GRAPH/NODE EMBEDDING

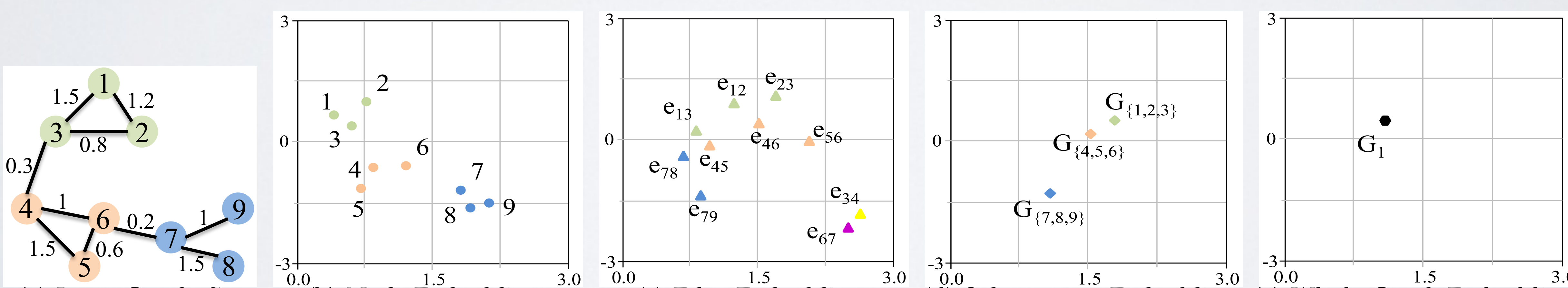
Goyal, P., & Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, *151*, 78-94.

Cai, H., Zheng, V. W., & Chang, K. C. C. (2018). A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering*, *30*(9), 1616-1637.

VARIANT

- We can differentiate:
  - Node embedding
  - Edge Embedding
  - Substructure embedding
  - Whole graph Embedding

- In this course, only *node embedding* (often called graph embedding)



(a) Input Graph G_1 (b) Node Embedding (c) Edge Embedding (d) Substructure Embedding (e) Whole-Graph Embedding

Cai, H., Zheng, V. W., & Chang, K. C. C. (2018). A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering*, *30*(9), 1616-1637.

NAMES

- Representation learning on networks
  - **Representation learning = feature learning**, as opposed to **manual feature engineering (heuristics)**
- Embedding => Latent space

IN CONCRETE TERMS

- A graph is composed of
  - Nodes (possibly with labels)
  - Edges (possibly directed, weighted, with labels)
- A graph/node embedding technique in **d** dimensions will assign a vector of length **d** to each node, that will be useful for **what we want to do with the graph**.
  - It captures some aspect of the network structure
- A vector can be assigned to an edge *(u,v)* by combining vectors of *u* and *v*

WHAT TO DO WITH EMBEDDINGS?

- Two possible ways to use an embedding:
  - Unsupervised learning:
    - The *distance* between vectors in the embedding is used for *something*
  - Supervised learning:
    - Algorithm learn to predict *something* from the features in the embedding

WHAT CAN WE DO WITH EMBEDDINGS ?

EMBEDDING TASKS

- Common tasks:
  - Link prediction (supervised)
  - Graph reconstruction (unsupervised link prediction ? / ad hoc)
  - Community detection (unsupervised)
  - Node classification (supervised community detection ?)
  - Role definition (Variant of node classification, can be unsupervised)
  - Visualisation (distances, like unsupervised)

OVERVIEW OF MOST POPULAR METHODS

PRE-DEEPWALK

MATRIX DECOMPOSITION

LE: LAPLACIAN EIGENMAPS

- Introduced 2001
- Objective function:
  - $y^* = \min \sum_{i \neq j} \|y_i - y_j\|^2 S_{ij}$
    - y^*: optimal embedding
    - y_i: embedding of node i
    - S_{ij}: similarity between nodes *i* and *j (A, heuristic, …)*
- Minimize the product between **distance in the embedding** and **similarity in the graph**
  - If nodes are *similar*, they must be *close* in the embedding

LE: LAPLACIAN EIGENMAPS

- $y* = \min \sum_{i \neq j} \|y_i - y_j\|^2 S_{ij}$
  - Solution: d eigenvectors of lowest eigenvalues of $D^{-1/2}LD^{-1/2}$
  - L: Laplacian, with S=A

Goyal, P., & Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, *151*, 78-94.

HOPE: HIGHER-ORDER PROXIMITY PRESERVED EMBEDDING

- Preserve a proximity matrix
- $y* = \min \sum_{i,j} |S_{ij} - y_i y_j^T|$
- S can be the adjacency matrix, or number of common neighbors, Adamic Adar, etc.
- As similarity tends towards 0, embedding vectors must tend towards orthogonality (orthogonal vectors: $y_i y_j^T = 0$)

Goyal, P., & Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, *151*, 78-94.

LLE: LOCALLY LINEAR EMBEDDING

- Introduced 2000
- A node features can be represented as a linear combination of its neighbors'
  - $y_i = \sum_j A_{ij} y_j$
- Objective function:
  - $y* = \min \sum_i \|y_i - \sum_j A_{ij} y_j\|^2$

Goyal, P., & Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, *151*, 78-94.

RANDOM WALKS BASED

DEEPWALK

- The first *Random Walk+Neural Networks* graph embedding method.
  - First of a long series
- Adaptation of **word2vec/skipgram** to graphs

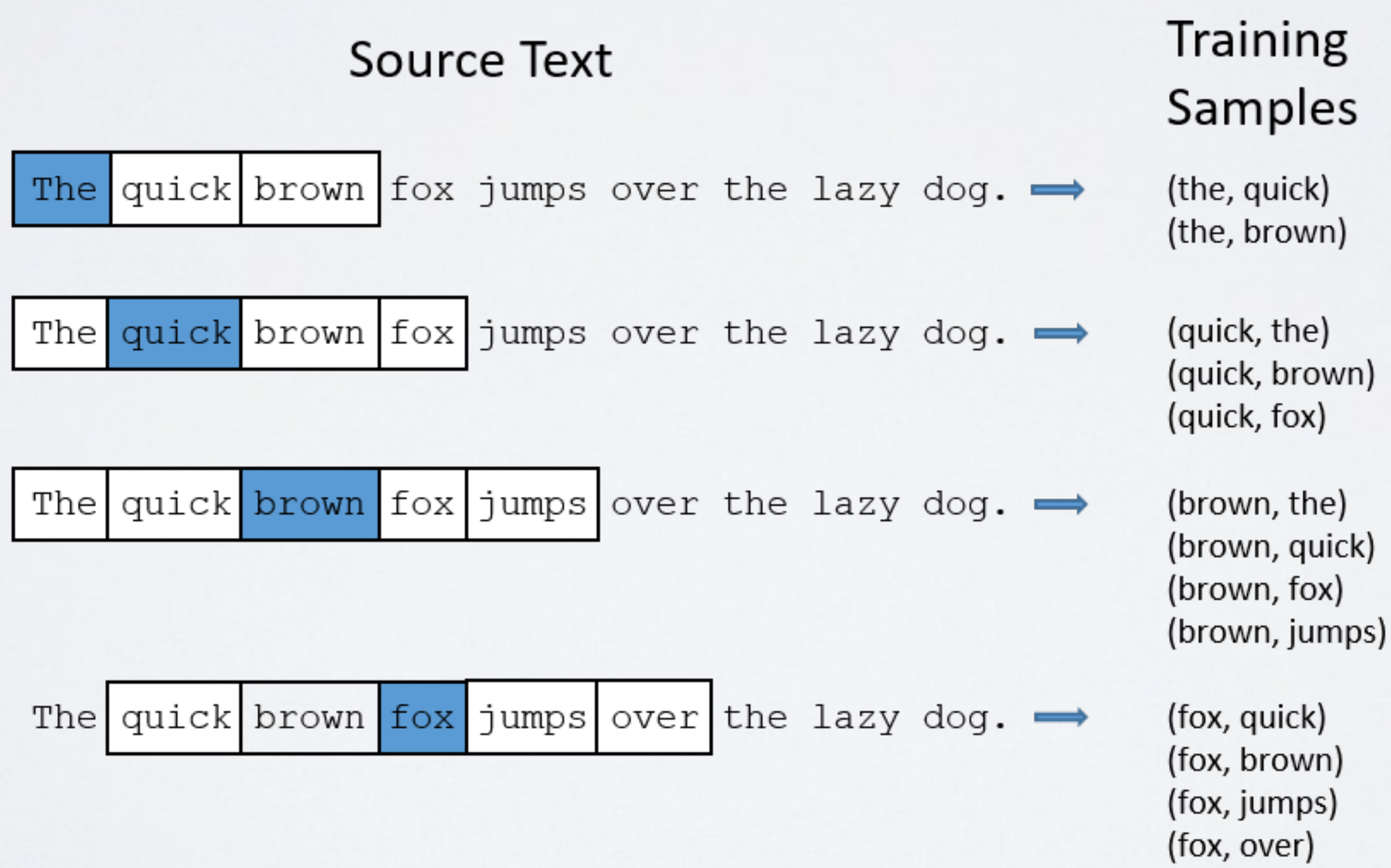
Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710). ACM.

