MACHINE LEARNING ON GRAPHS

Link prediction, Node classification, Graph Embedding

MACHINE LEARNING

- Wikipedia:
  - Machine learning(ML) involves computers discovering how they can perform tasks without being explicitly programmed to do so
- Subset of *artificial intelligence*
- Objective of machine learning: make a program learn automatically something about your data

MACHINE LEARNING

- Supervised Machine learning:
  - Train the program with examples (properties => associated value), the program can then predict the result given input properties

- Unsupervised Machine learning:
  - Given the data, the program should find by itself its rules/organization.
  - =>Most common example: clustering.
  - =>Community detection is unsupervised machine learning on graphs

MACHINE LEARNING

- Examples of supervised machine learning
  - Given properties of an apartment, predict its energy consumption
  - Given a picture, recognize objects in it
  - Given a student profile, predict its success
  - Given a criminal profile, predict its probability of recidivism
  - Given past values and collected news, predict market fluctuations
  - Given a patient profile, predict effect of a drug
  - Given a fingerprint/face, recognize the user
  - …

MACHINE LEARNING

Difference between linear regression and Advanced machine learning ?

("Black box" models, random forest, deep neural networks)

What about multicolinearity, heteroscedasticity ?

SUPERVISED MACHINE LEARNING I: LINK PREDICTION

LINK PREDICTION

- Do you know why Facebook "People you may know" is so accurate?
- How youtube/Spotify/amazon recommend you the right item?
- =>Link prediction
  - More generally, recommendation, but link prediction is a popular way to do it

LINK PREDICTION

- Observed network: current state

- Link prediction: What edge
  - Might appear in the future (*future link* prediction)
  - Might have been missed (*missing link* prediction)

LINK PREDICTION

- Overview:
- Link prediction based on network structure:
  - Local: High clustering (friends of my friends will become my friends)
  - Global: Two unrelated hubs more likely to have links that unrelated small nodes
  - Meso-scale organisation: different edge probability for nodes in different communities/blocks
- Link prediction can also be based on node properties
  - e.g., age, revenue, genre, etc.
  - Combining with usual machine learning, outside of the scope of this course

FIRST APPROACH TO LINK PREDICTION:

HEURISTIC BASED

(HEURISTICS, NOT MACHINE LEARNING)

HEURISTICS

- Network science experts can design **heuristics** to predict where new edge might appear/be missing

- Principle: design a score based on network topology f(v1,v2) which, given two nodes, express their likeliness of being connected (if they aren't already)
  - Common neighbors
  - Jaccard coefficient
  - Hub promoted
  - Adamic Adar
  - Ressource allocation
  - Community based

Zhou, T., Lü, L., & Zhang, Y. C. (2009). Predicting missing links via local information. *The European Physical Journal B*, *71*(4), 623-630.

COMMON NEIGHBORS

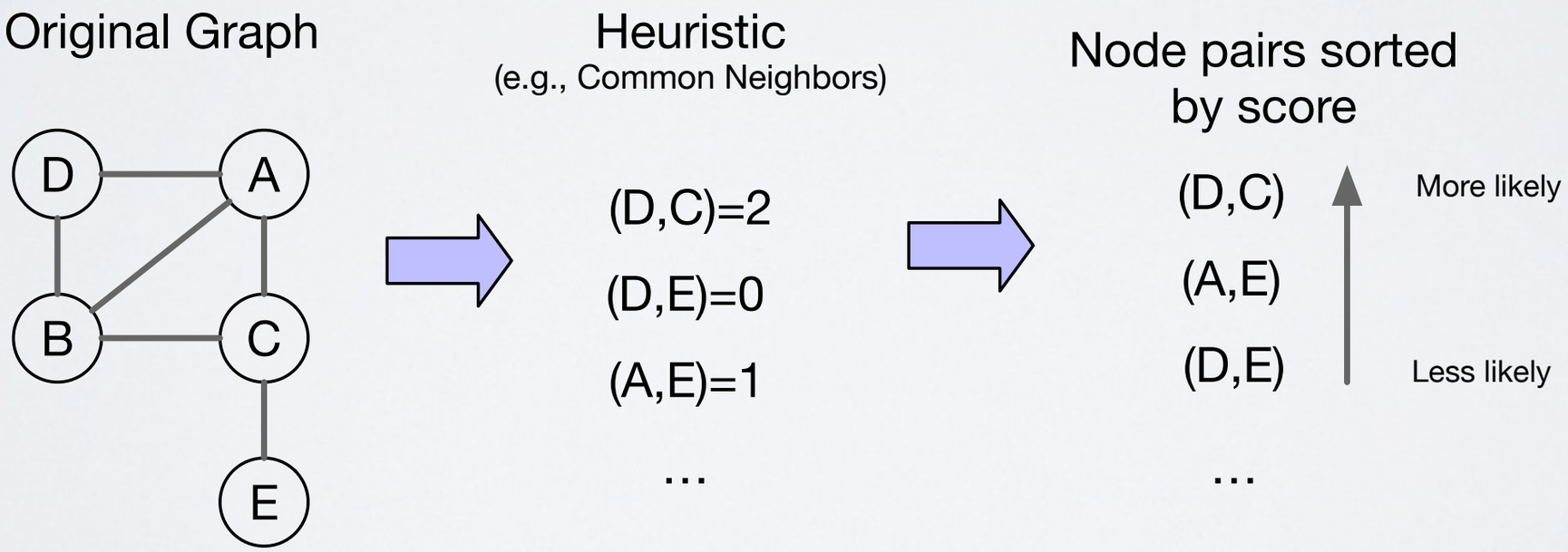
- "Friends of my friends are my friends"
- High clustering in most networks
- =>The more friends in common, the highest probability to become friends

