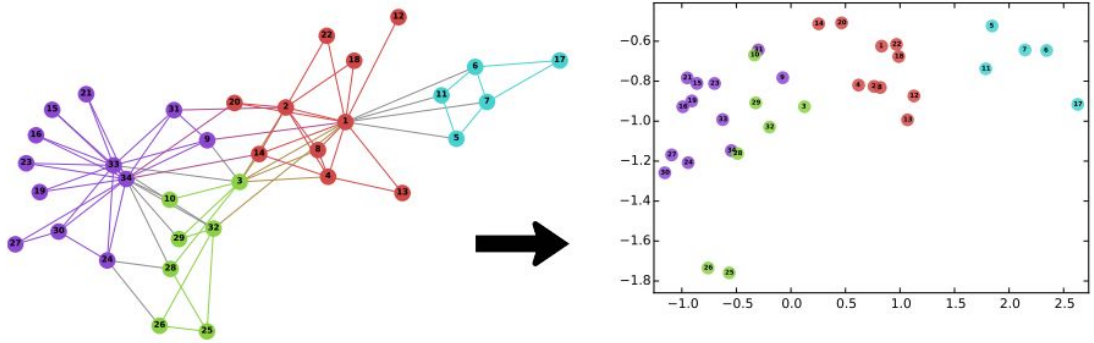
DeepWalk: Online Learning of Social Representations

Complessità nei Sistemi e nelle Reti

Enrico Fini

1. What is DeepWalk?

➔ **A method to learn Latent Representations of Vertices in a Network,** encoding a *sparse, high dimensional* representation (adjacency/weight matrix) into a set of *dense, low dimensional* vectors (embeddings)

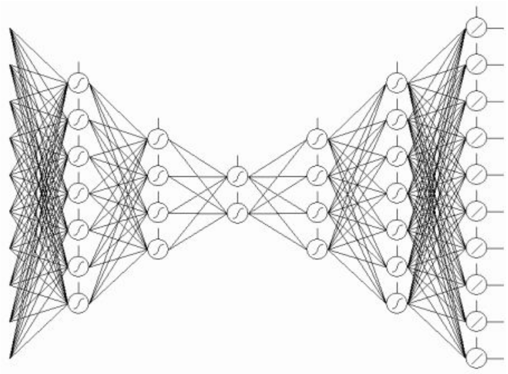


2. Main Contributions

- ➔ **Deep Learning approach to Graph Analysis**
  B. Perozzi et al. drew inspiration from Natural Language Processing to generate unsupervised features of the nodes.
- ➔ **Generalization Capability**
  The resulting representations are general and can be combined with any classifier, yielding good performance.
- ➔ **Parallelizability and Scalability**
  Very efficient algorithm, suitable for analysing real world networks.
- ➔ **Extensive Testing and Strong Performance**
  Evaluation on 3 different datasets shows SOTA results and data efficiency.

3. Latent Representations

- In statistics a **Latent Variable** is a variable that cannot be observed directly but is inferred with a mathematical model from real observations
- Very popular in **Machine Learning**:
  - **Principal Component Analysis**
  - **Word Embeddings**
  - **Autoencoders**
  - **...**



4. Random Walk

➔ **"A Random Walk is a path formed by a sequence of random steps"**, C. Piccardi
It can be seen as an ordered list of nodes where each node is a neighbour of the previous node in the list.

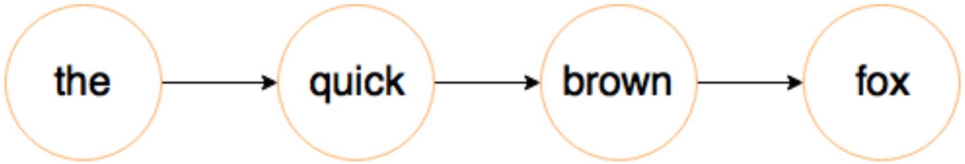
➔ **Interesting properties of short Random Walks:**

- **Locality:** strong connection to the local structure of a network
- **Parallelizability:** several random walkers can explore the network simultaneously
- **Generality:** tolerance to small changes in the graph structure