SKIPGRAM

Word embedding

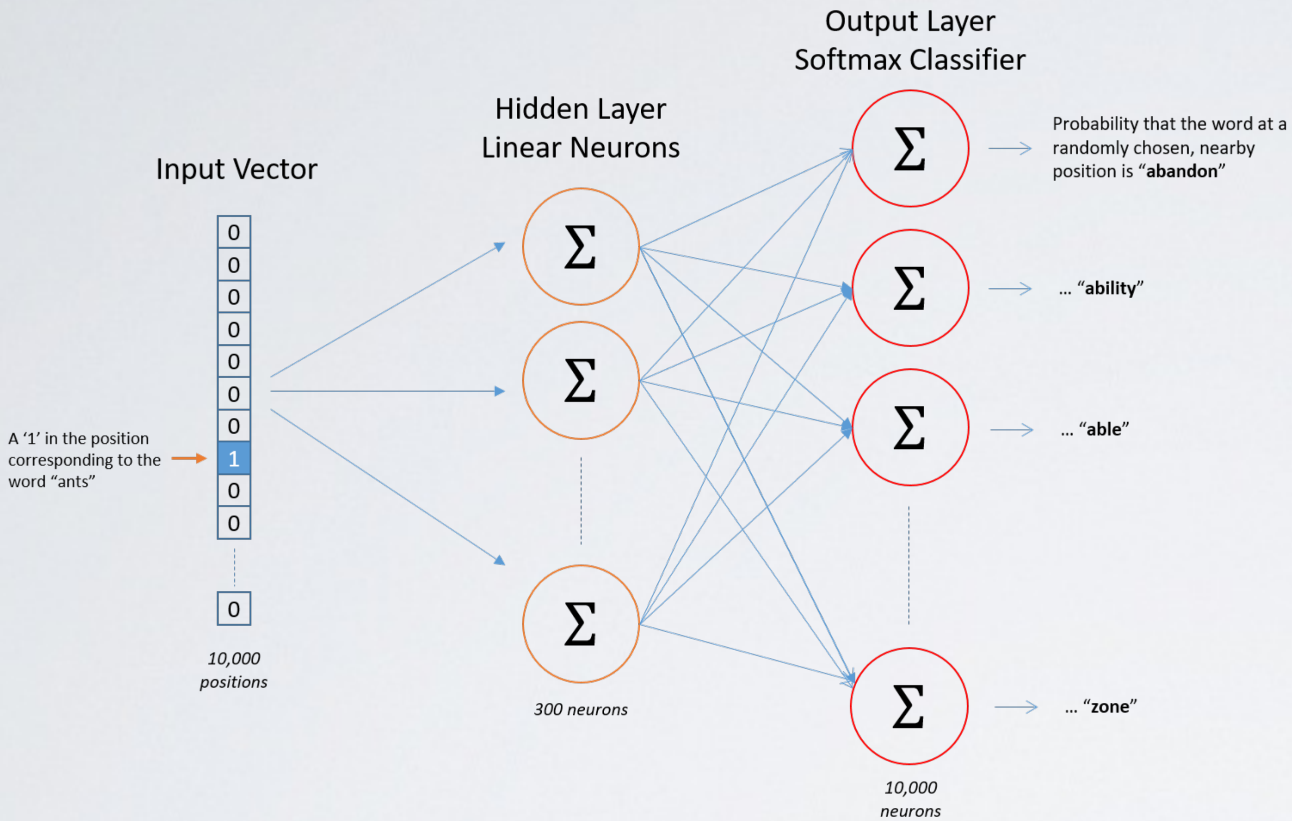
Corpus => Word = vectors

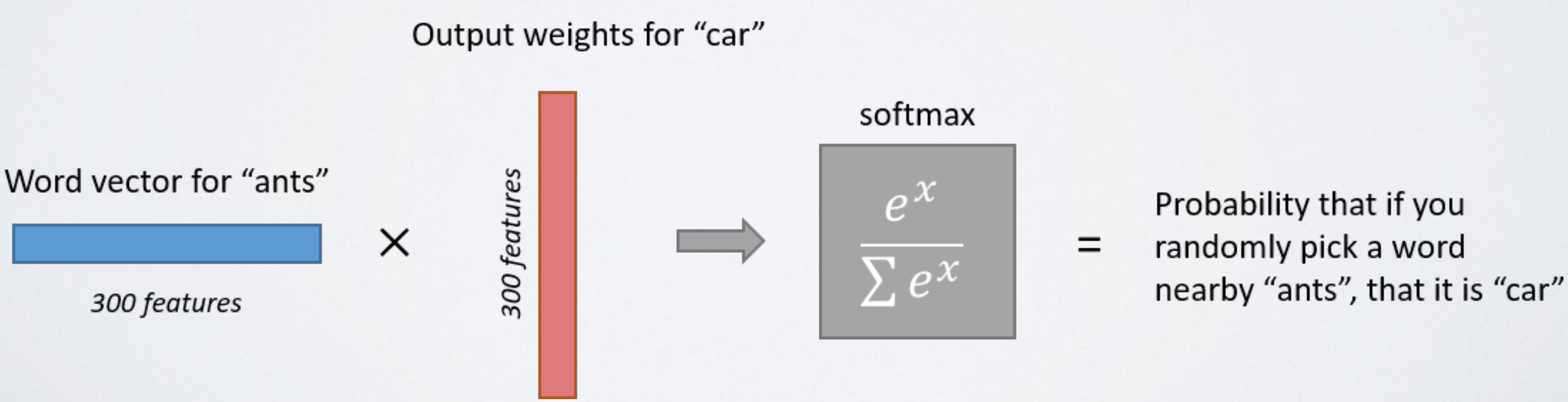
Similar embedding= similar **context**



[http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/]

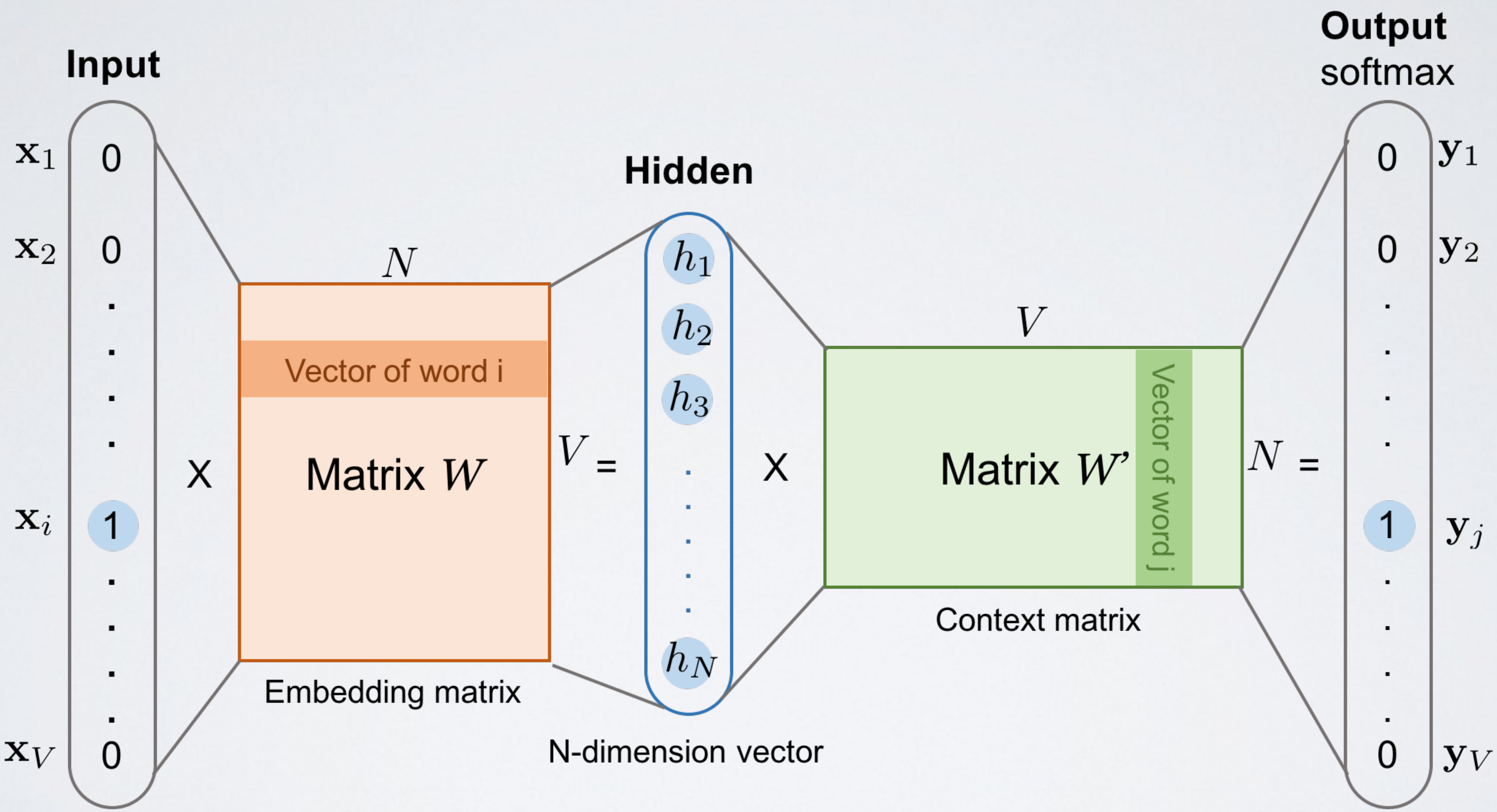
SKIPGRAM





https://towardsdatascience.com/word2vec-skip-gram-model-part-1-intuition-78614e4d6e0b

