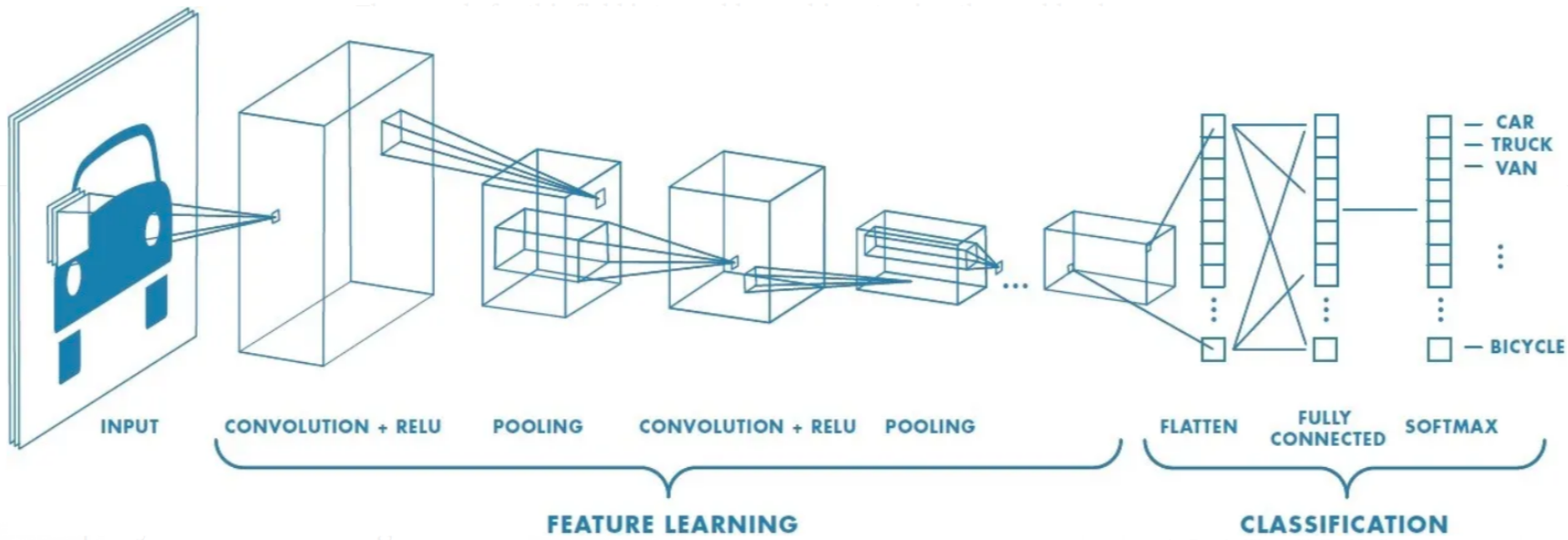
Figura: Filtro passa-alta. (a) imagem original; (b) resultado após aplicação de filtro passa-alta.

Filtros e Convolução

- **Questões Importantes:**
  - Tamanho de máscara impacta diretamente o filtro
  - Pesos da máscara definem o filtro
  - Como definir os pesos?

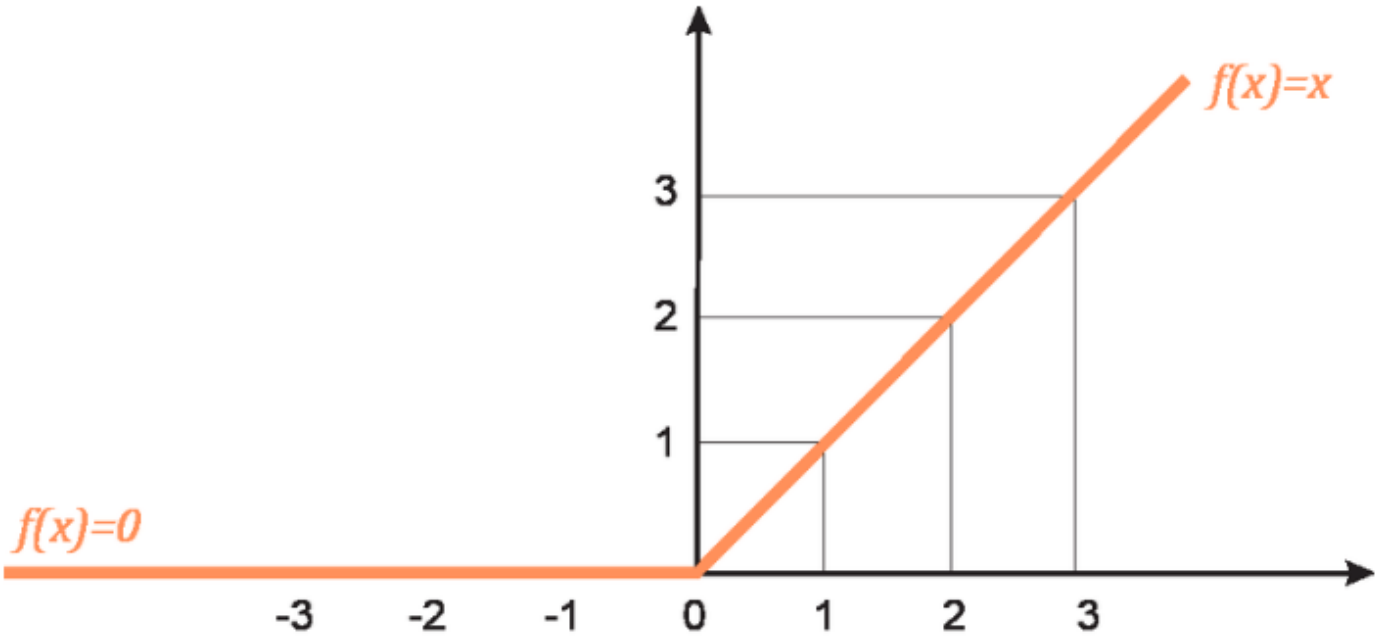
Convolutional Neural Networks

Convolutional Neural Networks



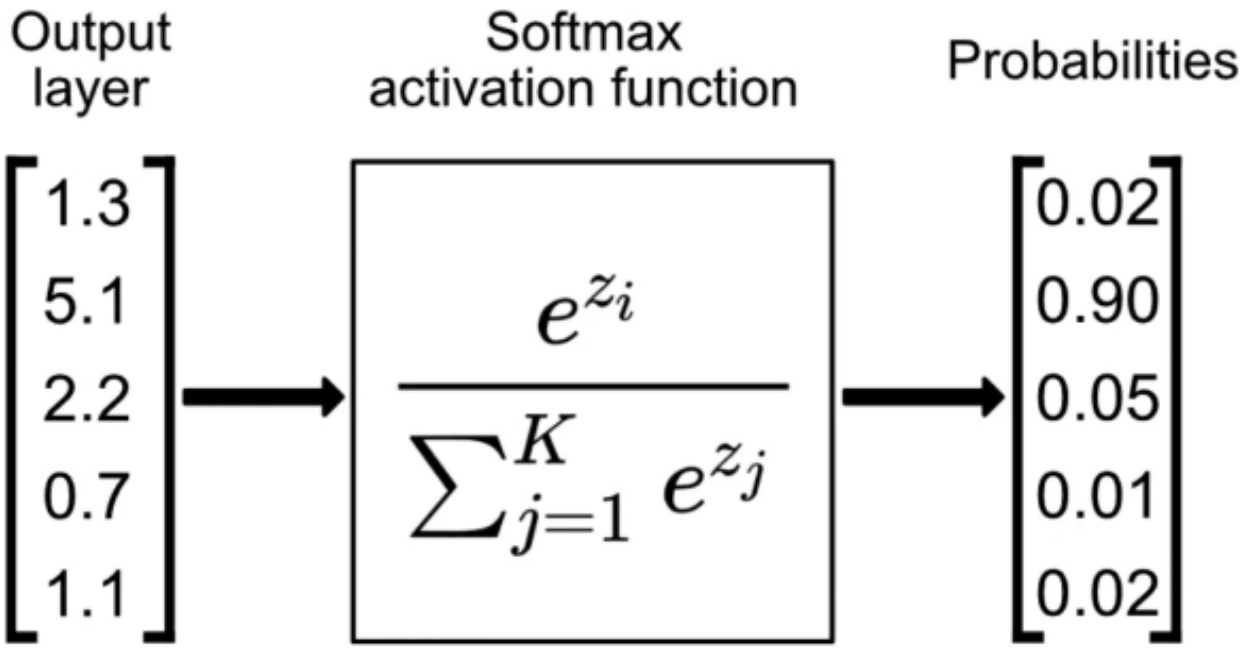
Função de Ativação: ReLU

Função de Ativação ReLU: $f(x) = max(0, x)$



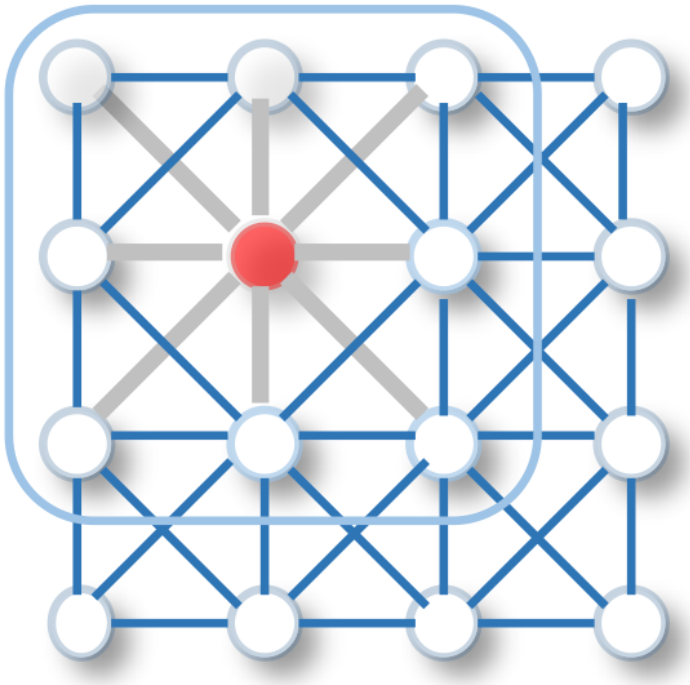
Softmax

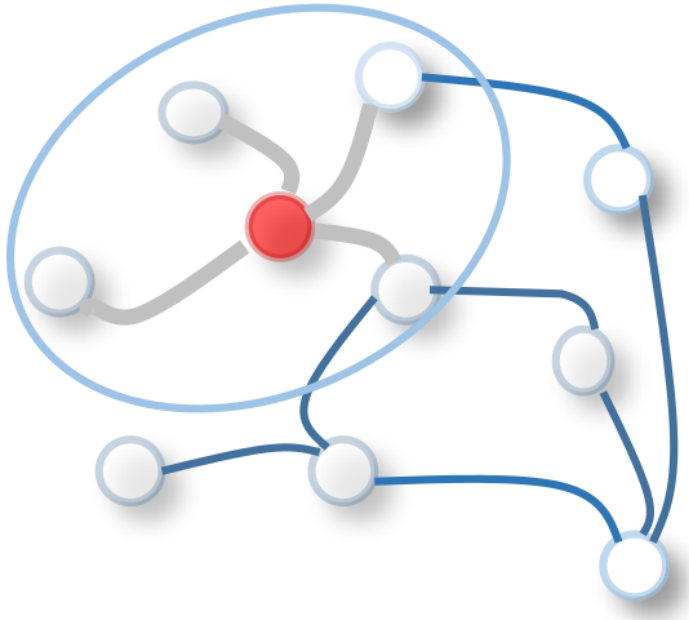
Softmax: transforma vetor de saída em probabilidades