5. From NLP to Graph Analysis

➔ **Sentences can be seen as short Random Walks:**

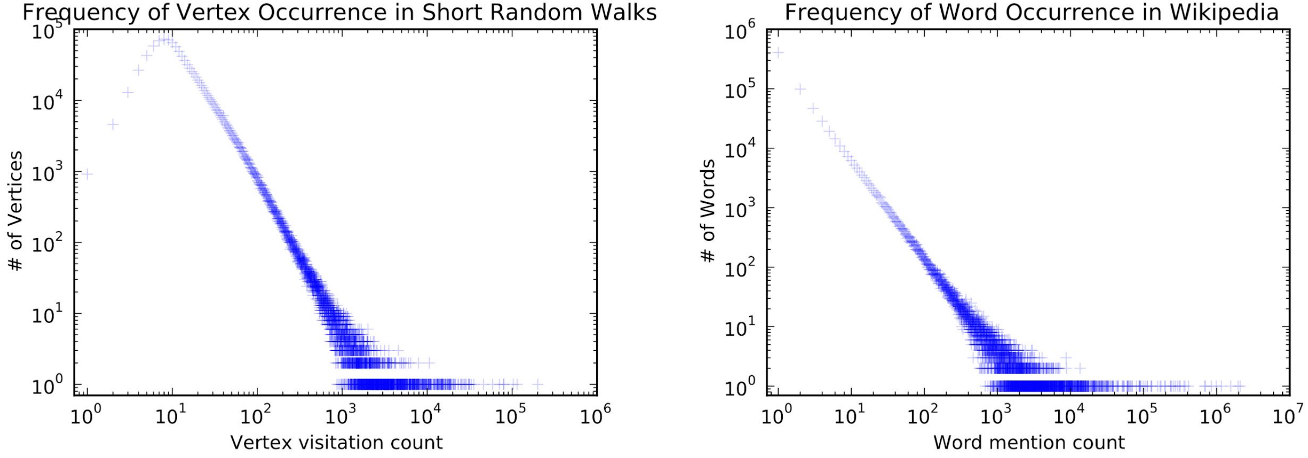
- Word → Node
- Transition between words → Edge



Sentence in a graph representation

5. From NLP to Graph Analysis

➔ **Power Law connection:**



(a) YouTube Social Graph (b) Wikipedia Article Text

6. Language Modeling

➔ Likelihood estimation of a sequence of words:

$$Pr(w_{n+1} | (w_1, w_2, \cdots, w_n))$$

➔ Equivalently, given a mapping function $\Phi : w \in V \rightarrow R^{|V| \times d}$

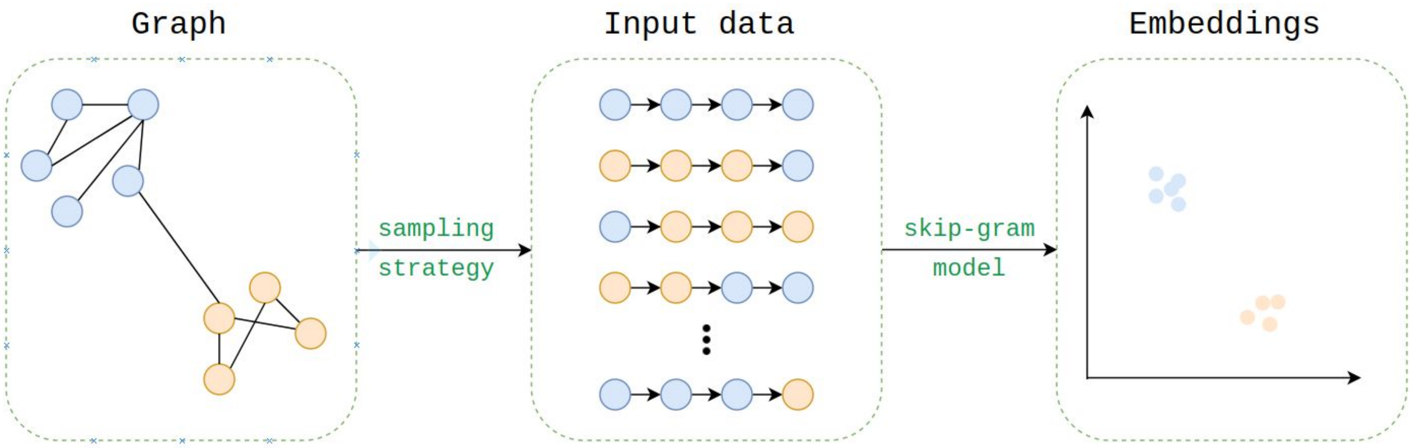
$$Pr(w_{n+1} | (\Phi(w_1), \Phi(w_2), \cdots, \Phi(w_n)))$$