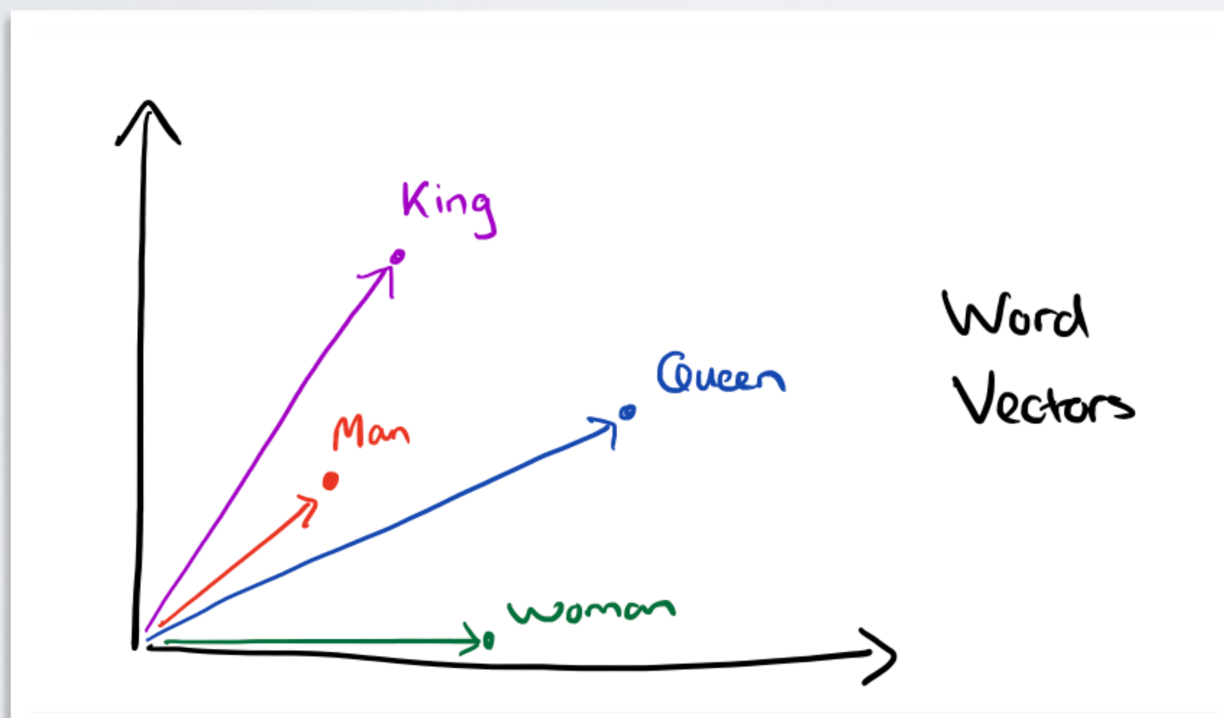
SKIPGRAM

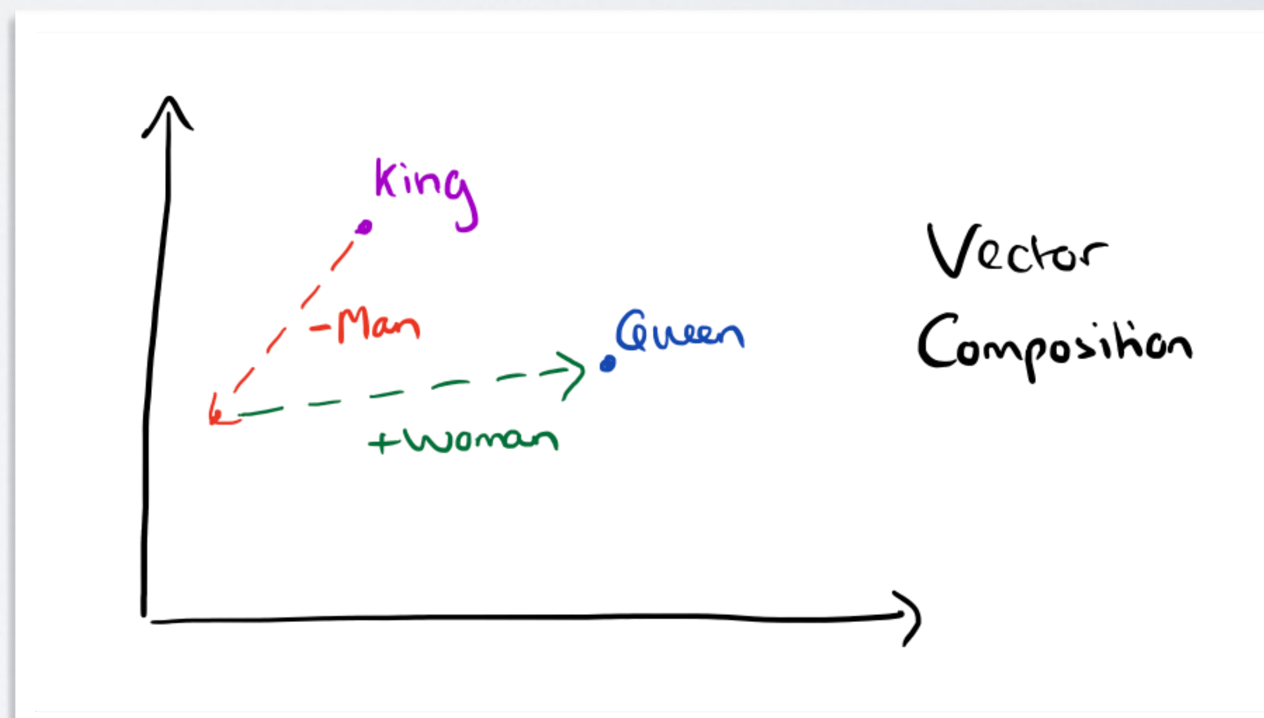


N=embedding size. **V**=vocabulary size

https://towardsdatascience.com/word2vec-skip-gram-model-part-1-intuition-78614e4d6e0b

SKIPGRAM





[https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/]

SKIPGRAM

Table 8: *Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).*

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

[https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/]

GENERIC "SKIPGRAM"

- Algorithm that takes an input:
  - The element to embed
  - A list of "context" elements
- Provide as output:
  - An embedding with interesting properties
    - Works well for machine learning
    - Similar elements are close in the embedding
    - Somewhat preserves the overall structure

DEEPWALK

- Skipgram for graphs:
  - 1)Generate "sentences" using random walks
  - 2)Apply Skipgram

- Parameters:
  - Embedding dimensions *d*
  - Context size
  - More technical parameters: length of random walks, number of walks starting from each node, etc.

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710). ACM.

NODE2VEC

- Use biased random walk to tune the context to capture *what we want*
  - "Breadth first" like RW => local neighborhood (edge probability ?)
  - "Depth-first" like RW => global structure ? (Communities ?)
  - 2 parameters to tune:
    - **p**: bias towards revisiting the previous node
    - **q**: bias towards exploring undiscovered parts of the network

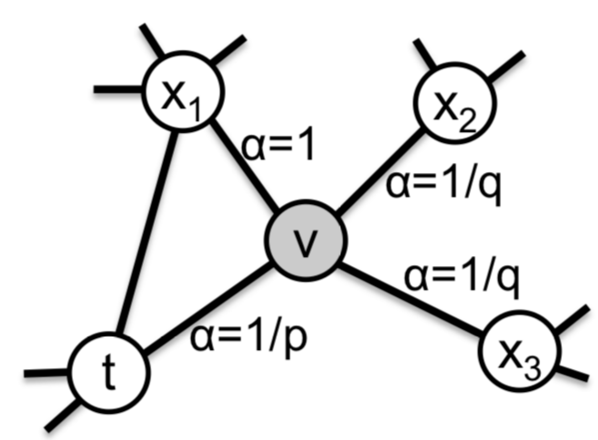


Figure 2: Illustration of the random walk procedure in *node2vec*. The walk just transitioned from t to v and is now evaluating its next step out of node v. Edge labels indicate search biases α.

Grover, A., & Leskovec, J. (2016, August). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 855-864). ACM.

RANDOM WALK METHODS

- What is the objective function ?
- How to interpret the distance between nodes in the embedding ?

RANDOM WALK METHODS

Approximately

$$y = \min \sum_{(i,j)} p(n_j|n_i) - \sigma(y_i y_j^T)$$

with $p(w_j|w_i)$ the probability to encounter node n_j in a random walk of a chosen length starting from node n_i. Its objective is therefore to make the distance in the embedding proportional to a random walk based distance in the graph.

with σ the softmax function defined as $\frac{e^x}{\sum e^x}$, a function commonly used in neural networks to add non-linearity and to ensure that the solution is a probability.