2D Convolution vs Graph Convolutional

2D Convolution vs Graph Convolutional

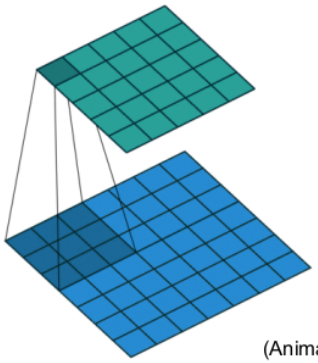




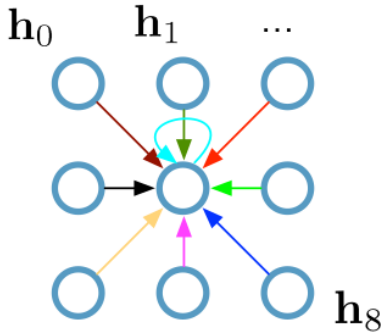
2D Convolution Update

2D Convolution Update

Single CNN layer with 3x3 filter:



(Animation by Vincent Dumoulin)



Update for a single pixel:

- Transform messages individually $\mathbf{W}_i\mathbf{h}_i$
- Add everything up $\sum_i \mathbf{W}_i\mathbf{h}_i$

$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

Full update:

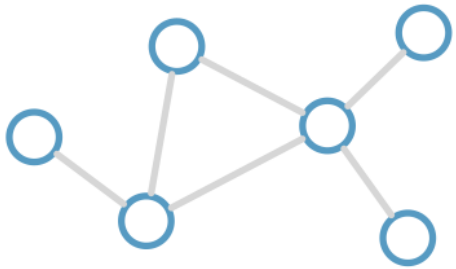
$$\mathbf{h}'_4 = \sigma(\mathbf{W}_0\mathbf{h}_0 + \mathbf{W}_1\mathbf{h}_1 + \cdots + \mathbf{W}_8\mathbf{h}_8)$$

Kipf, T. *Unsupervised Learning with Graph Neural Networks*

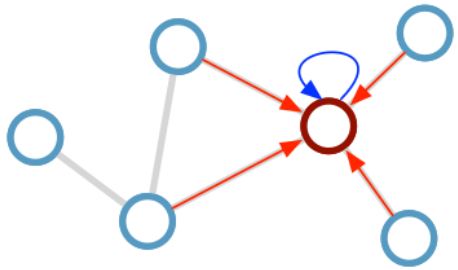
Graph Convolution Update

Graph Convolution Update

Consider this undirected graph:



Calculate update for node in red:



Update rule:

$$\mathbf{h}_i' = \sigma\left(\mathbf{W}_0\mathbf{h}_i + \sum_{j\in\mathcal{N}_i} \alpha_{ij}\mathbf{W}_1\mathbf{h}_j\right)$$

$\mathcal{N}_i$: neighbor indexes; α_{ij} : *norm.const*

Kipf, T. *Unsupervised Learning with Graph Neural Networks*

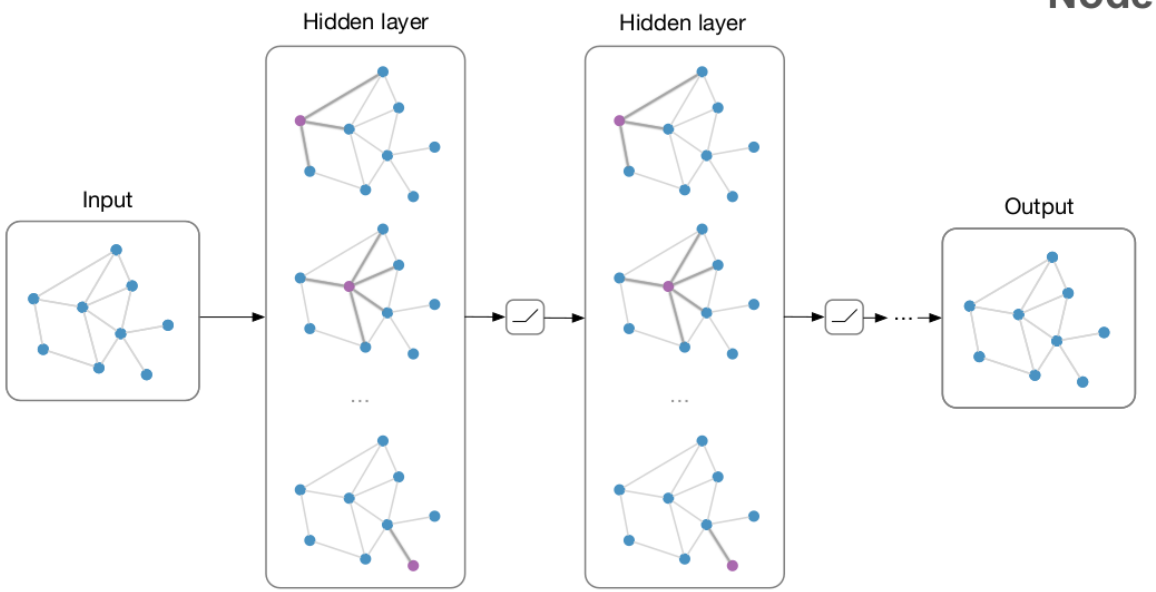
GCN - The Big Picture

Graph Convolutional Network

The bigger picture:

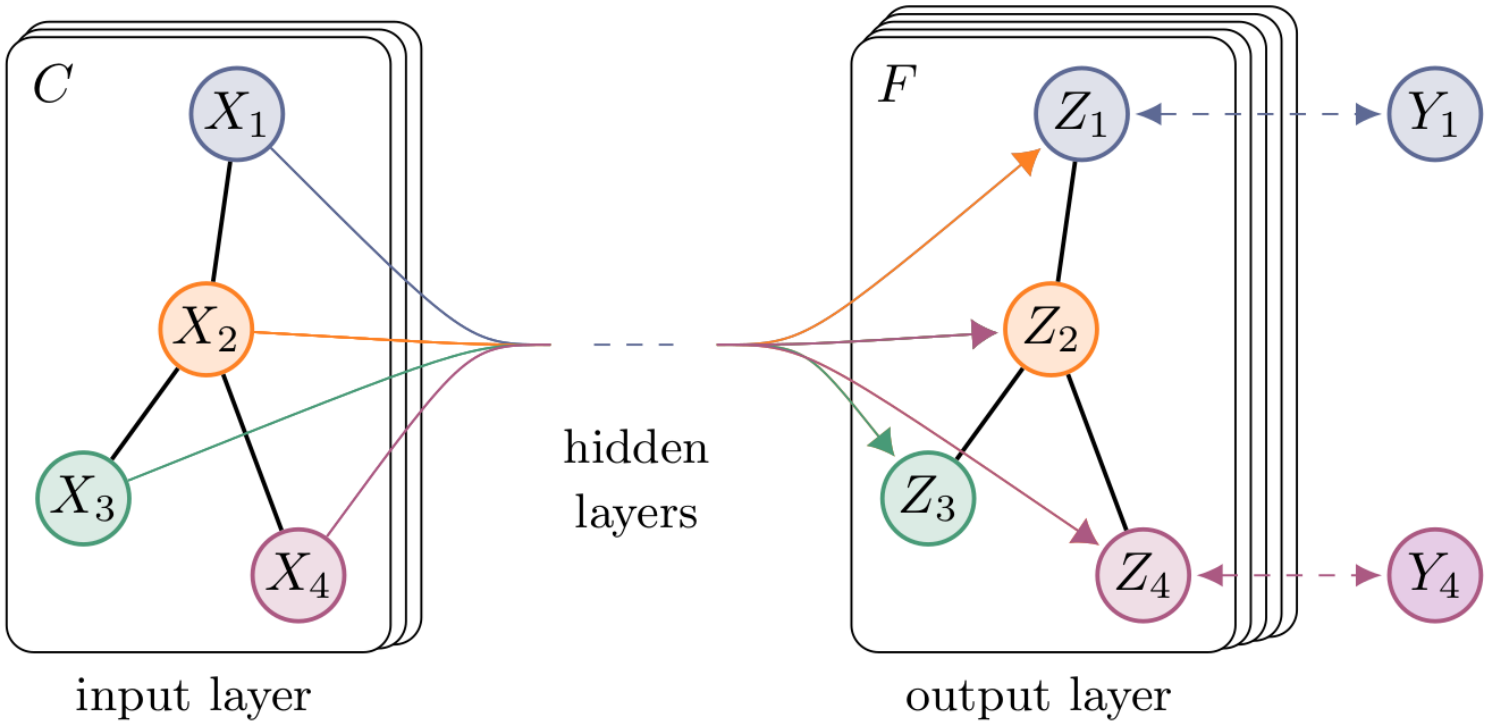
Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

Node features $\mathbf{h}_i$



GCN - Semi-Supervised Classification

Graph Convolutional Network



Semi-Supervised Classification with Graph Convoltuional Networks, ICLR 2017 [2].

GCN - Semi-Supervised Classification:

GCN Notation:

- N: number of samples (nodes)
- C: number of input channels
  - C-dimensional feature vector for every node
- F: feature maps in the output layer
- X: feature matrix (signal), with $\mathrm{X} \in \mathbb{R}^{N \times C}$
- A: graph adjacency matrix, with $\mathrm{A} \in \mathbb{R}^{N \times N}$
- Z: embedding matrix , with $\mathrm{X} \in \mathbb{R}^{N \times F}$
- Y_L: class labels
- GCN as a function $f(X, A)$

GCN - Semi-Supervised Classification:

GCN Model Definition

$$Z = f(X, A) = softmax(\hat{A}ReLU(\hat{A}XW^{(0)})W^{(1)})$$

- $\hat{A}$ is a normalized adjacency matrix, where D is a degree matrix and $\hat{A} = D^{-1/2}AD^{-1/2}$
- $W^{(0)} \in \mathbb{R}^{C \times H}$ is an input-to-hidden weight matrix for a hidden layer with H feature maps.
- $W^{(1)} \in \mathbb{R}^{H \times F}$ is a hidden-to-output P weight matrix.