$$\mathrm{CN}(x, y) = |\Gamma(x) \cap \Gamma(y)|$$

$\Gamma(x) =$ Neighbors of x

PREDICTION

- How to predict links based on Common Neighbors (CN)?



JACCARD COEFFICIENT

- Used in many applications:
  - Measure of similarity of sets of different sizes
- Intuition:
  - Two people who know only the same 3 people
    - =>high probability
  - Two people who know 1000 people, only 3 in commons
    - =>Lower probability

$$\mathrm{JC}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}$$

ADAMIC ADAR

- Intuition:
  - For previous scores: all common nodes are worth the same
  - For AA:
    - A common node with ONLY them in common is worth the most
    - A common node connected to everyone is worth the less
    - The higher the size of its neighborhood, the lesser its value

$$\mathrm{AA}(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log|\Gamma(z)|}$$

PREFERENTIAL ATTACHMENT

- Preferential attachment:
  - Every time a node join the network, it creates a link with nodes with probability proportional to their degrees
  - In fact, closer to the definition of the configuration model

- Score not based on common neighbors
  - =>Assign different scores to nodes at network distance >2

- Intuition: Two nodes with many neighbors more likely to have new ones than nodes with few neighbors

$$\mathrm{PA}(x, y) = |\Gamma(x)| \cdot |\Gamma(y)|$$

COMMUNITY STRUCTURE

- General idea:
  - 1)Compute community structure on the whole graph
  - 2)Assign high score for 2 nodes in a same community, a low score otherwise

- How to choose the score?

ML APPROACH TO LINK PREDICTION:

SIMILARITY SCORE, SUPERVISED

SUPERVISED MACHINE LEARNING

- Use Machine Learning algorithms to **learn** how to combine heuristics for optimizing predictions

- Two steps:
  - Training: show features + value to predict
  - Using/Validating: try to predict value from features