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*.

RANDOM WALK METHODS

- Scalability:
  - Skipgram uses techniques from machine learning developed for very large datasets: highly **scalable** (not necessarily *fast* or *cost efficient*)
- Matrix factorization methods require the similarity matrix S as input
  - Computing all random walk distance: $\mathcal{O}(n^2)$
  - k random walks of length ℓ from each node: $\mathcal{O}(n)$

ENCODER DECODER FRAMEWORK

Minimize a global loss defined as:

$$L = \sum_{(v_i, v_j) \in E} \ell(DEC(z_i, z_j), s_{\mathcal{G}}(v_i, v_j))$$

DEC: Decoder function (e.g., $DEC(z_i, z_j) = z_i^T z_j$)

$s_{\mathcal{G}}$: Ground truth similarity (e.g., $s_{\mathcal{G}(v_i, v_j)} = A_{ij}$)

ℓ: Chosen loss function (e.g., $\ell(a, b) = |a - b|$)

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*.

ENCODER DECODER FRAMEWORK

Type	Method	Decoder	Proximity measure	Loss function (ℓ)
Matrix factorization	Laplacian Eigenmaps [4]	$\|\mathbf{z}_i - \mathbf{z}_j\|_2^2$	general	$\text{DEC}(\mathbf{z}_i, \mathbf{z}_j) \cdot s_{\mathcal{G}}(v_i, v_j)$
	Graph Factorization [1]	$\mathbf{z}_i^\top \mathbf{z}_j$	$\mathbf{A}_{i,j}$	$\|\text{DEC}(\mathbf{z}_i, \mathbf{z}_j) - s_{\mathcal{G}}(v_i, v_j)\|_2^2$
	GraRep [9]	$\mathbf{z}_i^\top \mathbf{z}_j$	$\mathbf{A}_{i,j}, \mathbf{A}_{i,j}^2, ..., \mathbf{A}_{i,j}^k$	$\|\text{DEC}(\mathbf{z}_i, \mathbf{z}_j) - s_{\mathcal{G}}(v_i, v_j)\|_2^2$
	HOPE [44]	$\mathbf{z}_i^\top \mathbf{z}_j$	general	$\|\text{DEC}(\mathbf{z}_i, \mathbf{z}_j) - s_{\mathcal{G}}(v_i, v_j)\|_2^2$
Random walk	DeepWalk [46]	$\frac{e^{\mathbf{z}_i^\top \mathbf{z}_j}}{\sum_{k \in \mathcal{V}} e^{\mathbf{z}_i^\top \mathbf{z}_k}}$	$p_{\mathcal{G}}(v_j \| v_i)$	$-s_{\mathcal{G}}(v_i, v_j) \log(\text{DEC}(\mathbf{z}_i, \mathbf{z}_j))$
	node2vec [27]	$\frac{e^{\mathbf{z}_i^\top \mathbf{z}_j}}{\sum_{k \in \mathcal{V}} e^{\mathbf{z}_i^\top \mathbf{z}_k}}$	$p_{\mathcal{G}}(v_j \| v_i)$ (biased)	$-s_{\mathcal{G}}(v_i, v_j) \log(\text{DEC}(\mathbf{z}_i, \mathbf{z}_j))$

$p_{\mathcal{G}}(v_j | v_i)$: probability of visiting v_j on a fixed-length random walk started from v_i

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*.

SOME REMARKS ON WHAT ARE EMBEDDINGS

ADJACENCY MATRIX

- An adjacency matrix is an “embedding”… in high dimension
- That represents the structural equivalence
  - 2 nodes have similar “embeddings” if they have similar neighborhoods
  - Distance=># of different neighbors (Manhattan Distance)
- Standard dimensionality reduction (T-SNE, PCA) of this matrix?
  - Small dimensions
  - But still unintuitive notion of distance

GRAPH LAYOUT

- Graph layouts are also embeddings.
  - Force layout, kamada-kawai ….
- They try to put connected nodes close to each other and non-connected ones "not close"
- Problem: they usually try to avoid overlaps
- Often not scalable

NODE EMBEDDING: VISUALIZATION

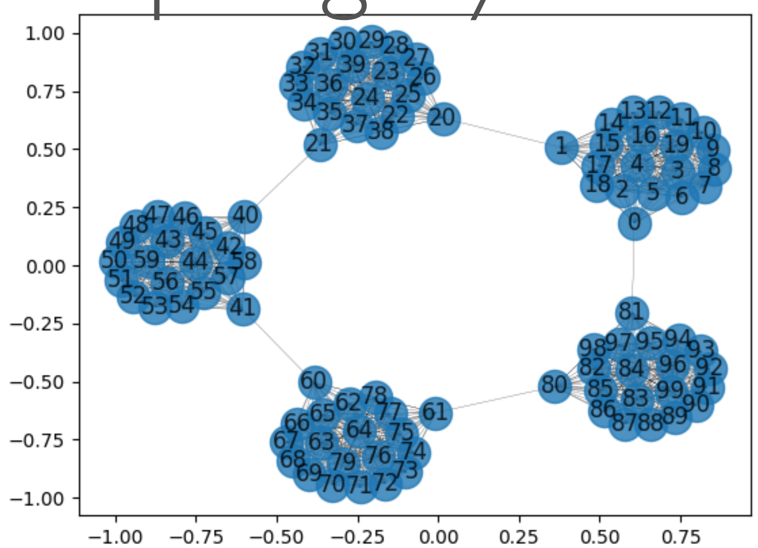
FROM D TO 2

- Graph embedding can be used to visualize graphs
- Requires to reduce the embedding from d to 2
  - TSNE
  - PCA
  - …
- Interpretable positions of nodes
- But not necessarily optimized for human reading