6. Language Modeling

- Instead of using the context to predict the missing word, **use the word to predict the context**
- Use context on both **left** and **right** side of the word
- **Order independence assumption**
- Rewrite as optimization problem on the node embedding:

$$minimize_{\Phi} \quad -log(Pr(\{v_{i-w}, \cdots, v_{i-1}, v_{i+1}, \cdots, v_{i+w}\} | \Phi(v_i)))$$

7. Algorithm



Graph
Input data
Embeddings
sampling strategy
skip-gram model

7. Algorithm

- **Random Walk generator:**
  - For each node in the graph start a random walk with length **t = 40**
  - Repeat **γ = 30** times (training epochs)
  - Random restarts can be used but it doesn't help
- Each sequence generated by a short Random Walk is divided into **subsequences of length 2w + 1, with window size w = 10**

7. Algorithm

Compute error

➔ **Embedding update procedure:**

- **Skip-Gram:**

$$J(\Phi) = -\log \Pr(u_k \mid \Phi(v_j))$$
$$\Phi = \Phi - \alpha * \frac{\partial J}{\partial \Phi}$$

Take the gradient and update

`The quick brown fox jumps over the lazy dog.`

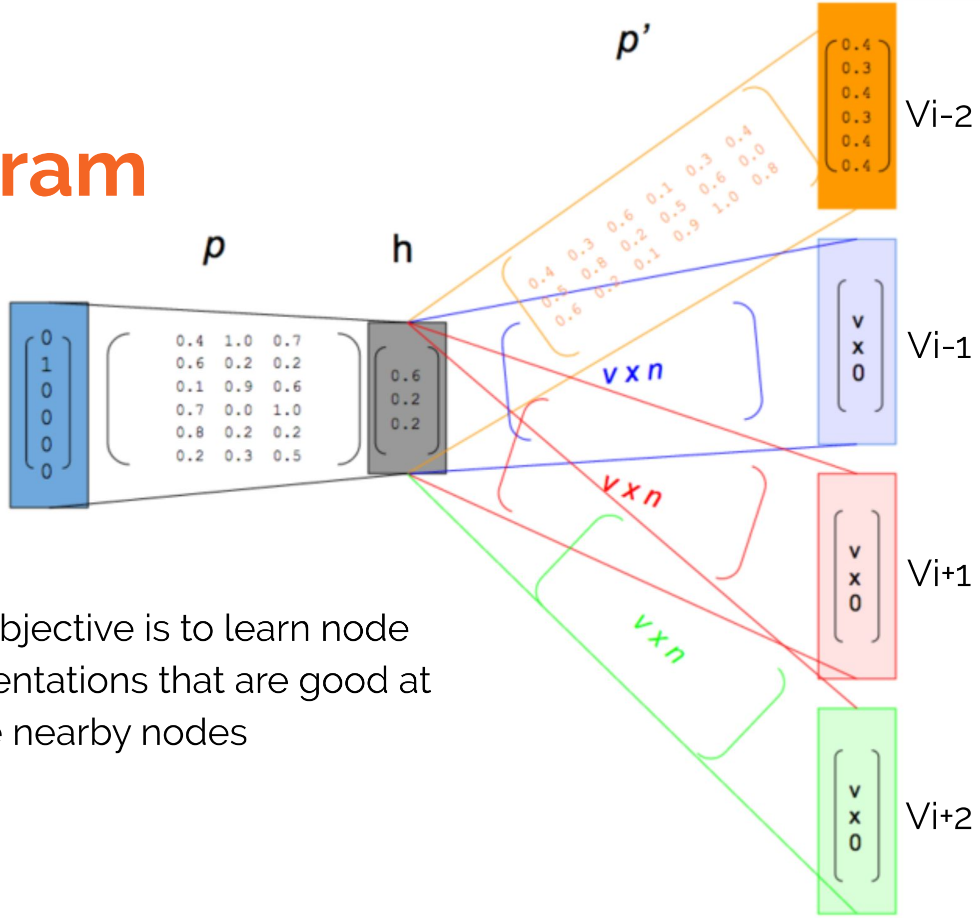
`The quick brown fox jumps over the lazy dog.`