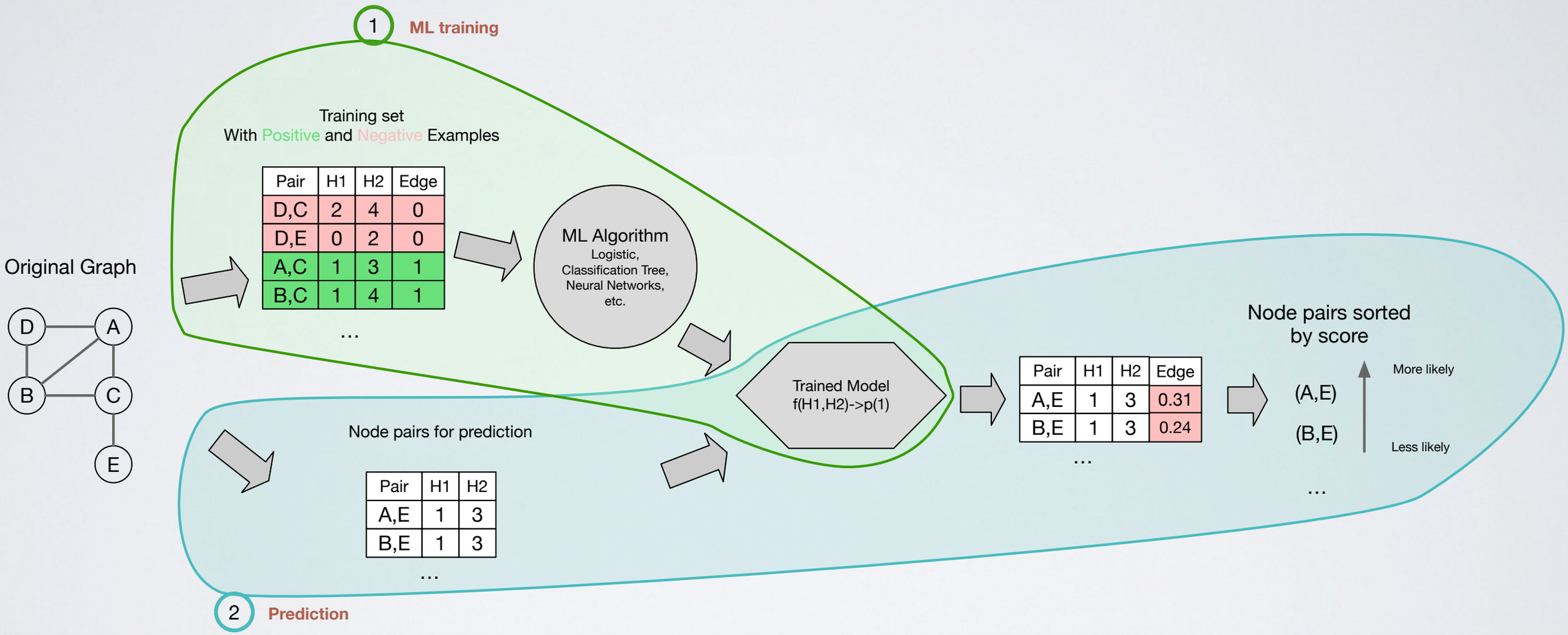
SUPERVISED MACHINE LEARNING

- Our features: similarity indices (CN, AA, PA, …)
  - Nodes attributes can be added if available (age, salary, etc.) (pairs, average…)

- Our label/value to predict: *Link(1)* or *No link(0)* (2 **classes**)

- These types of ML algorithms are called **classifiers**
  - Logistic Classifier
  - Decision Tree Classifier
  - Neural networks Classifier
  - …

SUPERVISED MACHINE LEARNING



SUPERVISED MACHINE LEARNING

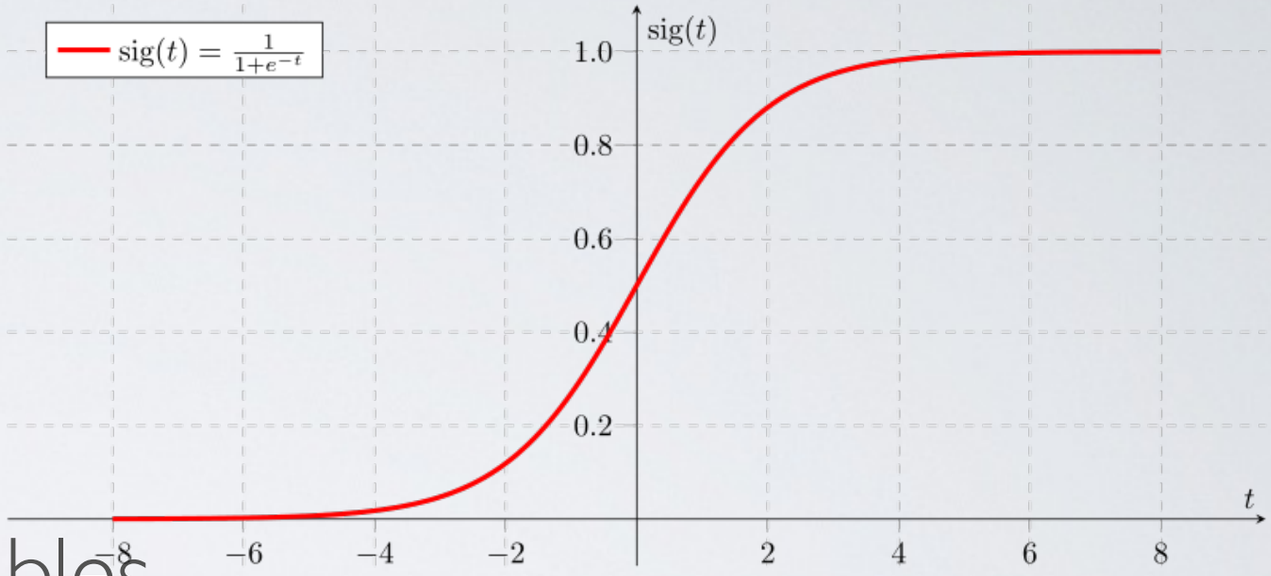
- Dozens of methods, very different in their mechanisms, but same **input** and **output**

```
#lm = linear_model.LinearRegression()
#lm = linear_model.ElasticNet()
#lm = ensemble.GradientBoostingRegressor()
#lm = ensemble.RandomForestRegressor()
lm = MLPRegressor(hidden_layer_sizes=(3,3,3)



lm.fit(X_train,y_train)
```