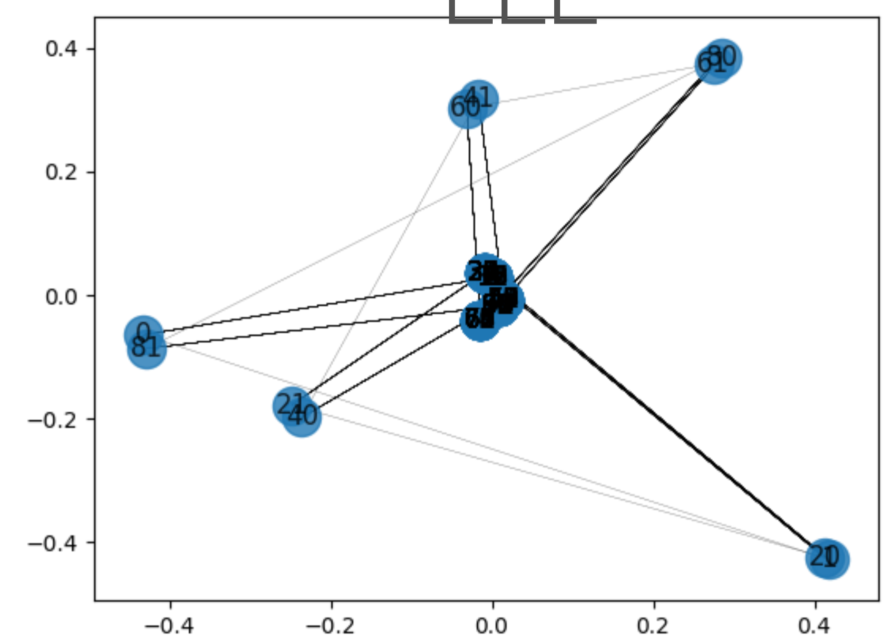
CLIQUE RING

5 cliques of size 20 with 1 edge between them

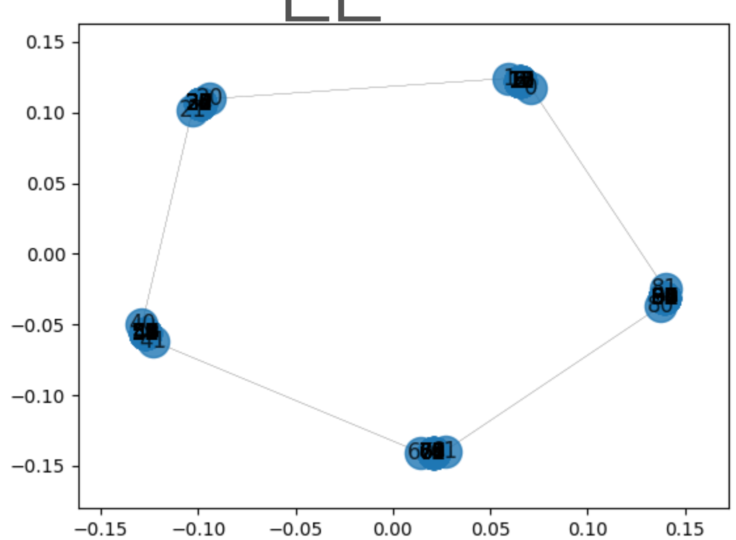
Spring layout



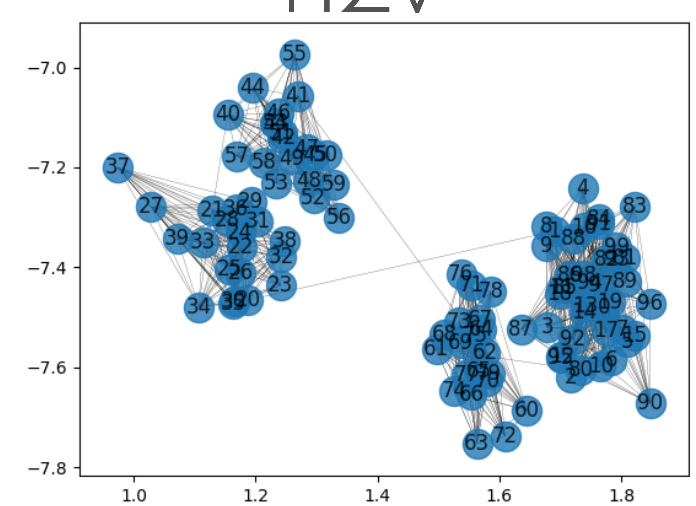
LLE



LE



n2v

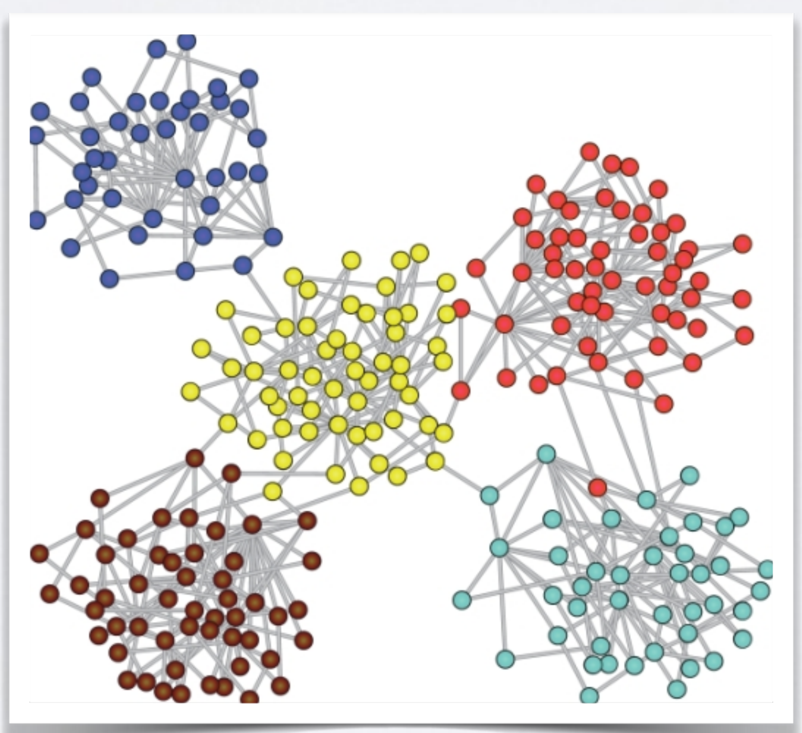


NODE EMBEDDING: COMMUNITY DETECTION

CLUSTERING EMBEDDINGS

- Many algorithm exists for **clustering** non-network data
  - K-means, DBscan, etc.

- Clustering: group nodes that are close in the feature space.