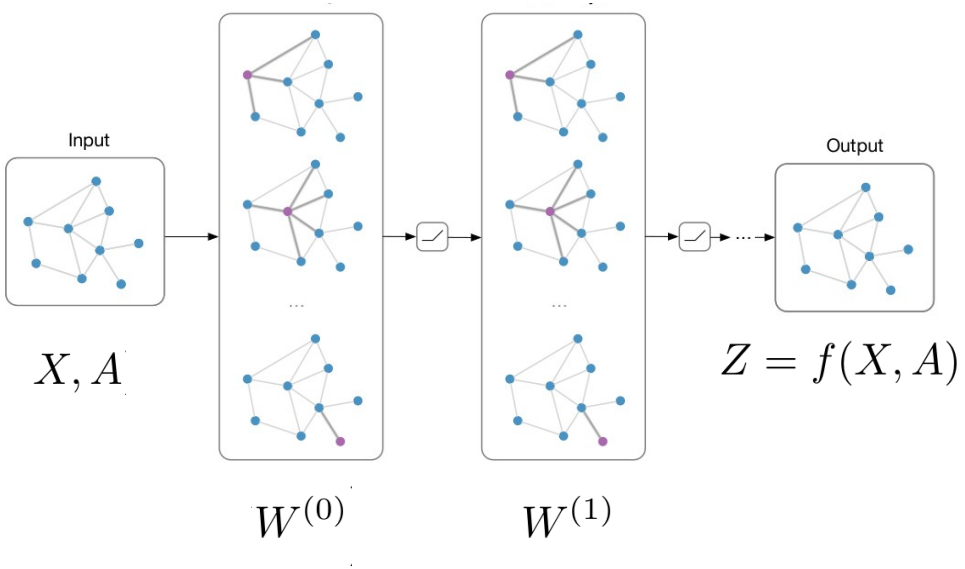
Graph Convolutional Network - Training

Cross Entropy Loss - GCN Training

- "Cross-entropy builds upon the idea of entropy from information theory and calculates the number of bits required to *represent an average event from one distribution compared to another distribution*."
- Em gera, o conceito fundamental consistem em:
  - **minimizar a diferença entre a distribuição de classes predita e distribuição real**.
- Outras funções de perda podem ser utilizadas

Graph Convolutional Network - Definition

Graph Convolutional Network - Definition



$$Z = f(X, A) = \text{softmax}\Big(\hat{A}\ \text{ReLU}\Big(\hat{A}XW^{(0)}\Big)\,W^{(1)}\Big)$$

Semi-Supervised Classification with Graph Convoltuional Networks, ICLR 2017 [2].

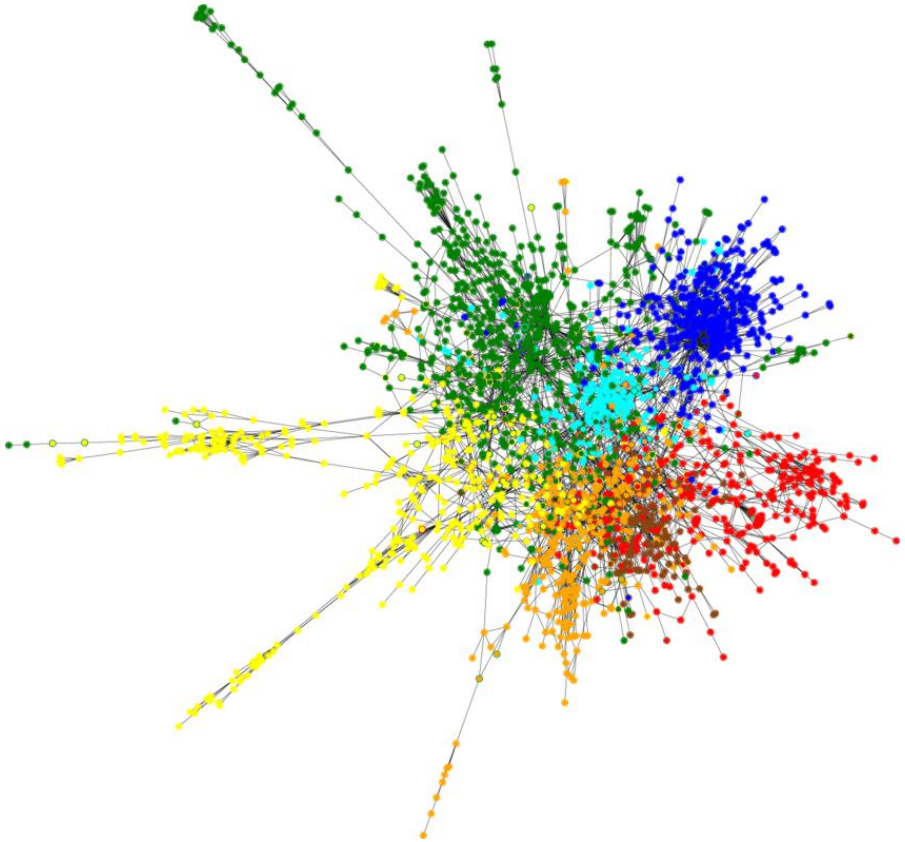
Graph Convolutional Network

Cora Dataset - Semi-Supervised Classification

- Publicações Científicas
- Features são vetores binários em indicando a presença ou não de palavras
- Arestas no grafo indicam referências entre as publicações
- Semi-supervised: only 20 labels per class

Graph Convolutional Network

Cora Dataset - Semi-Supervised Classification



Graph Convolutional Network - Results

Graph Convolutional Network - Definition

Method	Citeseer	Cora	Pubmed	NELL
ManiReg [3]	60.1	59.5	70.7	21.8
SemiEmb [28]	59.6	59.0	71.1	26.7
LP [32]	45.3	68.0	63.0	26.5
DeepWalk [22]	43.2	67.2	65.3	58.1
ICA [18]	69.1	75.1	73.9	23.1
Planetoid* [29]	64.7 (26s)	75.7 (13s)	77.2 (25s)	61.9 (185s)
GCN (this paper)	**70.3** (7s)	**81.5** (4s)	**79.0** (38s)	**66.0** (48s)

Semi-Supervised Classification with Graph Convoltuional Networks, ICLR 2017 [2].

Graph Convolutional Networks

Graph Convolutional Networks - Variações

- Modelo de GCN proposto teve grande impacto:

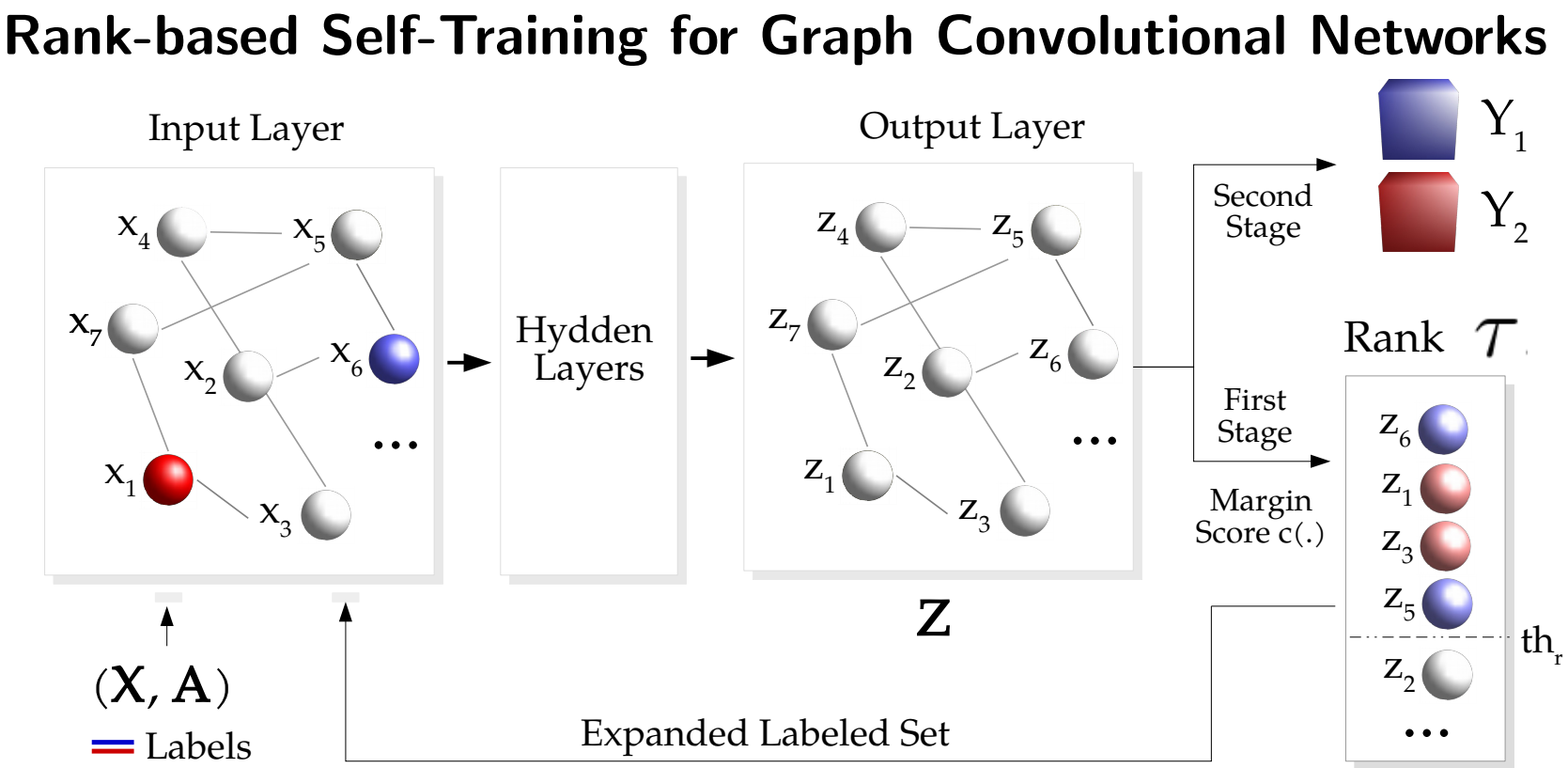
- Variações Propostas:
  - Graph Attention Network (GAT) [4]
  - Simplifying Graph Convolutional Networks (SGC) [5]
  - GCN with ARMA Filters [1]
  - APPNP [3]
  - (...)

Aspectos de Pesquisa

Aspectos de Pesquisa:

- Como aumentar acurácia de modelos de GCNs em tarefas de **classificação**?
  - Utilizando informações do próprio modelo (**self-training**)
  - Pré-processando as entradas da GCN: grafo e features
- Como explorar modelos de GCN em ausência de rótulos?
  - **Self-Supervised Clustering**
- Como utilizar modelos de GCN para tarefas de recuperação (**re-ranking**)?