Let's see 2 simple examples: Logistic classification, Decision Trees

https://en.wikipedia.org/wiki/Logistic_regression

LOGISTIC CLASSIFICATION



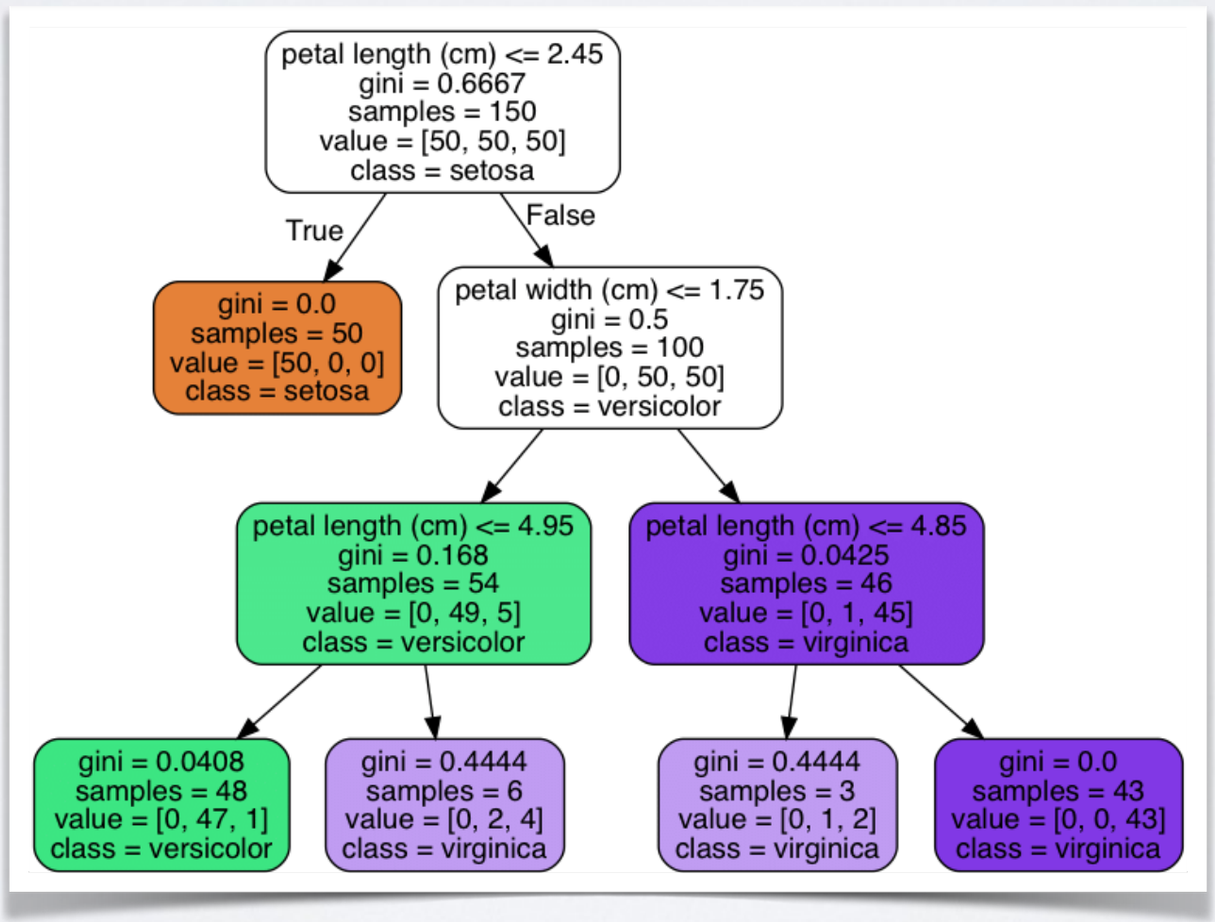
- Value to predict y_t:
  - 0 (no edge)
  - 1 (edge)

- Linear relations between variables
  - $y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i$

- Find $\beta_0, \beta_1, \ldots$ that minimizes $y_t - y_i$

DECISION TREES

- Measure of heterogeneity (Gini, entropy…)
- Split recursively data in 2 to maximize homogeneity in child nodes



https://en.wikipedia.org/wiki/Decision_tree

LINK PREDICTION EVALUATION

EVALUATION

- In order to choose a method for link prediction, it is needed to evaluate the quality of the prediction

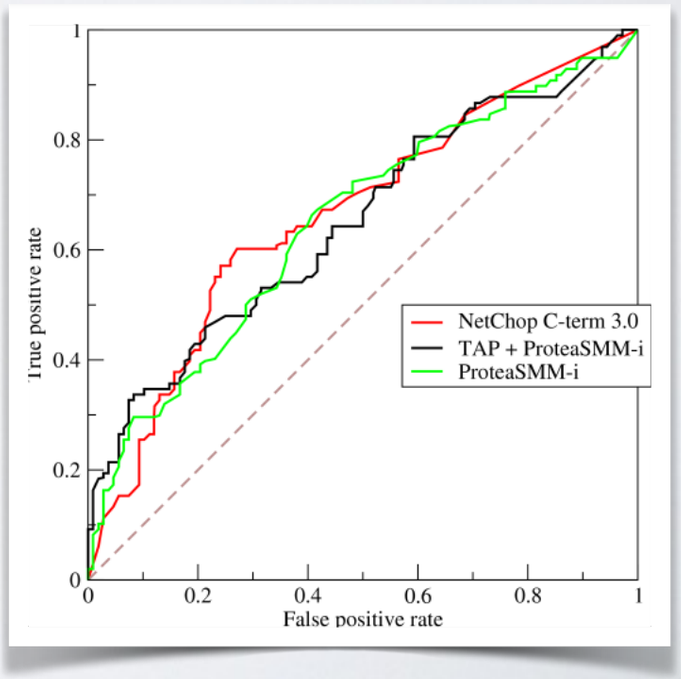
- Several measures of prediction quality exists, but all takes the same inputs:
  - A set of test examples, and for each of them:
    - The ground truth value to predict (edge/not-edge)
    - The score provided by the prediction algorithm
  - We introduce two scores:
    - Precision @k
    - Area Under the Receiver Operating Characteristic Curve (AUROC, usually only AUC)