`The quick brown fox jumps over the lazy dog.`

8. Skip-Gram

Target node

Vi

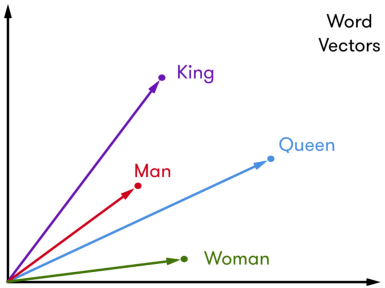


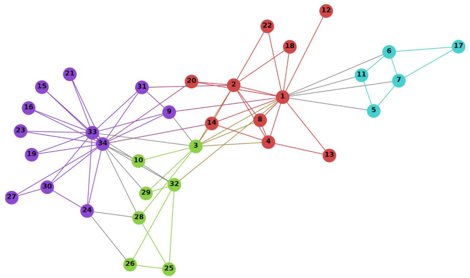
➔ The training objective is to learn node vector representations that are good at predicting the nearby nodes

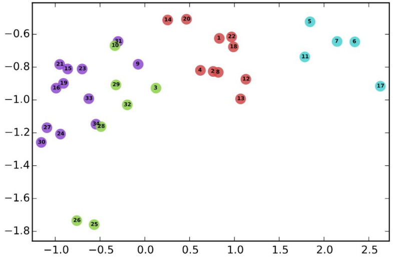
9. Properties

➔ Nodes that belong to the **same community** will have **similar embeddings** (i.e. the Euclidean distance will be small)

➔ The latent representations also capture the **relationships between nodes**







10. Evaluation

- Now that we have a set of features that encode the relationships among nodes we can test them on **multi-label classification**
- Against **5 state of the art baselines** with **Micro/Macro-F1 score**
- On **3 datasets:**

Name	BlogCatalog	Flickr	YouTube
$\|V\|$	10,312	80,513	1,138,499
$\|E\|$	333,983	5,899,882	2,990,443
$\|\mathcal{Y}\|$	39	195	47
Labels	Interests	Groups	Groups

11. Results (BlogCatalog)

	% Labeled Nodes	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1(%)	DeepWalk	**36.00**	**38.20**	**39.60**	**40.30**	**41.00**	**41.30**	41.50	41.50	42.00
	SpectralClustering	31.06	34.95	37.27	38.93	39.97	40.99	**41.66**	**42.42**	**42.62**
	EdgeCluster	27.94	30.76	31.85	32.99	34.12	35.00	34.63	35.99	36.29
	Modularity	27.35	30.74	31.77	32.97	34.09	36.13	36.08	37.23	38.18
	wvRN	19.51	24.34	25.62	28.82	30.37	31.81	32.19	33.33	34.28
	Majority	16.51	16.66	16.61	16.70	16.91	16.99	16.92	16.49	17.26
Macro-F1(%)	DeepWalk	**21.30**	**23.80**	25.30	26.30	27.30	27.60	27.90	28.20	28.90
	SpectralClustering	19.14	23.57	**25.97**	**27.46**	**28.31**	**29.46**	**30.13**	**31.38**	**31.78**
	EdgeCluster	16.16	19.16	20.48	22.00	23.00	23.64	23.82	24.61	24.92
	Modularity	17.36	20.00	20.80	21.85	22.65	23.41	23.89	24.20	24.97
	wvRN	6.25	10.13	11.64	14.24	15.86	17.18	17.98	18.86	19.57
	Majority	2.52	2.55	2.52	2.58	2.58	2.63	2.61	2.48	2.62