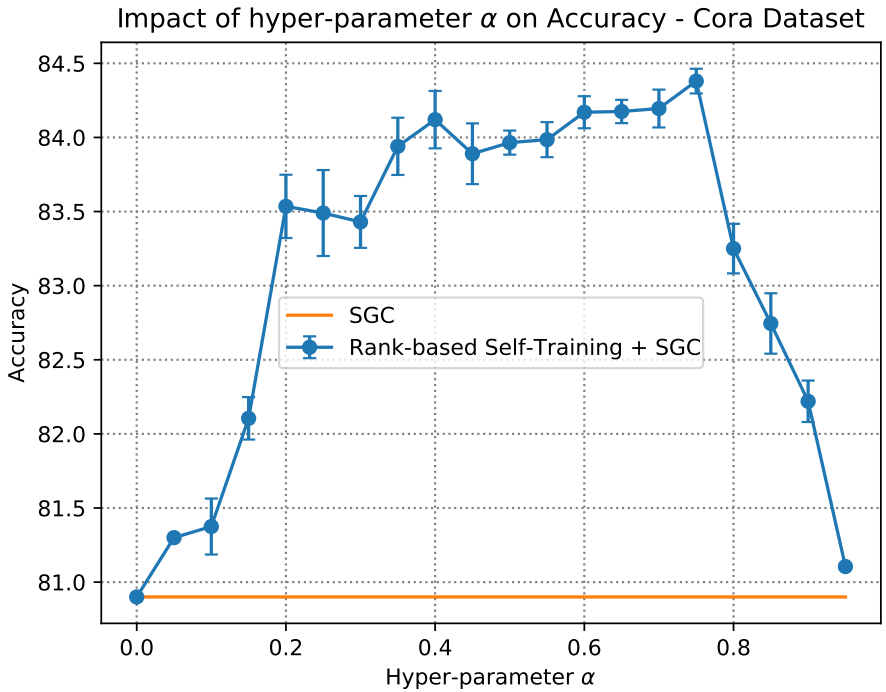
Graph Convolutional Network - Self-Training



Information Processing and Management 2021.

Graph Convolutional Network - Self-Training

Rank-based Self-Training for Graph Convolutional Networks



Information Processing and Management 2021.

Graph Convolutional Network - Self-Training

Rank-based Self-Training for Graph Convolutional Networks

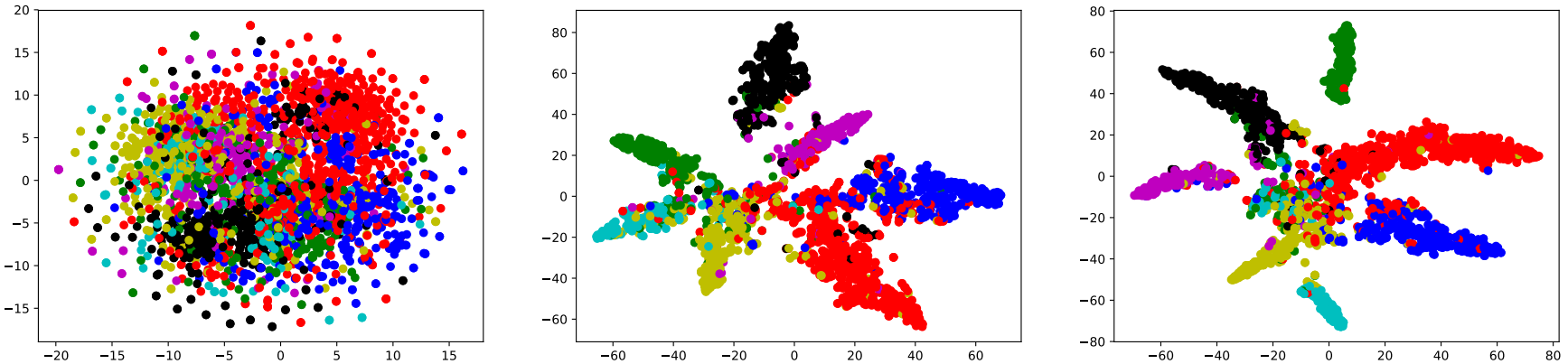
Accuracy of Rank-based Self-Training on citation datasets. Results without feature normalization.

Method	Cora	CiteSeer	PubMed
GCN (Kipf & Welling, 2017)	80.5 ± 0.6	66.9 ± 0.7	78.9 ± 0.3
Rank-based Self-Trainig + GCN	83.3 ± 0.9	69.4 ± 1.9	80.3 ± 0.4
SGC (Wu, Souza, Zhang, Fifty, Yu, & Weinberger, 2019)	80.9 ± 0.0	69.3 ± 0.0	78.9 ± 0.0
Rank-based Self-Trainig + SGC	84.1 ± 0.2	**73.1** ± 0.2	75.9 ± 0.0
ARMA (Bianchi et al., 2019)	80.1 ± 1.0	64.6 ± 2.2	78.2 ± 0.4
Rank-based Self-Trainig + ARMA	82.9 ± 1.0	68.0 ± 4.4	79.5 ± 0.6
APPNP (Klicpera, Bojchevski, & Günnemann, 2019)	82.8 ± 0.7	70.1 ± 0.7	79.9 ± 0.2
Rank-based Self-Trainig + APPNP	**84.7** ± 0.6	71.2 ± 0.7	**81.2** ± 0.7

Information Processing and Management 2021.

Graph Convolutional Network - Self-Training

Rank-based Self-Training for Graph Convolutional Networks



Cora Raw features / Cora SGC embeddings / Cora Self-Training SGC embeddings

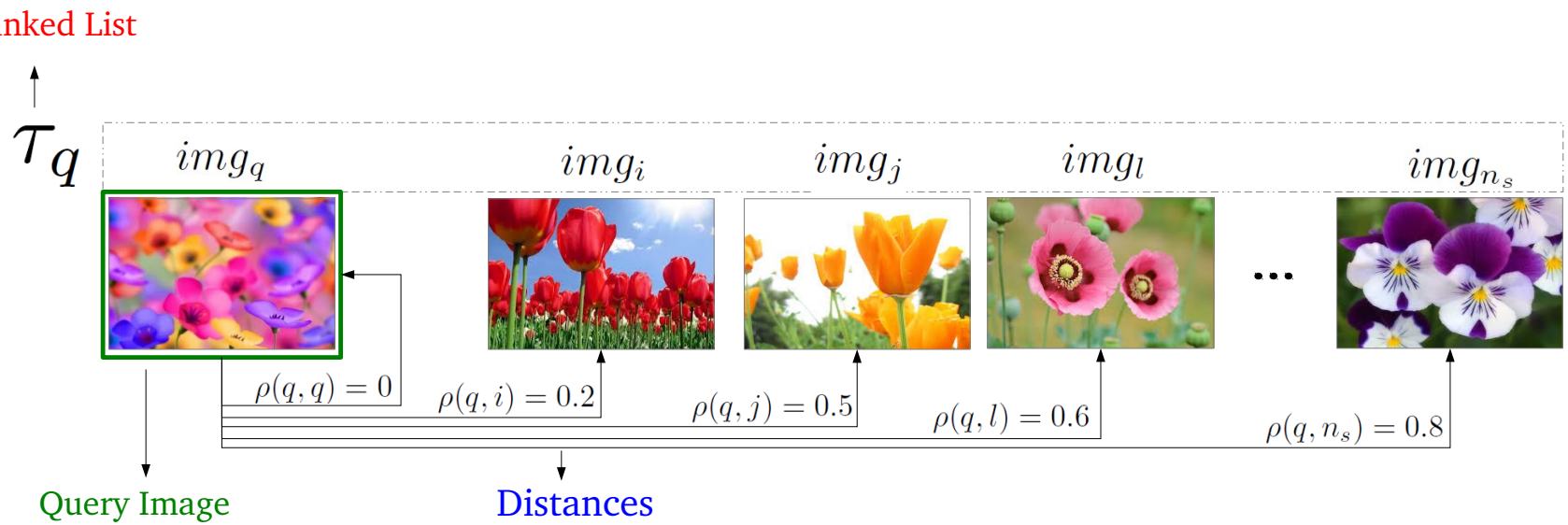
GCNs e Grafo de Entrada

GCNs e Grafo de Entrada

- *Qualidade* do grafo tem impacto direto nos resultado de classificação
  - Quantidade de arestas entre nós da mesma classe
- Há tarefas em que os grafos são pré-definidos
- Situações em que não há grafos, eles podem ser computados: **grafos kNN**
- Como aumentar a eficácia dos grafos?
  - **Manifold Learning by Contextual Similarity**

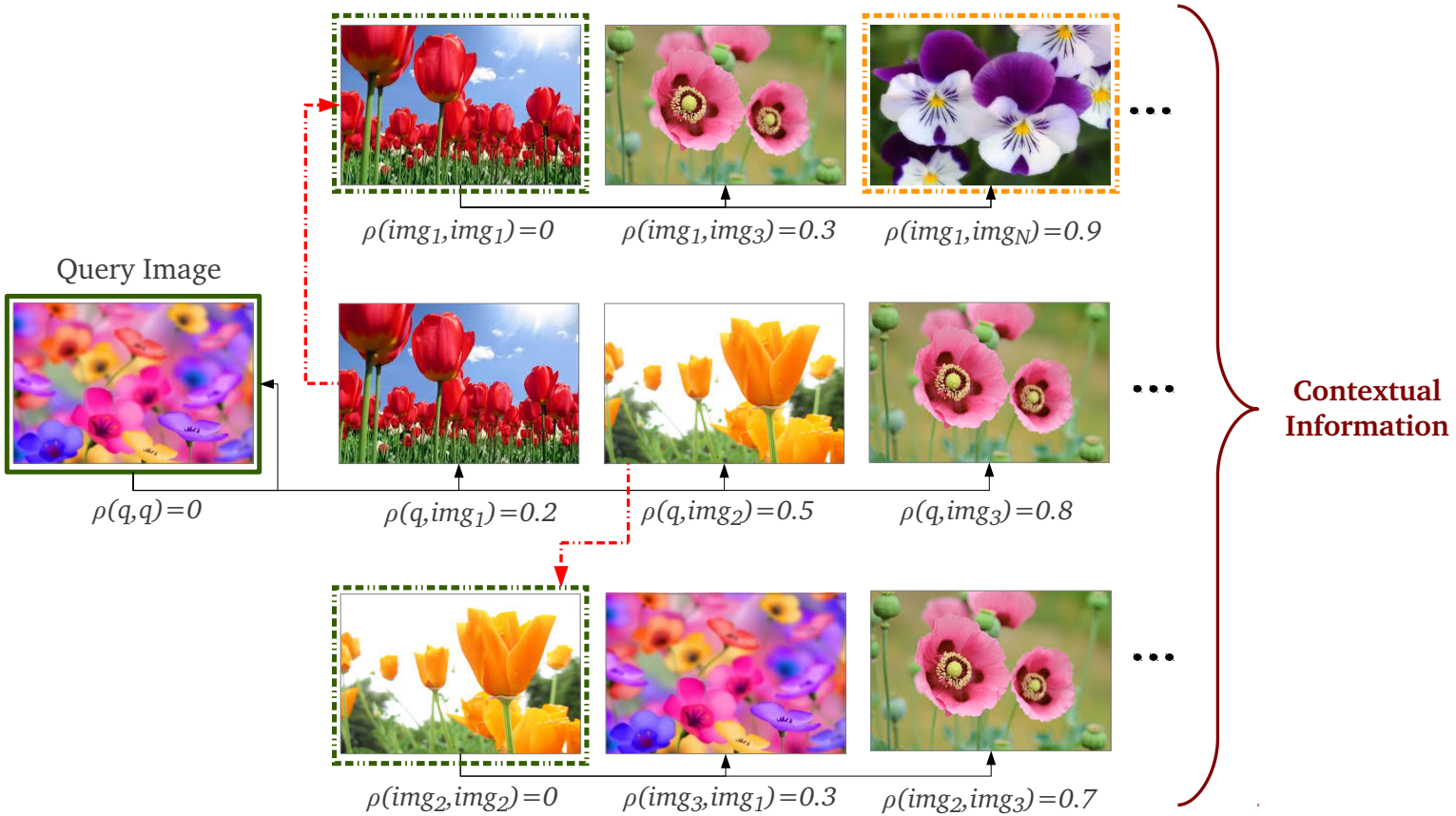
Pairwise vs Contextual Similarity

Pairwise Similarity:



Pairwise vs Contextual Similarity

Contextual Similarity:



Manifold Learning by Contextual Similarity

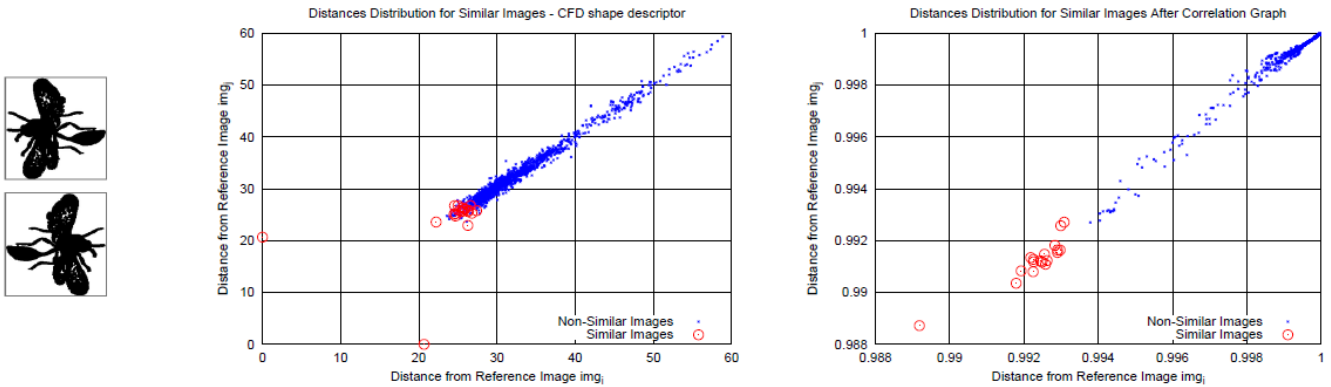


Fig. 9. Impact of the algorithm on distances distribution for similar reference images: (a) Similar Reference Images (fly-2.gif and fly-3.gif) from the MPEG-7 [25] dataset; (b) Original distances distribution; (c) Distances distribution after the algorithm.

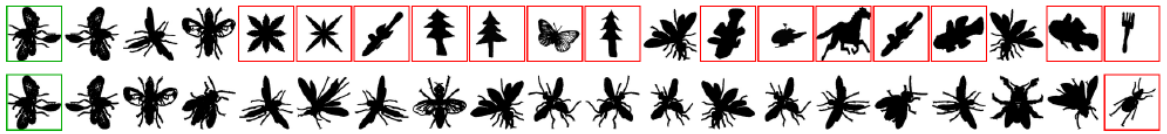


Fig. 10. Visual example of the effectiveness gain. Retrieval results before (first row) and after the use of the algorithm (second row). Query image (fly-2.gif) from the MPEG-7 [25] dataset with green border and wrong images with red borders.

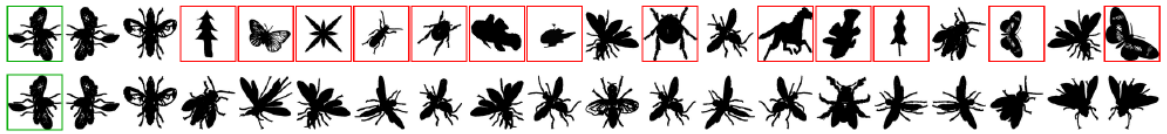


Fig. 11. Visual example analogous to Figure 10, considering other similar query image (fly-3.gif) from the MPEG-7 [25] dataset.

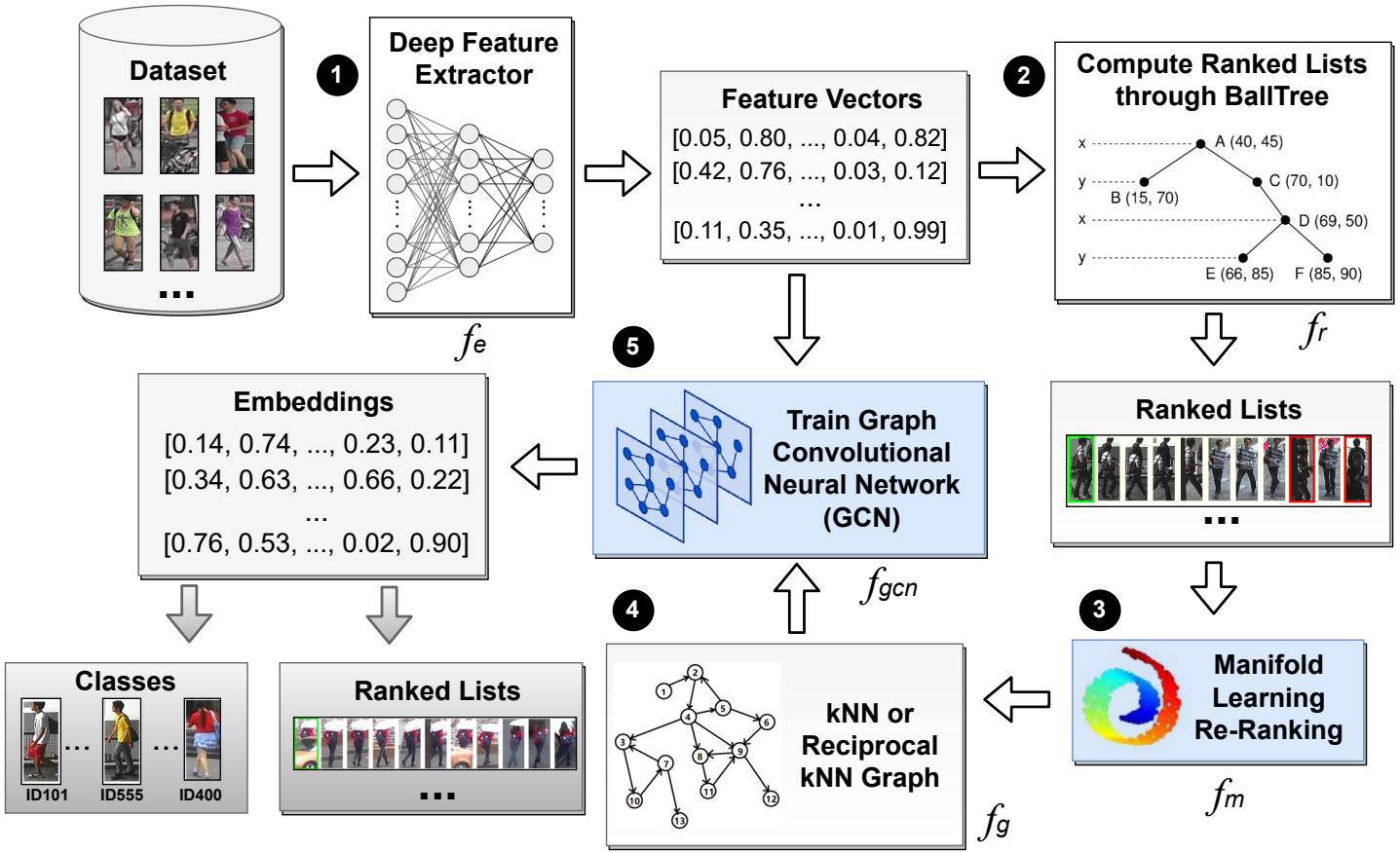
Neurocomputing 2016.

Manifold Learning by Contextual Similarity

Manifold Learning by Contextual Similarity

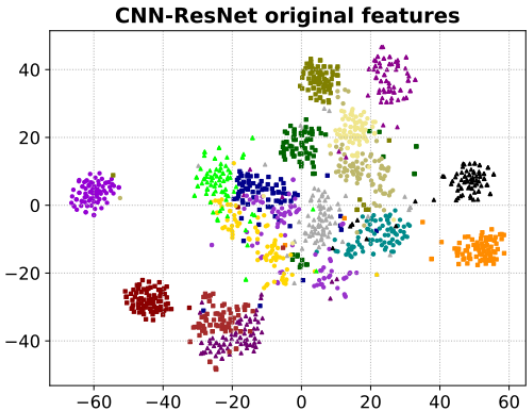
- Estratégias Totalmente Não Supervisionadas
- Aumento de Eficácia em Tarefas de Retrieval
- Pré-Processamento para outras Tarefas de Machine Learning
- Diversas abordagens para codificar e explorar a **informação contextual**
  - Grafos e Hipergrafos
  - Correlação de Ranqueamento
  - Clustering
- Unsupervised Distance Learning Framework (UDLF)
  - `https://github.com/UDLF/`
- **Combinação de Estratégias com GCNs**

Manifold GCN

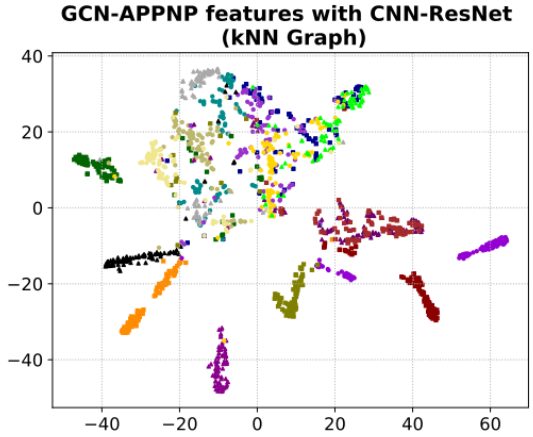


Computer Vision and Image Understanding (CVIU) 2023.