PRECISION @K

- Simple approach : Precision @k
- Fraction of correct prediction among k pairs of highest score
- Problem: which value of k to choose?
  - Affects strongly the score
  - =>Solution: a value combining scores of any k

AUC - AUROC

- AUC: Area Under the Curve. Short (erroneous) name for AUROC (Area under the Receiver Operating Characteristic Curve)
- Analyze the relationship between
  - False positives rate
  - True positives rate
- Take the area under the curve



AUC - AUROC

- Probabilistic interpretation:
  - If we pick a random positive example and a random negative example, probability that the positive one has a higher score

- Pros:
  - Independent on the fraction of positive examples, i.e., a balanced dataset can be used

- Cons:
  - Often very high values, (>0.95), thus small relative improvements

NODE CLASSIFICATION

Bhagat, S., Cormode, G., & Muthukrishnan, S. (2011). Node classification in social networks. In *Social network data analytics* (pp. 115-148). Springer, Boston, MA.

NODE CLASSIFICATION

- For the node classification task, we want to predict the class/ category (or numerical value) of some nodes
  - Missing values in a dataset
  - Learn to predict, in a social network/platform(Netflix…) individuals':
    - Political position, opinion on a given topic, possible security threat, …
    - Interests, tastes, etc.
    - Age, genre, sexual orientation, language spoken, salary, etc.
    - Fake accounts, spammers, bots, malicious accounts, etc.
    - …
  - Wikipedia article category, types of road in an urban network, etc.

NODE FEATURES

- Non-network approach: Use a classification algorithm based on features of the node itself (age, salary, etc.)
- The network structure can be integrated using node centralities: Degree, clustering coefficient, betweenness, etc.
- But we can do much better:
  - "Tell me who your friends are, and I will tell you who you are"

NEIGHBORHOOD BASED CLASSIFICATION

- Classification based on the distribution of features in the neighborhood

- For each node, compute the distribution of labels in its neighborhood (vectors of length *m*, with *m* the set of all possible labels)
  - Pick the most frequent
    - e.g., political opinions
  - Train a classifier on this distribution
    - e.g., distribution of age, language in the neighborhoods to recognize bots (unexpectedly random)

EXAMPLE: BITCOIN USER CATEGORY PREDICTION

- Suppose we have clusters of addresses (actors)
  - For some, we know the category (exchange, mining pool, mixer, ransomware, etc.)
  - For others, we don't
- What do you propose to predict the class of unknown actors ?

GRAPH/NODE EMBEDDING

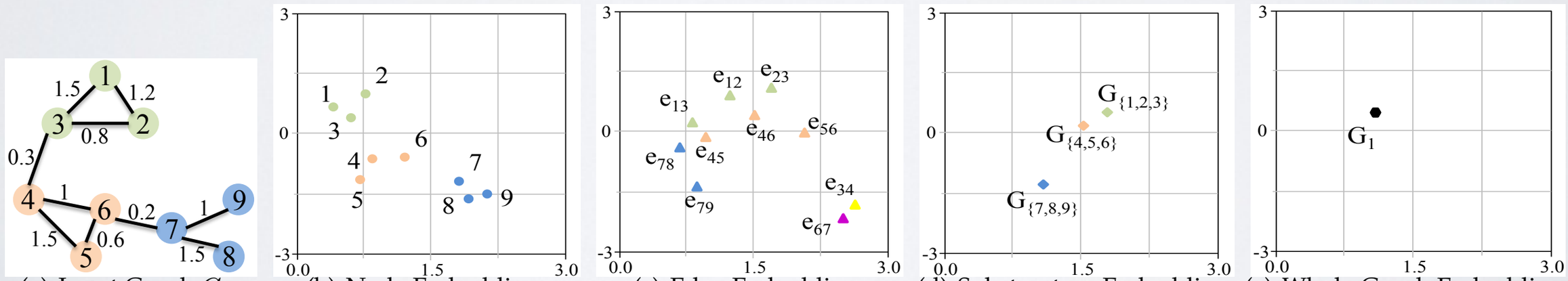
Goyal, P., & Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, *151*, 78-94.

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)



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.

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.