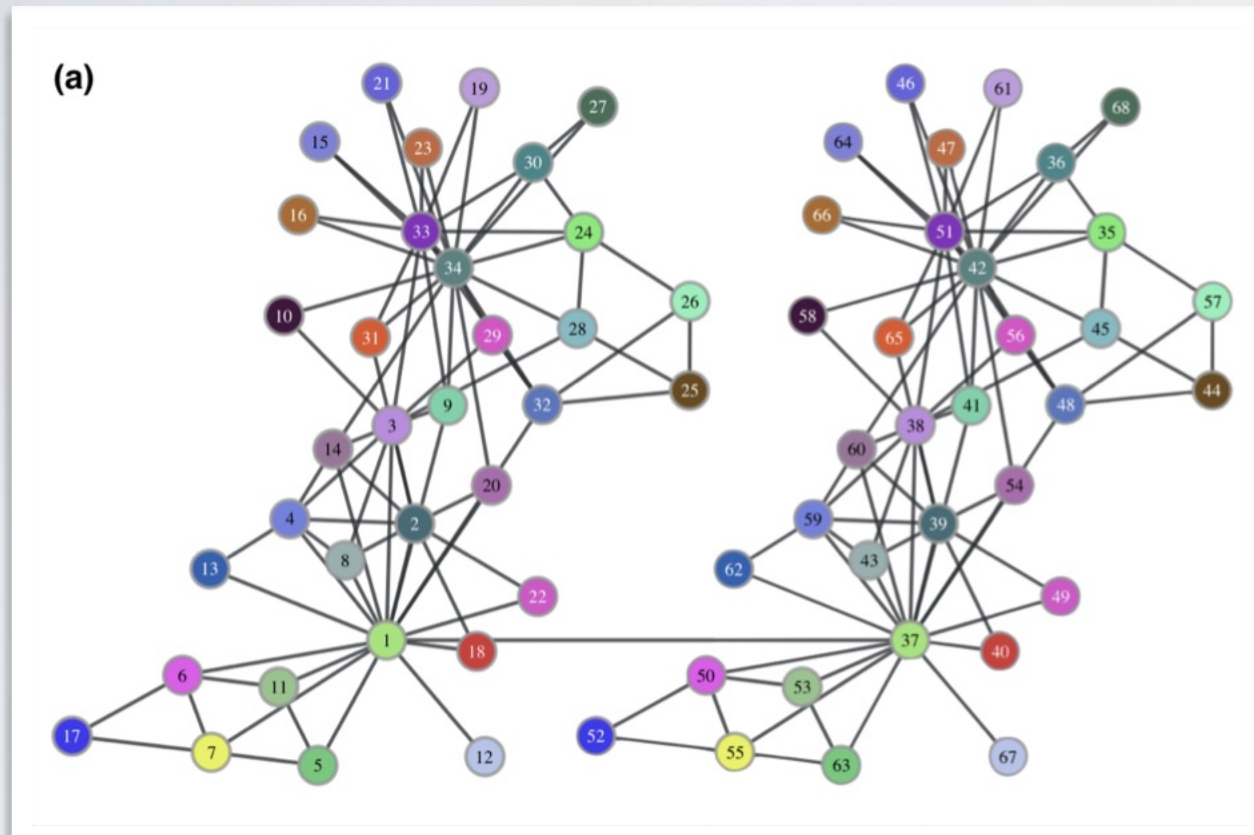
EMBEDDING ROLES

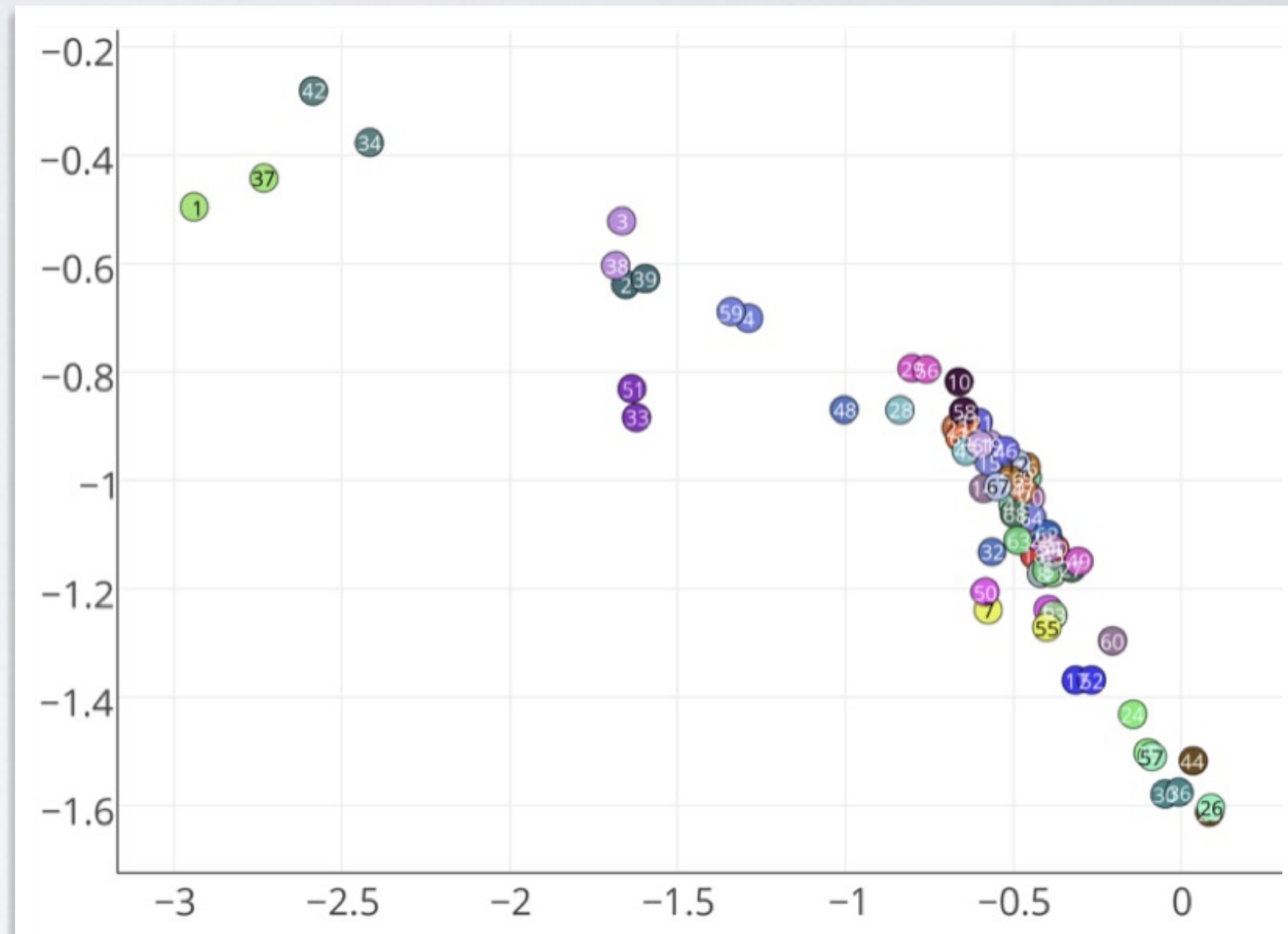
STRUC2VEC/ROLE2VEC

- In node2vec/Deepwalk, the context collected by RW contains the **labels** of encountered nodes
- Instead, we could memorize the **properties** of the nodes: attributes if available, or computed attributes (degrees, CC, …)
- =>Nodes with a same context will be nodes in a same "position" in the graph
- =>Capture the role of nodes instead of proximity

Ribeiro, L. F., Saverese, P. H., & Figueiredo, D. R. (2017, August). struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 385-394). ACM.

STRUCT2VEC : DOUBLE ZKC





(c) *struc2vec*

Ribeiro, L. F., Saverese, P. H., & Figueiredo, D. R. (2017, August). struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 385-394). ACM.

NODE CLASSIFICATION WITH EMBEDDINGS

NODE CLASSIFICATION

- To each node is associated a vector in the embedding
  - This vector corresponds to topological features of the node, used instead of, for instance, centralities
  - Both types of features can be combined
- As usual, a classifier can be trained using those features

NODE CLASSIFICATION

Algorithm	Dataset		
	BlogCatalog	PPI	Wikipedia
Spectral Clustering	0.0405	0.0681	0.0395
DeepWalk	0.2110	0.1768	0.1274
LINE	0.0784	0.1447	0.1164
node2vec	**0.2581**	**0.1791**	**0.1552**
node2vec settings (p,q)	0.25, 0.25	4, 1	4, 0.5
Gain of *node2vec* [%]	**22.3**	**1.3**	**21.8**

Controversies…

Grover, A., & Leskovec, J. (2016, August). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 855-864). ACM.

LINK PREDICTION WITH EMBEDDINGS

Sinha, A., Cazabet, R., & Vaudaine, R. (2018, December). Systematic Biases in Link Prediction: comparing heuristic and graph embedding based methods. In *International Conference on Complex Networks and their Applications* (pp. 81-93). Springer, Cham.

UNSUPERVISED LINK PREDICTION

- Unsupervised link prediction **from embeddings**
- =>Compute the distance between nodes in the embedding
- =>Use it as a similarity score

SUPERVISED LINK PREDICTION

- Supervised link prediction **from embeddings**
- =>embeddings provide features for nodes (nb features: dimensions)
  - Combine nodes features to obtain edge features
- =>Train a classifier to predict edges based on features from the embedding

SUPERVISED LINK PREDICTION

Operator	Result
Average	$(\mathbf{a} + \mathbf{b})/2$
Concat	$[\mathbf{a}_1, \ldots, \mathbf{a}_d, \mathbf{b}_1, \ldots, \mathbf{b}_d]$
Hadamard	$[\mathbf{a}_1 * \mathbf{b}_1, \ldots, \mathbf{a}_d * \mathbf{b}_d]$
Weighted L1	$[\lvert\mathbf{a}_1 - \mathbf{b}_1\rvert, \ldots, \lvert\mathbf{a}_d - \mathbf{b}_d\rvert]$
Weighted L2	$[(\mathbf{a}_1 - \mathbf{b}_1)^2, \ldots, (\mathbf{a}_d - \mathbf{b}_d)^2]$

Combining nodes vectors into edge vectors

SUPERVISED LINK PREDICTION

- How well does it works ?
- According to creators articles
  - Node2vec (2016)
  - VERSE (2018)
- =>These methods are better than the state of the art

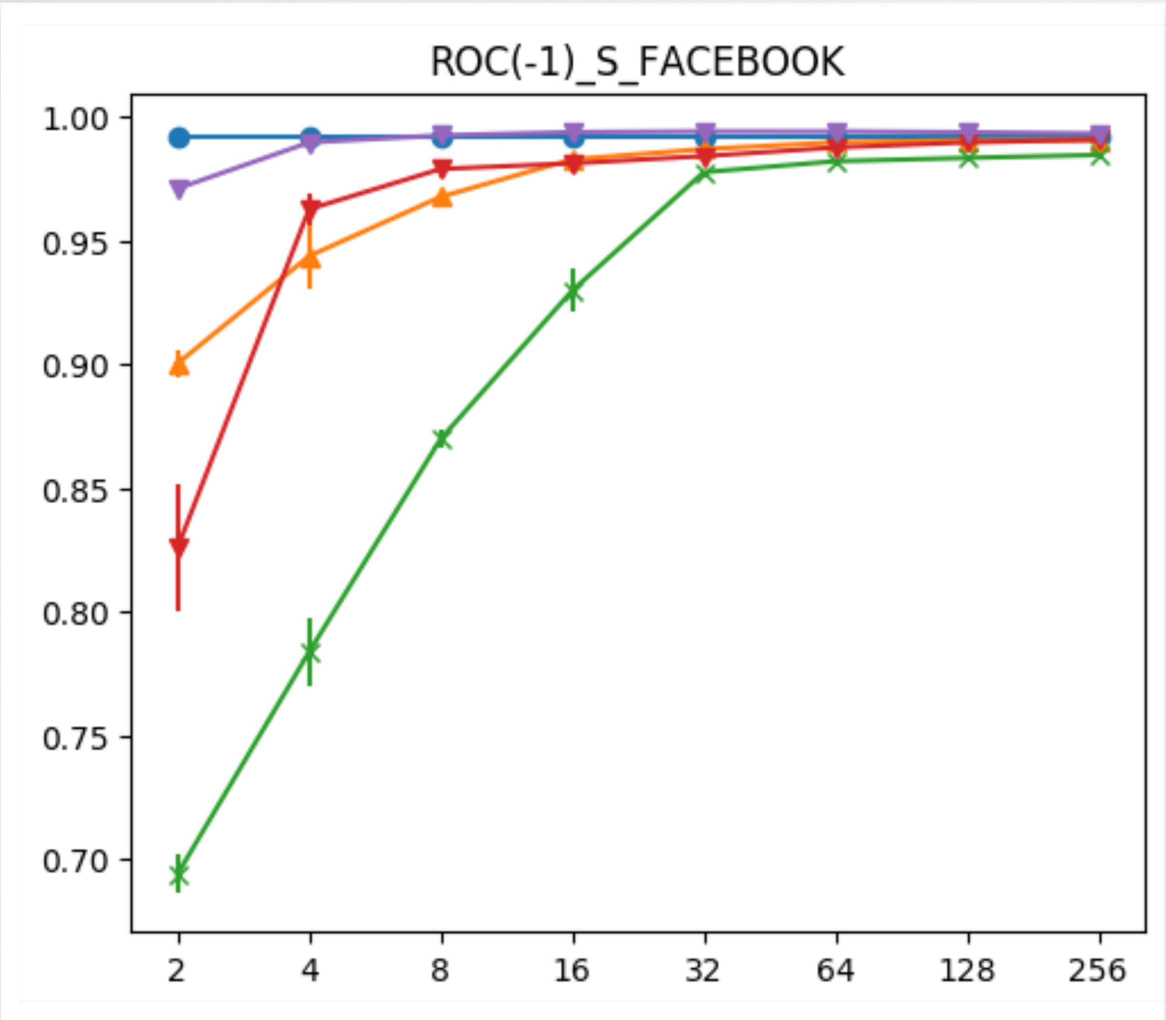
Op	Algorithm	Dataset		
		Facebook	PPI	arXiv
	Common Neighbors	0.8100	0.7142	0.8153
	Jaccard’s Coefficient	0.8880	0.7018	0.8067
	Adamic-Adar	0.8289	0.7126	0.8315
	Pref. Attachment	0.7137	0.6670	0.6996
(a)	Spectral Clustering	0.5960	0.6588	0.5812
(a)	DeepWalk	0.7238	0.6923	0.7066
(a)	LINE	0.7029	0.6330	0.6516
(a)	*node2vec*	0.7266	0.7543	0.7221
(b)	Spectral Clustering	0.6192	0.4920	0.5740
(b)	DeepWalk	**0.9680**	0.7441	0.9340
(b)	LINE	0.9490	0.7249	0.8902
(b)	*node2vec*	**0.9680**	**0.7719**	**0.9366**
(c)	Spectral Clustering	0.7200	0.6356	0.7099
(c)	DeepWalk	0.9574	0.6026	0.8282
(c)	LINE	0.9483	0.7024	0.8809
(c)	*node2vec*	0.9602	0.6292	0.8468
(d)	Spectral Clustering	0.7107	0.6026	0.6765
(d)	DeepWalk	0.9584	0.6118	0.8305
(d)	LINE	0.9460	0.7106	0.8862
(d)	*node2vec*	0.9606	0.6236	0.8477