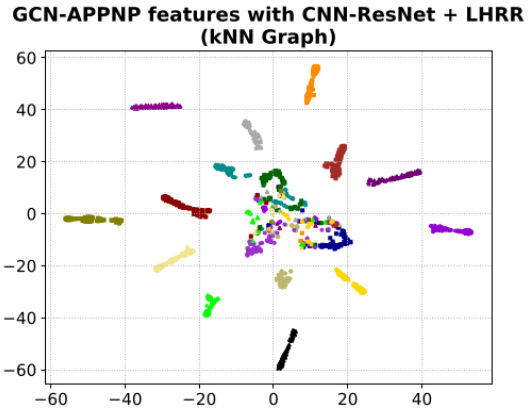
Manifold GCN



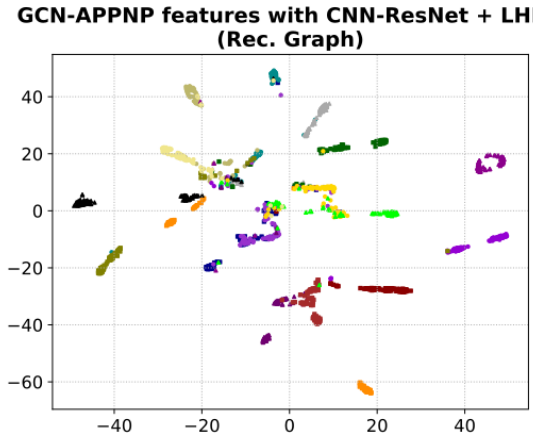
(a) Original Features



(b) GCN-APPNP Features (kNN Graph)



(c) GCN-APPNP Features with LHRR (kNN Graph)



(d) GCN-APPNP Features with LHRR (Rec. Graph)

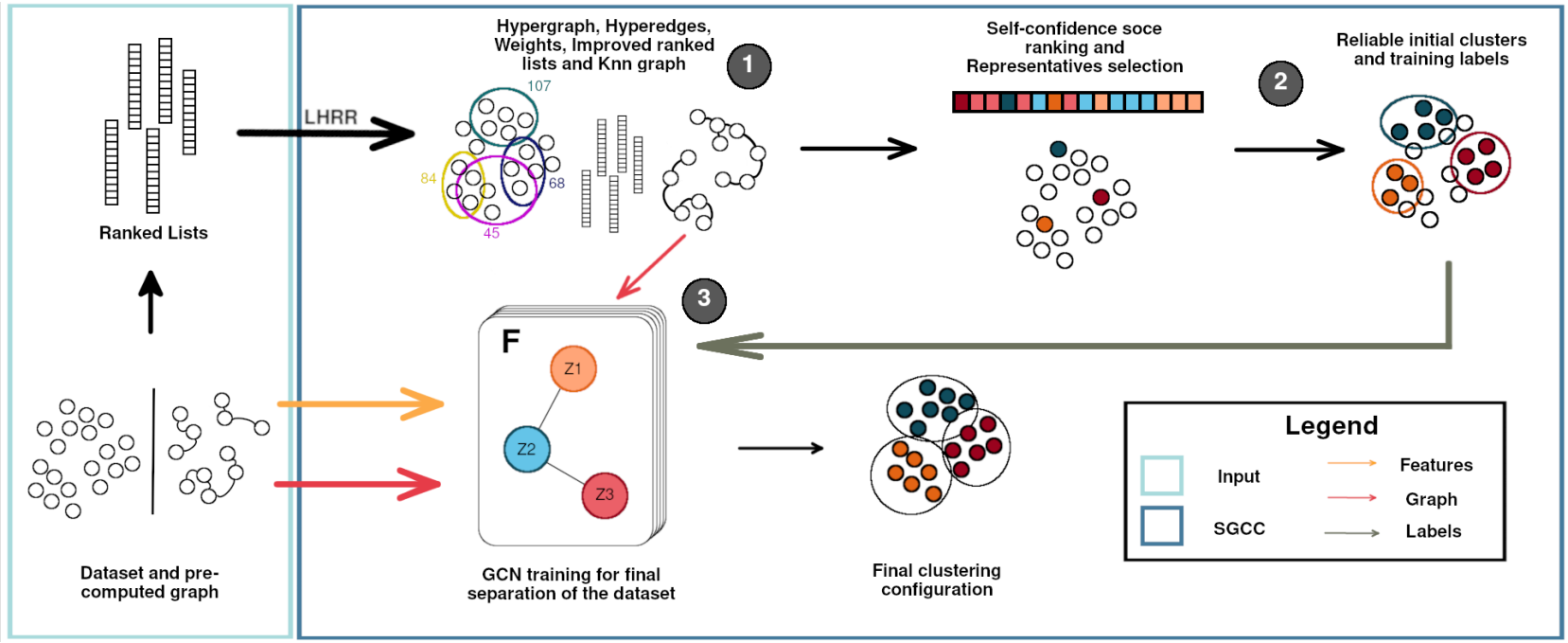
Computer Vision and Image Understanding (CVIU) 2023.

Manifold GCN

Classifier specification			Feature				
GCN	Graph	Re-rank	CNN-ResNet (He et al., 2016)	CNN-DPNet (Chen et al., 2017)	CNN-SENet (Hu et al., 2018)	T2T-VIT24 (Yuan et al., 2021)	VIT-B16 (Dosovitskiy et al., 2021)
GCN-Net	kNN	–	89.34 ± 0.0950	86.49 ± 0.0998	89.17 ± 0.0956	89.02 ± 0.1452	89.93 ± 0.2878
	kNN	LHRR	91.40 ± 0.0906	88.94 ± 0.1958	90.19 ± 0.1392	90.68 ± 0.0957	94.57 ± 0.121
	kNN	RDPAC	91.46 ± 0.1402	89.05 ± 0.1054	90.65 ± 0.0483	91.77 ± 0.1246	94.29 ± 0.139
	kNN	BFSTREE	**92.03 ± 0.1165**	89.28 ± 0.1858	91.19 ± 0.1102	91.78 ± 0.0432	94.30 ± 0.3362
	Rec.	–	91.68 ± 0.1064	**89.62 ± 0.1114**	**91.81 ± 0.1159**	92.19 ± 0.0908	93.42 ± 0.1987
	Rec.	LHRR	91.68 ± 0.0224	88.48 ± 0.1268	90.58 ± 0.0901	91.50 ± 0.0684	94.63 ± 0.139
	Rec.	RDPAC	92.00 ± 0.1434	89.55 ± 0.0944	90.93 ± 0.1654	91.96 ± 0.0705	**94.76 ± 0.1577**
	Rec.	BFSTREE	92.00 ± 0.0954	89.33 ± 0.1221	91.32 ± 0.0833	**92.43 ± 0.0401**	94.39 ± 0.2771
GCN-SGC	kNN	–	89.62 ± 0.0321	86.78 ± 0.0256	89.81 ± 0.0426	88.95 ± 0.0482	93.36 ± 0.0401
	kNN	LHRR	91.19 ± 0.0262	88.74 ± 0.0242	89.90 ± 0.044	90.49 ± 0.0518	95.20 ± 0.0219
	kNN	RDPAC	91.47 ± 0.0216	88.95 ± 0.0632	90.70 ± 0.0403	91.77 ± 0.0521	94.76 ± 0.078
	kNN	BFSTREE	91.98 ± 0.0246	89.23 ± 0.0453	91.40 ± 0.0061	91.71 ± 0.0444	95.26 ± 0.0759
	Rec.	–	91.98 ± 0.0133	89.83 ± 0.0415	**92.15 ± 0.0164**	**92.75 ± 0.0908**	95.49 ± 0.0107
	Rec.	LHRR	91.73 ± 0.0508	88.70 ± 0.0669	90.73 ± 0.0235	91.68 ± 0.0305	**95.57 ± 0.017**
	Rec.	RDPAC	92.00 ± 0.0247	**89.84 ± 0.1057**	90.85 ± 0.0396	92.31 ± 0.072	95.50 ± 0.020
	Rec.	BFSTREE	**92.04 ± 0.009**	89.49 ± 0.0627	91.30 ± 0.0257	92.54 ± 0.0591	95.30 ± 0.0479
GCN-GAT	kNN	–	90.48 ± 0.1727	83.28 ± 0.33	91.13 ± 0.1107	90.7 ± 0.1187	91.3 ± 0.1764
	kNN	LHRR	92.21 ± 0.1328	88.59 ± 0.4012	91.28 ± 0.2208	92.2 ± 0.0839	94.56 ± 0.1777
	kNN	RDPAC	91.86 ± 0.1403	89.78 ± 0.2723	91.41 ± 0.1429	92.82 ± 0.0956	94.46 ± 0.2555
	kNN	BFSTREE	**92.42 ± 0.1008**	89.61 ± 0.362	91.95 ± 0.1382	93.09 ± 0.1337	94.58 ± 0.2226
	Rec.	–	92.02 ± 0.0917	89.0 ± 0.2638	**92.23 ± 0.0844**	92.81 ± 0.113	93.64 ± 0.2373
	Rec.	LHRR	92.19 ± 0.1057	89.17 ± 0.2074	91.18 ± 0.1451	92.41 ± 0.1456	94.55 ± 0.1918
	Rec.	RDPAC	92.22 ± 0.0858	**90.48 ± 0.1718**	91.48 ± 0.1021	93.02 ± 0.1334	**94.89 ± 0.1492**
	Rec.	BFSTREE	92.30 ± 0.1128	90.01 ± 0.2374	91.88 ± 0.1081	**93.35 ± 0.1537**	94.75 ± 0.1385

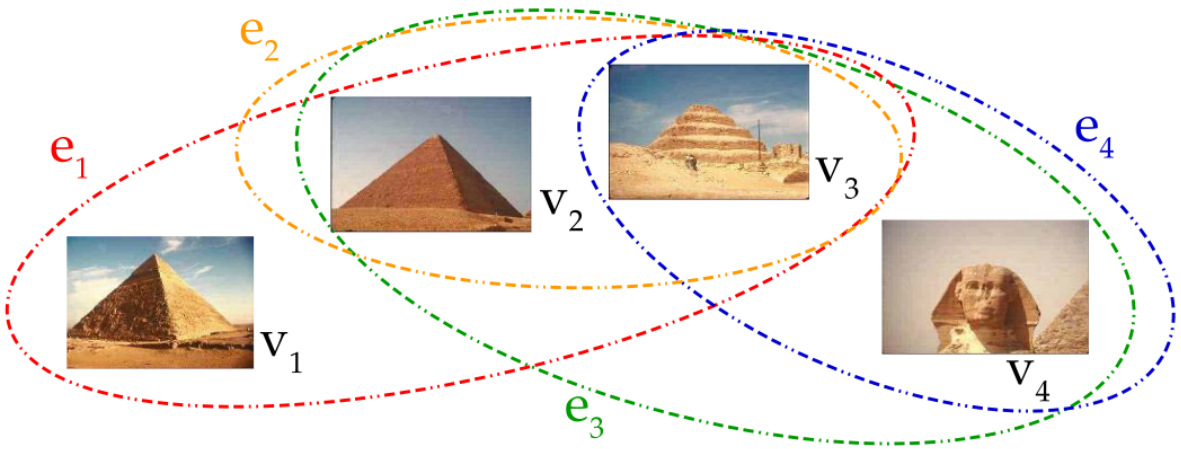
Computer Vision and Image Understanding (CVIU) 2023.

Self-Supervised Graph Convolutional Clustering (SGCC)



WACV 2023.