RANDOM WALKS BASED

DEEPWALK

- The first *Random Walk+Neural Networks* graph embedding method.
  - First of a long series

- Adaptation of **word2vec/skipgram** to graphs

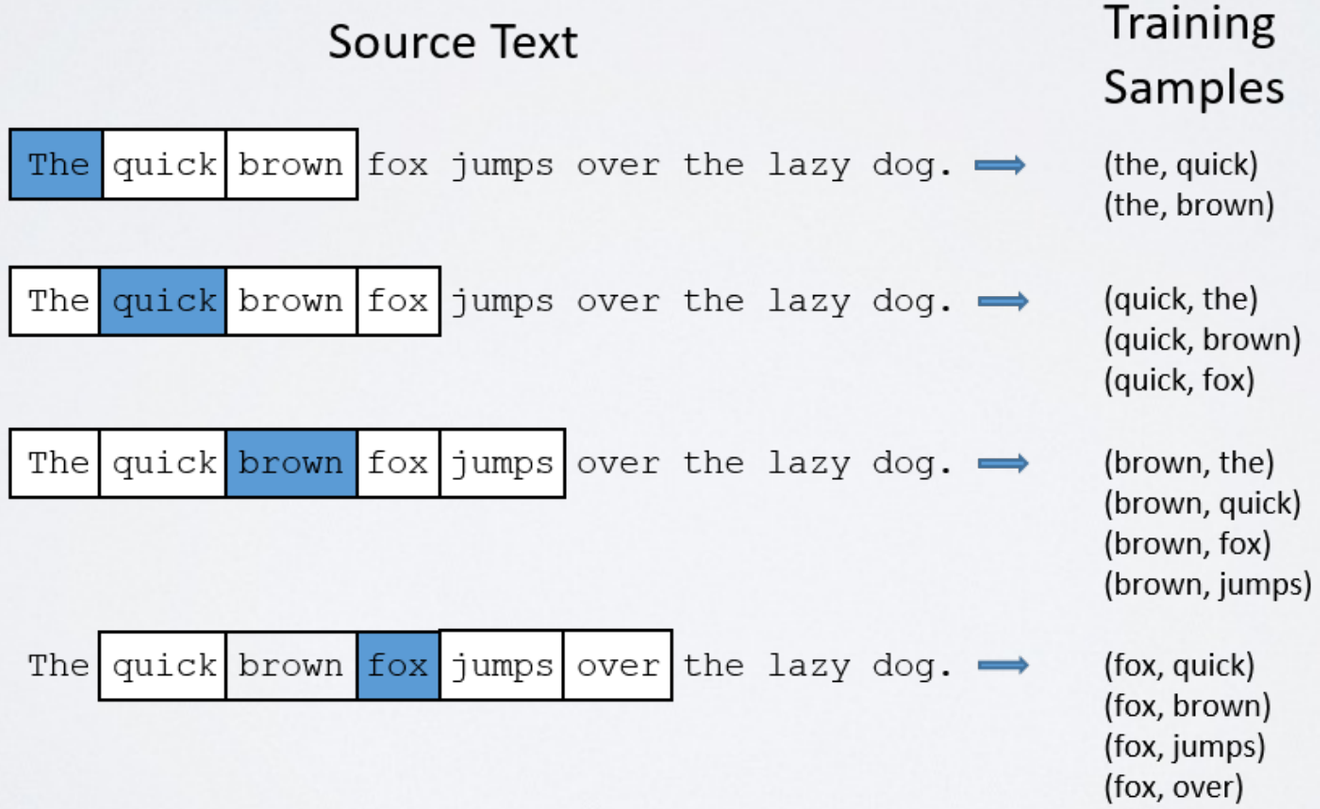
Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710). ACM.