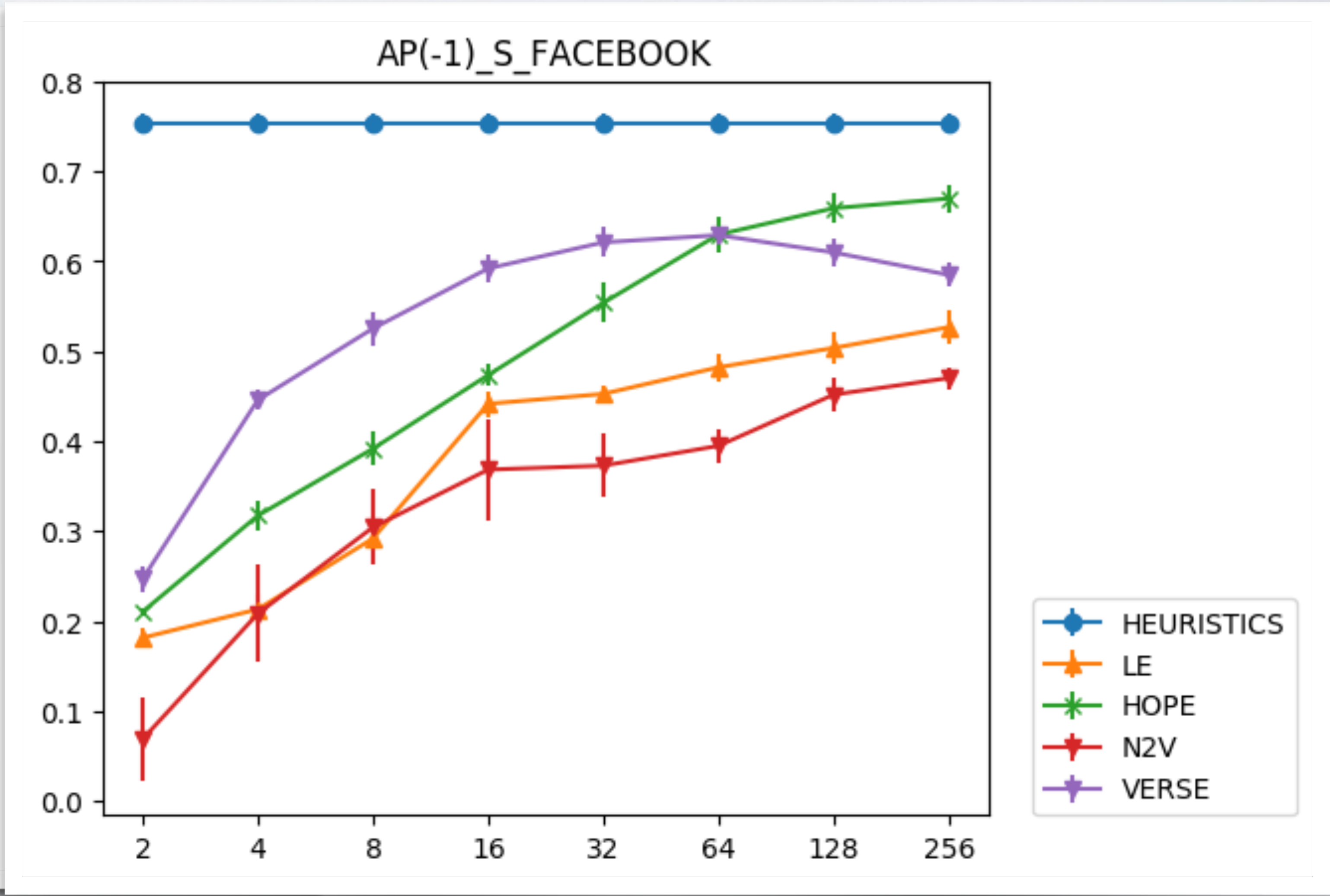
(a) Average, (b) Hadamard, (c) Weighted-L1, and (d) Weighted-L2

(AUC)

LINK PREDICTION

- Personal opinion: not that simple



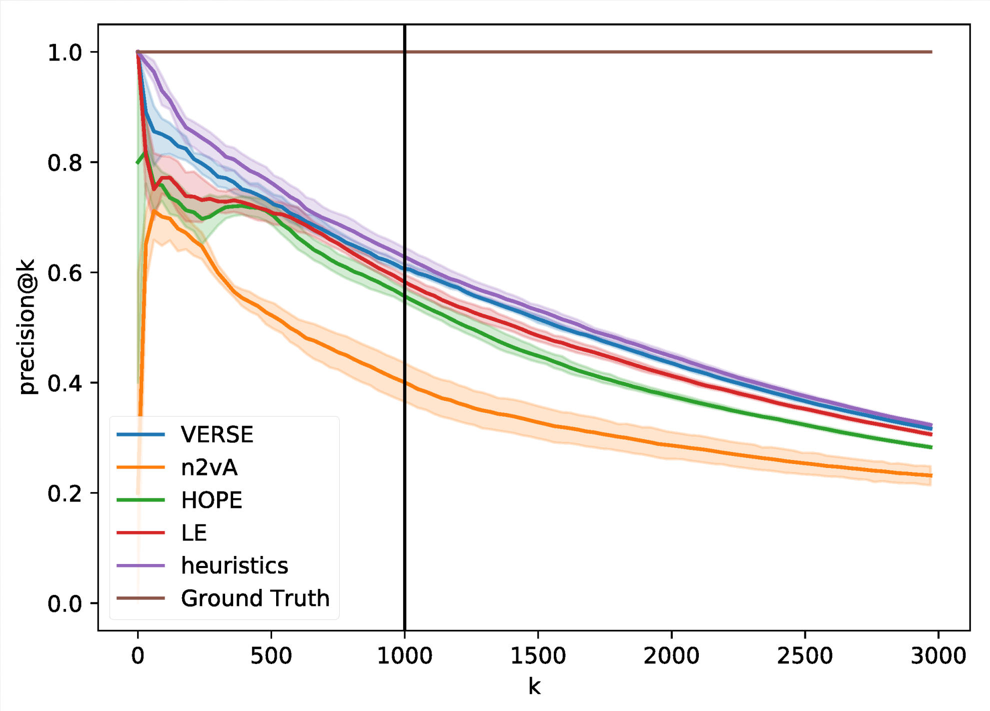


Sinha, A., Cazabet, R., & Vaudaine, R. (2018, December). Systematic Biases in Link Prediction: comparing heuristic and graph embedding based methods. In *International Conference on Complex Networks and their Applications* (pp. 81-93). Springer, Cham.