Table 2: Multi-label classification results in BlogCatalog

11. Results (Flickr)

	% Labeled Nodes	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1(%)	DeepWalk	**32.4**	**34.6**	**35.9**	**36.7**	**37.2**	**37.7**	**38.1**	**38.3**	**38.5**	**38.7**
	SpectralClustering	27.43	30.11	31.63	32.69	33.31	33.95	34.46	34.81	35.14	35.41
	EdgeCluster	25.75	28.53	29.14	30.31	30.85	31.53	31.75	31.76	32.19	32.84
	Modularity	22.75	25.29	27.3	27.6	28.05	29.33	29.43	28.89	29.17	29.2
	wvRN	17.7	14.43	15.72	20.97	19.83	19.42	19.22	21.25	22.51	22.73
	Majority	16.34	16.31	16.34	16.46	16.65	16.44	16.38	16.62	16.67	16.71
Macro-F1(%)	DeepWalk	**14.0**	17.3	**19.6**	**21.1**	**22.1**	**22.9**	**23.6**	**24.1**	**24.6**	**25.0**
	SpectralClustering	13.84	**17.49**	19.44	20.75	21.60	22.36	23.01	23.36	23.82	24.05
	EdgeCluster	10.52	14.10	15.91	16.72	18.01	18.54	19.54	20.18	20.78	20.85
	Modularity	10.21	13.37	15.24	15.11	16.14	16.64	17.02	17.1	17.14	17.12
	wvRN	1.53	2.46	2.91	3.47	4.95	5.56	5.82	6.59	8.00	7.26
	Majority	0.45	0.44	0.45	0.46	0.47	0.44	0.45	0.47	0.47	0.47

Table 3: Multi-label classification results in Flickr

11. Results (YouTube)

	% Labeled Nodes	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1(%)	DeepWalk	**37.95**	**39.28**	**40.08**	**40.78**	**41.32**	**41.72**	**42.12**	**42.48**	**42.78**	**43.05**
	SpectralClustering	—	—	—	—	—	—	—	—	—	—
	EdgeCluster	23.90	31.68	35.53	36.76	37.81	38.63	38.94	39.46	39.92	40.07
	Modularity	—	—	—	—	—	—	—	—	—	—
	wvRN	26.79	29.18	33.1	32.88	35.76	37.38	38.21	37.75	38.68	39.42
	Majority	24.90	24.84	25.25	25.23	25.22	25.33	25.31	25.34	25.38	25.38
Macro-F1(%)	DeepWalk	**29.22**	**31.83**	**33.06**	**33.90**	**34.35**	**34.66**	**34.96**	**35.22**	**35.42**	**35.67**
	SpectralClustering	—	—	—	—	—	—	—	—	—	—
	EdgeCluster	19.48	25.01	28.15	29.17	29.82	30.65	30.75	31.23	31.45	31.54
	Modularity	—	—	—	—	—	—	—	—	—	—
	wvRN	13.15	15.78	19.66	20.9	23.31	25.43	27.08	26.48	28.33	28.89
	Majority	6.12	5.86	6.21	6.1	6.07	6.19	6.17	6.16	6.18	6.19

Table 4: Multi-label classification results in YouTube

12. BONUS: my implementation

- Around 80 lines of **Python** code
- **Networkx** for the graph
- **PyTorch** for the neural network
- Tested on **Zachary's karate club** network with 2d embeddings

12. BONUS: my implementation

- ➔ The euclidean distance between central and peripheral nodes seems to be constant
- ➔ Average distance of nodes **15, 16, 19, 21, 23, 27** from node **34** is **6.23**
- ➔ Average distance of nodes **12, 13, 18, 22** from node **1** is **6.52**