SKIPGRAM

Word embedding

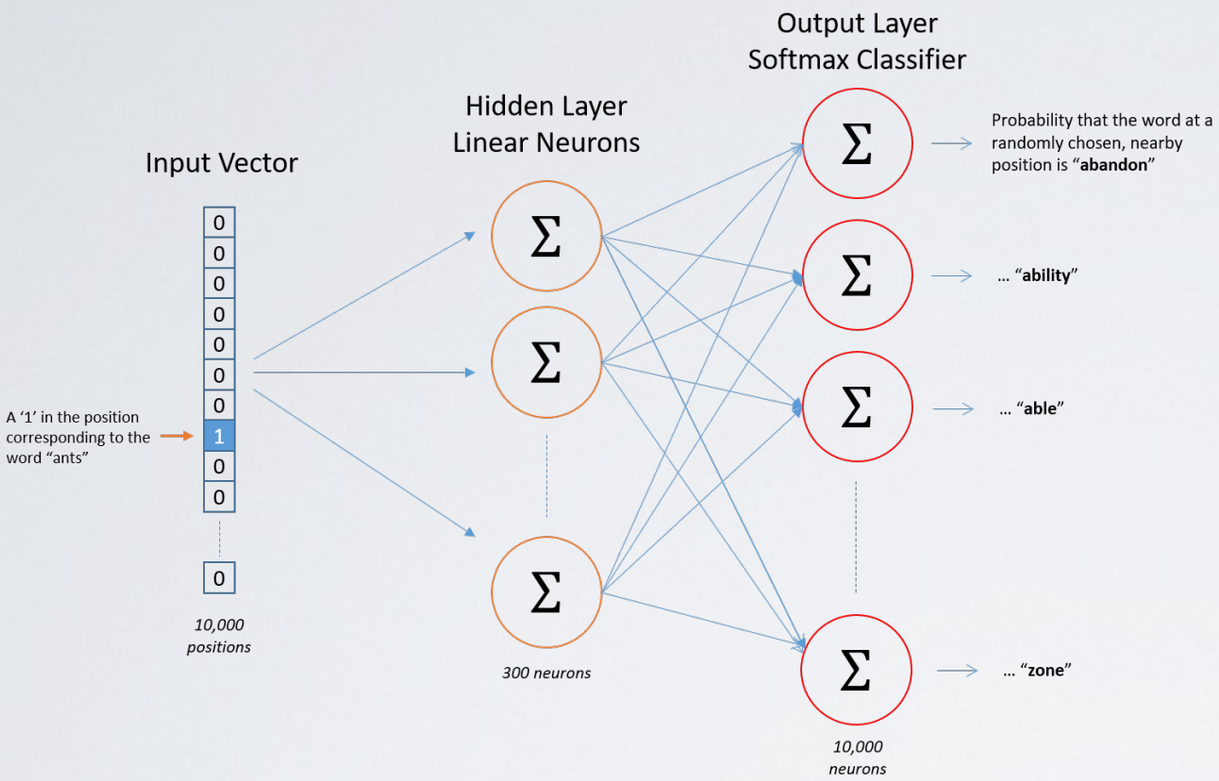
Corpus => Word = vectors

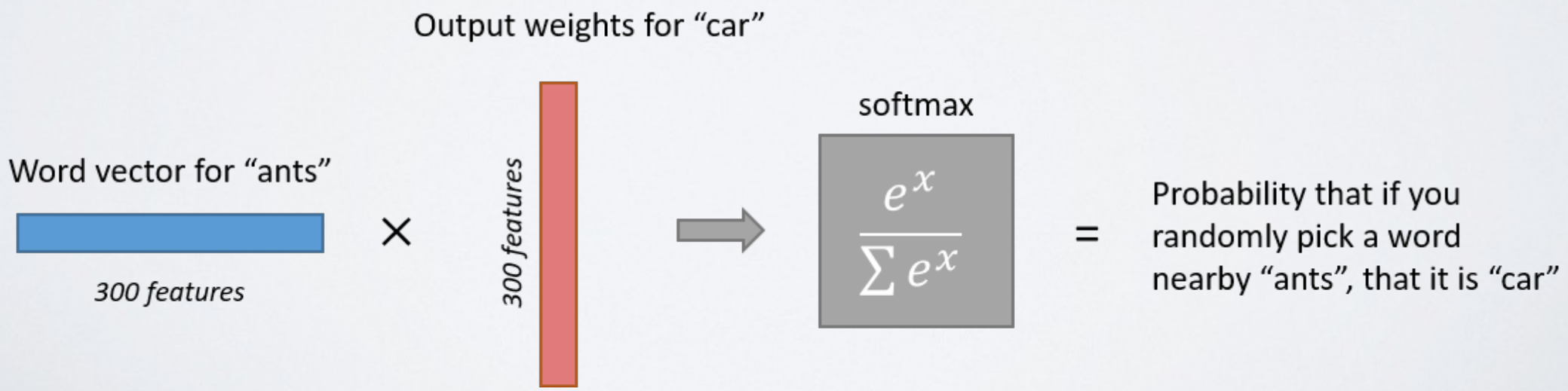
Similar embedding= similar **context**