SGCC: Rank-based Hypergraph Model

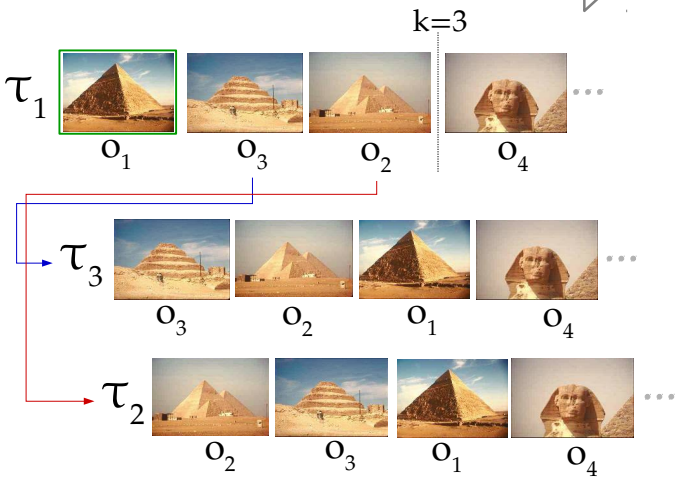


$$\mathbf{H} = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{pmatrix} 1 & \mathbf{0.40} & 0.63 & 0 \\ 0 & 1 & 0.43 & 0 \\ 0 & 0.52 & 1 & 0.18 \\ 0 & 0 & 0.22 & 1 \end{pmatrix} \end{array}$$

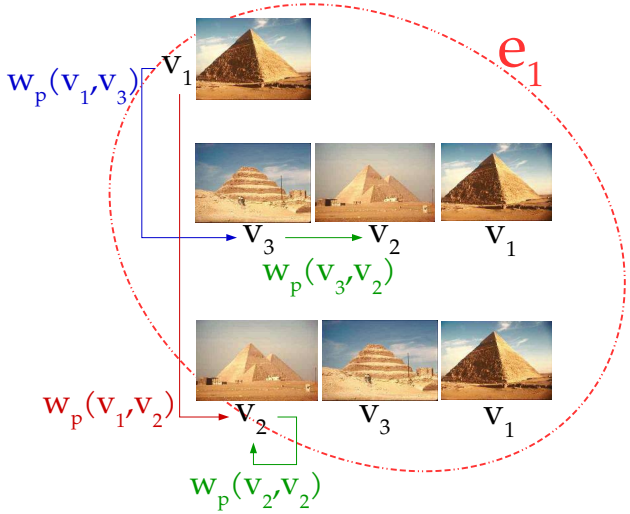
$$\mathbf{H}^T = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} e_1 & e_2 & e_3 & e_4 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ \mathbf{0.40} & 1 & 0.52 & 0 \\ 0.63 & 0.43 & 1 & 0.22 \\ 0 & 0 & 0.18 & 1 \end{pmatrix} \end{array}$$

SGCC: Rank-based Hypergraph Model

Ranking Results:



Hyperedge Definition:



Hyperedge *vs* Vertice

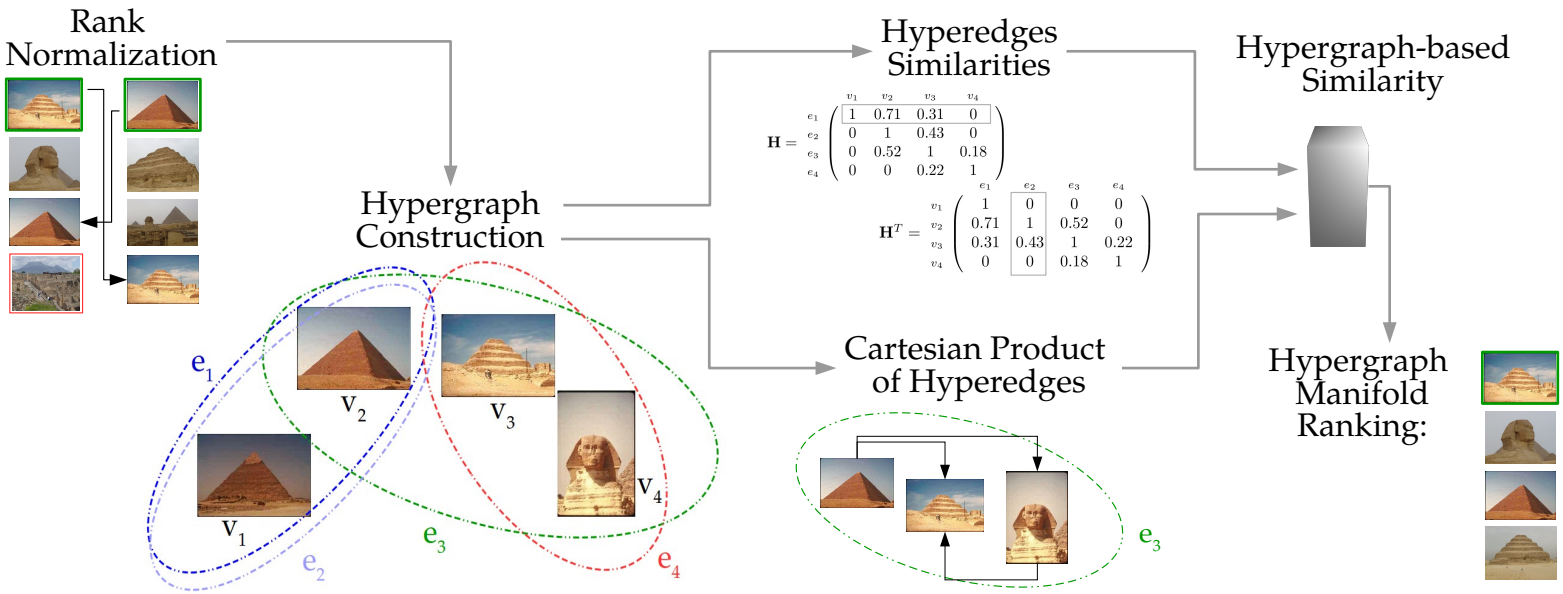
$$\mathbf{r(e_1,v_2)} = w_p(o_1,o_3) \times w_p(o_3,o_2) + w_p(o_1,o_2) \times w_p(o_2,o_2)$$

$$\mathbf{r(e_1,v_2)} = (1\text{-}\log_3 2) \times (1\text{-}\log_3 2) + (1\text{-}\log_3 3) \times (1\text{-}\log_3 1)$$

$$\mathbf{r(e_1,v_2)} = (0.63) \times (0.63) + (0) \times (1)$$

$$\mathbf{r(e_1,v_2) = 0.40}$$

SGCC: Rank-based Hypergraph Model



IEEE Transactions on Image Processing 2019.

Self-Supervised Graph Convolutional Clustering (SGCC)

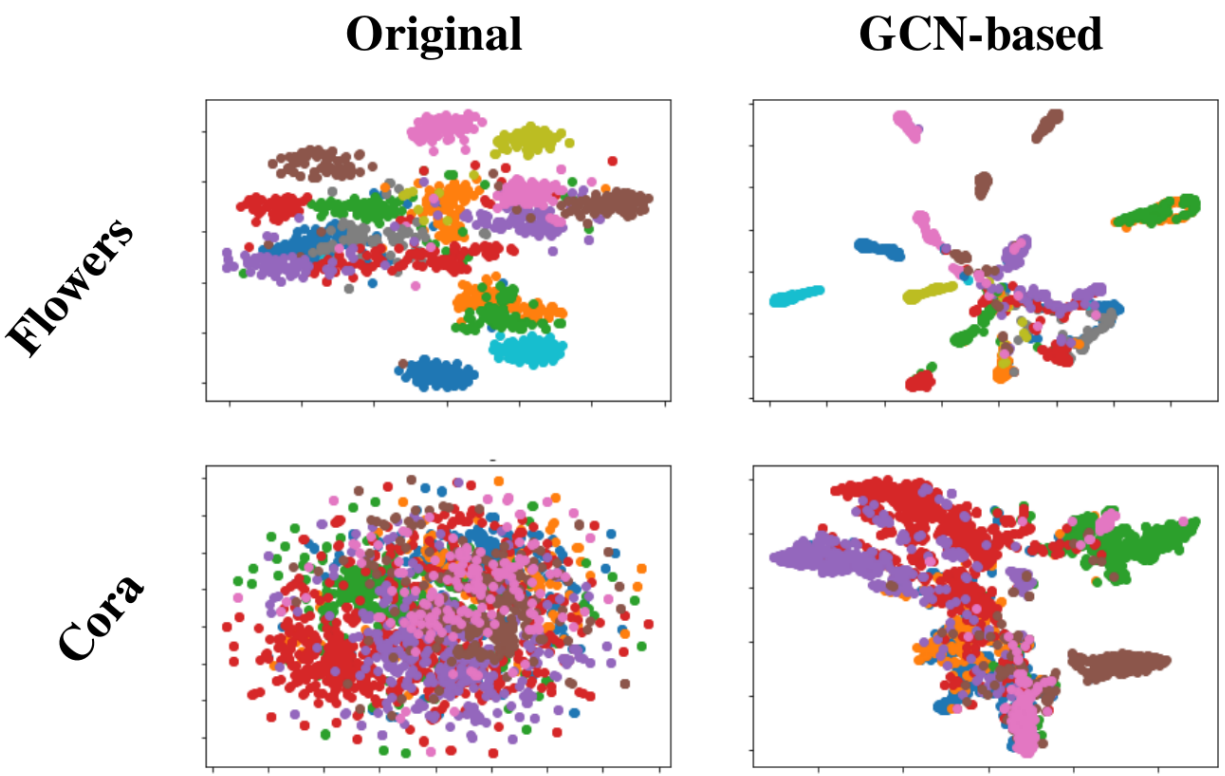


Figure 3. Visual analysis: the impact of the GCN embeddings.

WACV 2023.

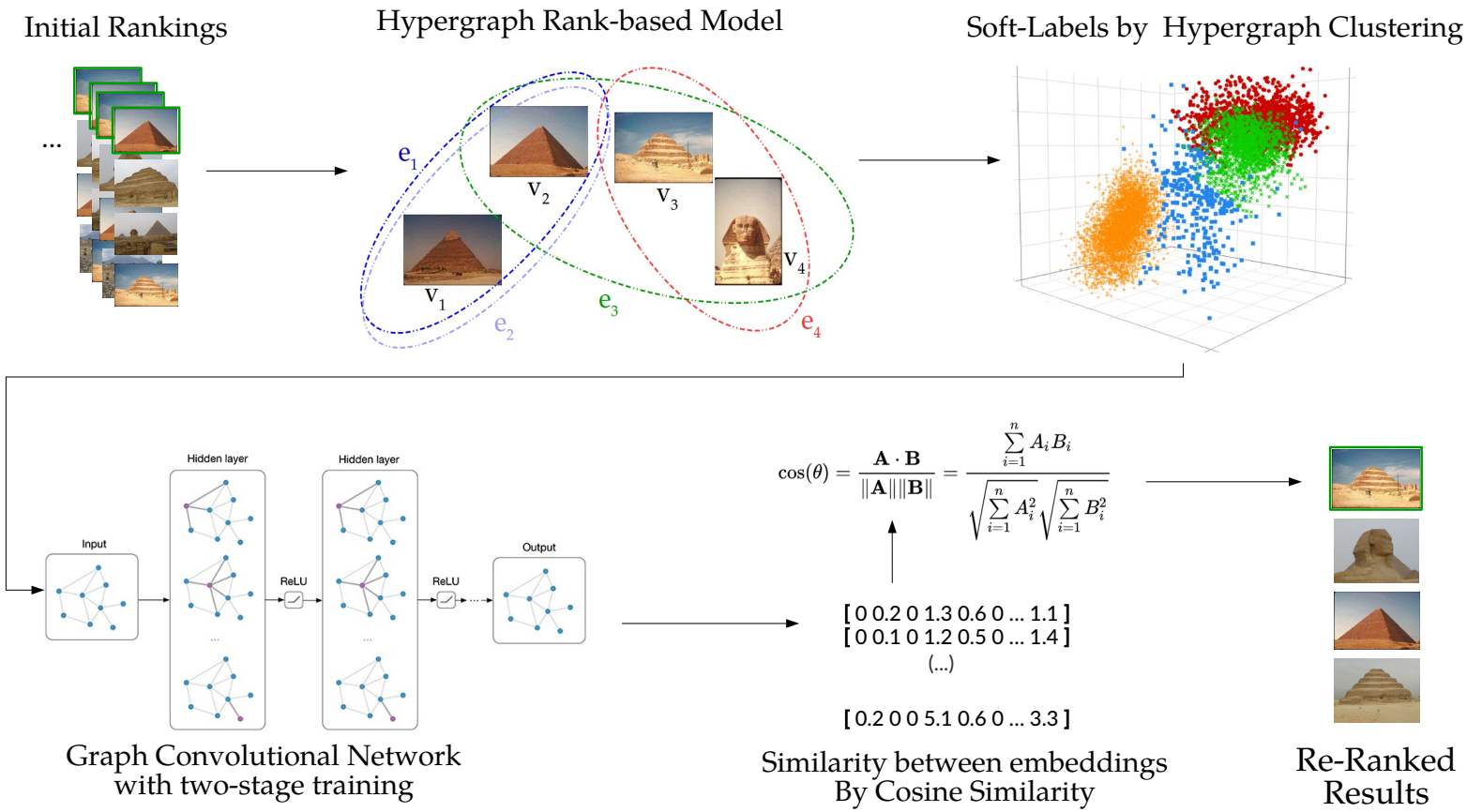
Self-Supervised Graph Convolutional Clustering (SGCC)

Table 3. Comparison of NMI, V-Measure (VM) and ACC values in image datasets between *SGCC*, classic and recent clustering methods.