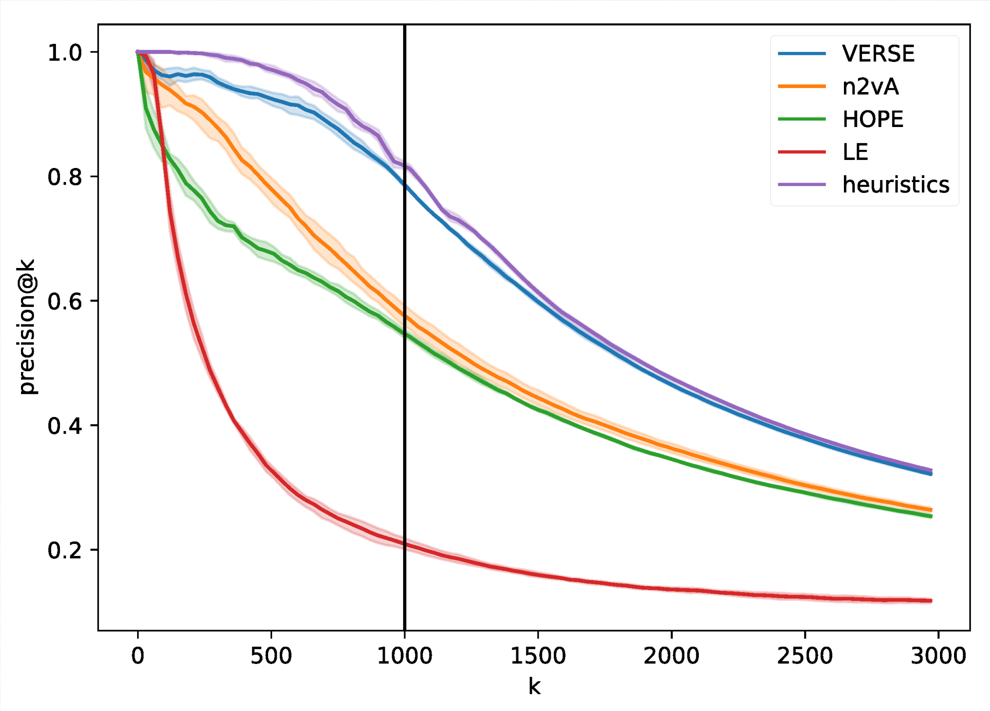
LINK PREDICTION

- First few predictions: advantage to heuristics

Better prediction at distance 2, worst otherwise



(a) FACEBOOK



(b) ASTROPH

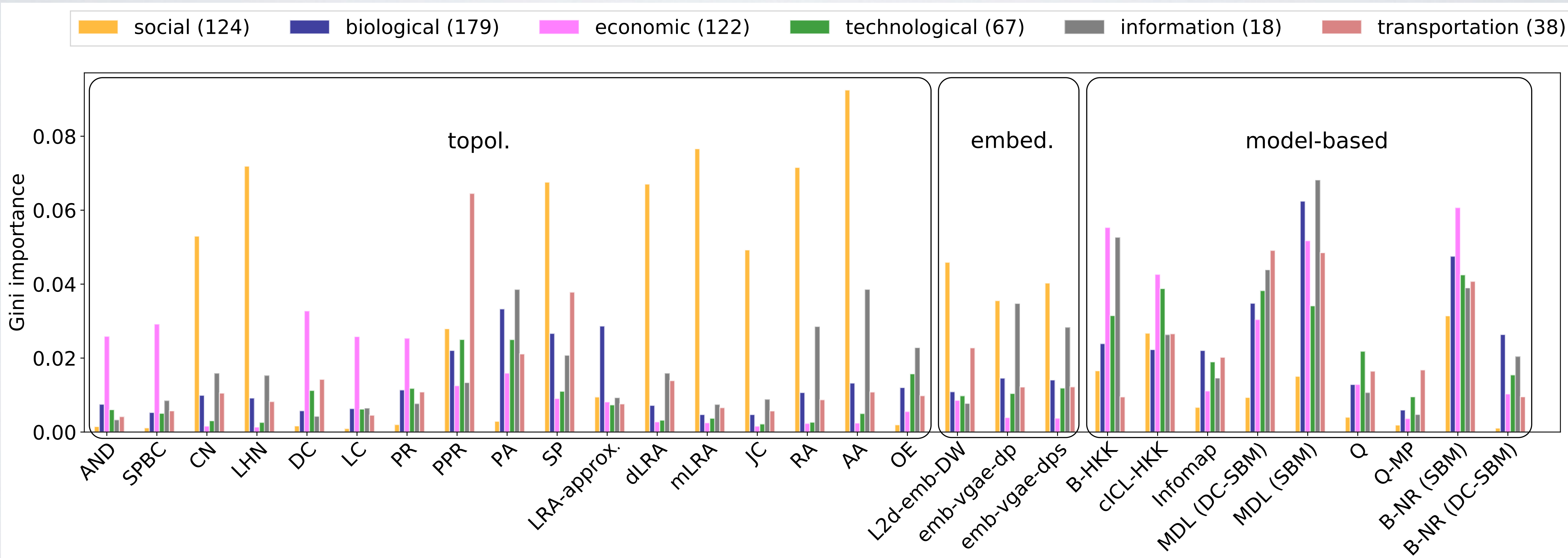
Sinha, A., Cazabet, R., & Vaudaine, R. (2018, December). Systematic Biases in Link Prediction: comparing heuristic and graph embedding based methods. In *International Conference on Complex Networks and their Applications* (pp. 81-93). Springer, Cham.

MODEL STACKING

FOR LINK PREDICTION

MODEL STACKING

Ghasemian, A., Hosseinmardi, H., Galstyan, A., Airoldi, E. M., & Clauset, A. (2020). Stacking models for nearly optimal link prediction in complex networks. *Proceedings of the National Academy of Sciences*, *117*(38), 23393-23400.



Ghasemian, A., Hosseinmardi, H., Galstyan, A., Airoldi, E. M., & Clauset, A. (2020). Stacking models for nearly optimal link prediction in complex networks. *Proceedings of the National Academy of Sciences*, *117*(38), 23393-23400.

MODEL STACKING

Table S12. Average AUC, precision, and recall performances of the link prediction algorithms over 124 social networks as a subset of CommunityFitNet corpus. A random forest is used for supervised stacking of methods. Here, the predictors are adjusted for maximum F measure using a model selection through a cross validation on training set. The results are reported on 20% holdout test set.

Algorithm	AUC	Precision	Recall
Q	0.89 ± 0.07	0.42 ± 0.13	0.85 ± 0.08
Q-MR	0.87 ± 0.07	0.38 ± 0.16	0.78 ± 0.07
Q-MP	0.86 ± 0.08	0.25 ± 0.07	0.83 ± 0.09
B-NR (SBM)	0.93 ± 0.06	0.3 ± 0.08	0.85 ± 0.12
B-NR (DC-SBM)	0.93 ± 0.07	0.28 ± 0.08	0.88 ± 0.08
cICL-HKK	0.93 ± 0.08	0.34 ± 0.1	0.85 ± 0.14
B-HKK	0.88 ± 0.07	0.17 ± 0.05	0.79 ± 0.17
Infomap	0.91 ± 0.04	0.29 ± 0.08	0.83 ± 0.05
MDL (SBM)	0.94 ± 0.07	0.31 ± 0.09	0.87 ± 0.16
MDL (DC-SBM)	0.93 ± 0.09	0.26 ± 0.09	0.89 ± 0.11
S-NB	0.94 ± 0.07	0.3 ± 0.1	0.87 ± 0.08
mean model-based	0.91 ± 0.08	0.3 ± 0.12	0.84 ± 0.12
mean indiv. topol.	0.64 ± 0.19	0.2 ± 0.27	0.56 ± 0.33
mean indiv. topol. & model	0.7 ± 0.21	0.22 ± 0.25	0.62 ± 0.32
emd-DW	0.95 ± 0.1	0.45 ± 0.16	0.92 ± 0.13
emb-vgae	0.95 ± 0.08	0.09 ± 0.02	0.96 ± 0.09
all topol.	0.97 ± 0.08	0.89 ± 0.21	0.88 ± 0.2
all model-based	0.95 ± 0.07	0.76 ± 0.2	0.68 ± 0.17
all embed.	0.95 ± 0.11	0.75 ± 0.23	0.74 ± 0.23
all topol. & model	0.98 ± 0.06	0.89 ± 0.22	0.88 ± 0.19
all topol. & embed.	0.96 ± 0.1	0.86 ± 0.22	0.83 ± 0.25
all model & embed.	0.96 ± 0.09	0.78 ± 0.21	0.74 ± 0.22
all topol., model & embed.	0.97 ± 0.09	0.86 ± 0.23	0.84 ± 0.23

Table 1. Link prediction performance (mean±std. err.), measured by AUC, precision, and recall, for link prediction algorithms applied to the 548 structurally diverse networks in our corpus.

algorithm	AUC	precision	recall
Q	0.7 ± 0.14	0.14 ± 0.17	0.67 ± 0.15
Q-MR	0.67 ± 0.15	0.12 ± 0.17	0.63 ± 0.13
Q-MP	0.64 ± 0.15	0.09 ± 0.11	0.59 ± 0.17
B-NR (SBM)	0.81 ± 0.13	0.13 ± 0.12	0.65 ± 0.22
B-NR (DC-SBM)	0.7 ± 0.2	0.12 ± 0.12	0.61 ± 0.24
cICL-HKK	0.79 ± 0.13	0.14 ± 0.14	0.58 ± 0.25
B-HKK	0.77 ± 0.13	0.11 ± 0.1	0.51 ± 0.26
Infomap	0.73 ± 0.14	0.12 ± 0.12	0.68 ± 0.13
MDL (SBM)	0.79 ± 0.15	0.14 ± 0.13	0.57 ± 0.3
MDL (DC-SBM)	0.84 ± 0.1	0.13 ± 0.11	0.78 ± 0.12
S-NB	0.71 ± 0.19	0.12 ± 0.13	0.66 ± 0.17
mean model-based	0.74 ± 0.16	0.12 ± 0.13	0.63 ± 0.21
mean indiv. topol.	0.6 ± 0.13	0.09 ± 0.16	0.53 ± 0.35
mean indiv. topol. & model	0.63 ± 0.15	0.09 ± 0.16	0.55 ± 0.33
emb-DW	0.63 ± 0.23	0.17 ± 0.19	0.42 ± 0.35
emb-vgae	0.69 ± 0.19	0.05 ± 0.05	0.69 ± 0.21
all topol.	0.86 ± 0.11	0.42 ± 0.33	0.44 ± 0.32
all model-based	0.83 ± 0.12	0.39 ± 0.34	0.3 ± 0.29
all embed.	0.77 ± 0.16	0.32 ± 0.32	0.32 ± 0.31
all topol. & model	0.87 ± 0.1	0.48 ± 0.36	0.35 ± 0.35
all topol. & embed.	0.84 ± 0.13	0.4 ± 0.34	0.39 ± 0.33
all model & embed.	0.84 ± 0.13	0.36 ± 0.32	0.36 ± 0.31
all topol., model & embed.	0.85 ± 0.14	0.42 ± 0.34	0.39 ± 0.33