Method	Input	Corel5K			CUB200			Flowers			MPEG-7		
		NMI	VM	ACC	NMI	VM	ACC	NMI	VM	ACC	NMI	VM	ACC
K-Means (1967)	X	89.38 ±00.46	88.71 ±00.42	82.12 ±01.07	67.07 ±00.19	66.45 ±00.14	41.03 ±00.44	73.23 ±00.93	72.90 ±00.99	71.85 ±02.06	79.32 ±00.32	78.65 ±00.37	59.55 ±00.93
Agglomerative (1983)	X	91.03	90.65	86.68	67.06	66.24	42.03	78.06	77.03	72.64	90.43	86.76	59.00
HDBSCAN (2017)	X	75.66	54.91	35.28	49.94	14.89	04.30	38.60	15.98	13.52	90.16	79.32	64.92
FINCH (2019)	X	90.06	81.13	52.32	77.23	25.65	04.57	79.60	66.54	52.20	87.04	83.72	60.64
SDCN (2020)	X & A	87.43 ±00.36	86.95 ±00.32	81.51 ±00.74	62.62 ±00.21	61.23 ±00.18	31.76 ±00.62	67.02 ±00.99	66.73 ±00.99	36.91 ±00.58	-	-	-
MinCutPool (2020)	X & A	85.76 ±00.78	77.71 ±17.28	33.96 ±12.38	-	-	-	72.55 ±02.05	72.46 ±02.07	74.54 ±02.82	30.07 ±36.89	06.59 ±08.62	00.02 ±00.69
Proposed Approach													
SGCC (GCN)	X & A	91.89 ±00.13	91.79 ±00.13	90.86 ±00.11	69.07 ±00.13	68.21 ±00.12	47.52 ±00.16	80.98 ±00.28	**80.70 ±00.31**	82.68 ±00.47	07.58 ±22.75	02.32 ±06.96	01.76 ±00.99
SGCC (SGC)	X & A	**92.62 ±00.06**	**92.44 ±00.06**	90.80 ±00.04	**69.97 ±00.02**	**69.19 ±00.02**	**48.38 ±00.02**	**81.27 ±00.07**	**81.01 ±00.07**	**83.49 ±00.09**	**96.45 ±00.15**	**96.37 ±00.15**	**94.56 ±00.16**
SGCC (APPNP)	X & A	92.27 ±00.12	92.16 ±00.11	**91.19 ±00.12**	69.68 ±00.13	68.93 ±00.12	47.77 ±00.30	**81.27 ±00.24**	**80.99 ±00.26**	82.86 ±00.33	32.39 ±39.81	12.71 ±20.09	04.82 ±08.77

WACV 2023.

SGRR: Self-Supervised GCN for Re-Ranking



Graph Convolutional Network with two-stage training

Similarity between embeddings By Cosine Similarity

Re-Ranked Results

SIBGRAPI 2024.

SGRR: Self-Supervised GCN for Re-Ranking

TABLE II: Mean Average Precision (MAP) results on the Corel5k dataset for k = 80.

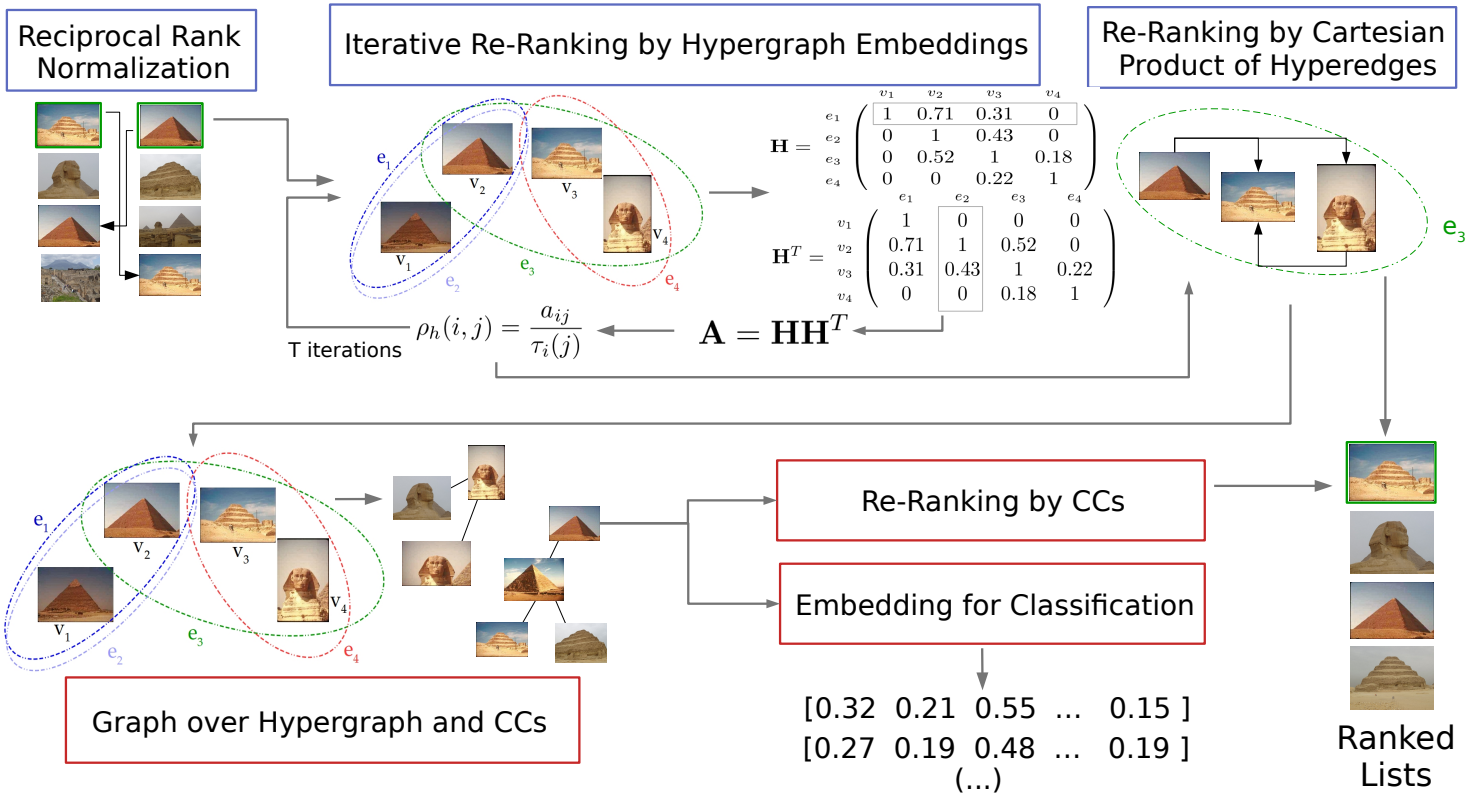
Method	Features					
	RESNET [16]	Rel. Gain	VIT-B16 [17]	Rel. Gain	SWIN-TF [18]	Rel. Gain
Original	62.93%	-	73.76%	-	72.93%	-
RFE [4]	86.08%	+36.78%	91.11%	+23.52%	94.63%	+29.75%
RDPAC [6]	79.73%	+26.69%	86.07%	+16.68%	84.20%	+15.45%
BFSTREE [7]	75.10%	+19.33%	82.41%	+11.97%	80.27%	+10.06%
LHRR [8]	86.85%	+38.01%	**91.40%**	**+23.91%**	95.93%	+31.53%
CPRR [9]	76.07%	+20.88%	83.05%	+12.59%	80.58%	+10.48%
Ours	**89.05% ± 00.12**	**+41.50%**	89.76% ± 00.07	+21.69%	**97.60% ± 00.34**	**+33.82%**

TABLE III: Mean Average Precision (MAP) results on the CUB200 dataset for k = 50.

Method	Features					
	RESNET [16]	Rel. Gain	VIT-B16 [17]	Rel. Gain	SWIN-TF [18]	Rel. Gain
Original	20.55%	-	59.00%	-	56.54%	-
RFE [4]	34.20%	+66.42%	66.37%	+12.49%	66.24%	+17.15%
RDPAC [6]	30.45%	+48.17%	68.07%	+15.37%	70.09%	+23.96%
BFSTREE [7]	27.30%	+32.86%	65.78%	+11.49%	66.31%	+17.27%
LHRR [8]	34.88%	+69.73%	69.64%	+18.03%	70.73%	+25.09%
CPRR [9]	28.37%	+38.05%	66.31%	+12.38%	67.31%	+19.04%
Ours	**38.44% ± 00.11**	**+87.05%**	**70.09% ± 00.06**	**+18.79%**	**76.61% ± 00.11**	**+35.49%**

SIBGRAPI 2024.

Rank Flow Embeeding: Unsupervised & Semi-Supervised



IEEE Transactions on Image Processing 2023.

Aplicações e Hands-On

- **Graph Convolutional Networks (GCN) with Applications: Dr. Lucas Pascotti Valem**
  - Framework Pytorch Geometric
  - Aplicações em Semi-Supervised Classification
  - Classificação de Dados Textuais
  - Classificação de Imagens
  - Link Prediction

Thanks!

Thanks!

daniel.pedronette@unesp.br

References I

[1] F. M. Bianchi, D. Grattarola, L. Livi, and C. Alippi.
Graph neural networks with convolutional ARMA filters.
CoRR, abs/1901.01343, 2019.

[2] T. N. Kipf and M. Welling.
Semi-supervised classification with graph convolutional networks.
In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[3] J. Klicpera, A. Bojchevski, and S. Günnemann.
Predict then propagate: Graph neural networks meet personalized pagerank.
In *International Conference on Learning Representations, ICLR 2019*, 2019.

[4] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio.
Graph attention networks.
In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[5] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger.
Simplifying graph convolutional networks.
In *International Conference on Machine Learning (ICML)*, volume 97, pages 6861–6871, 2019.