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

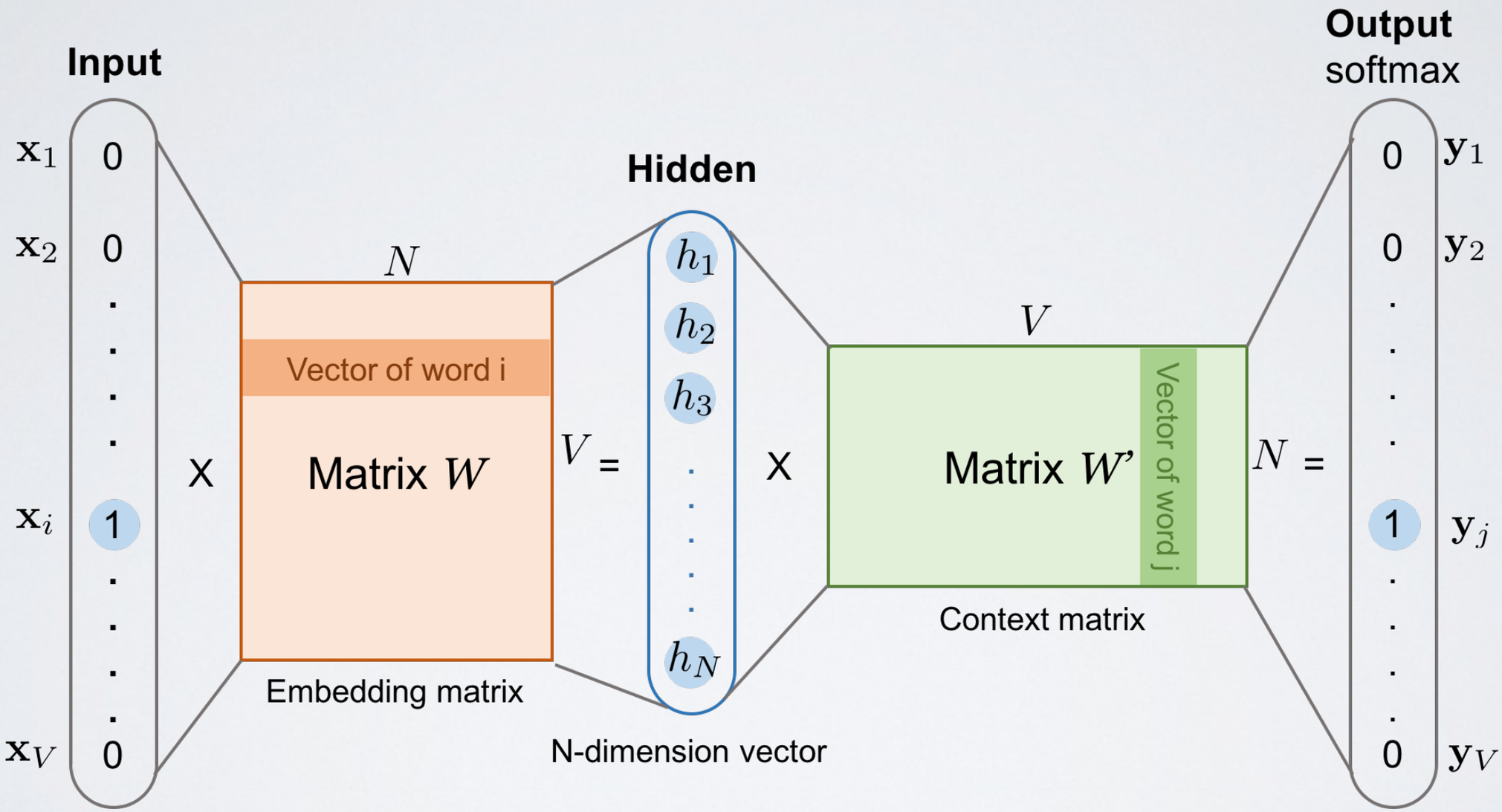
SKIPGRAM





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

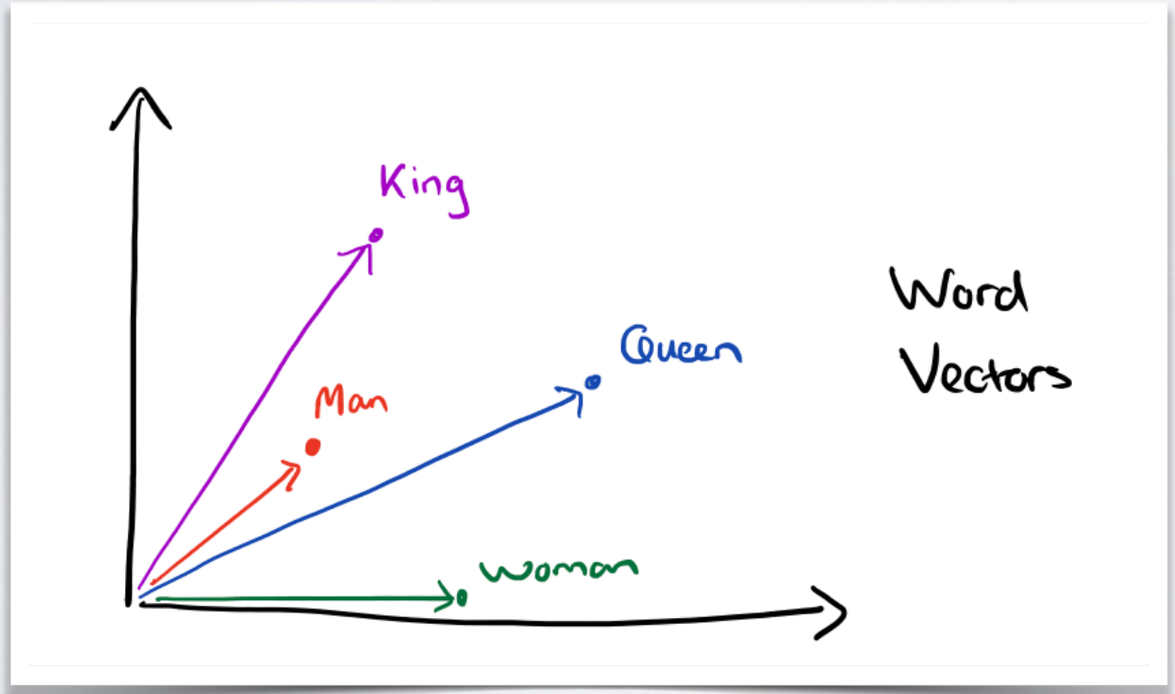
SKIPGRAM

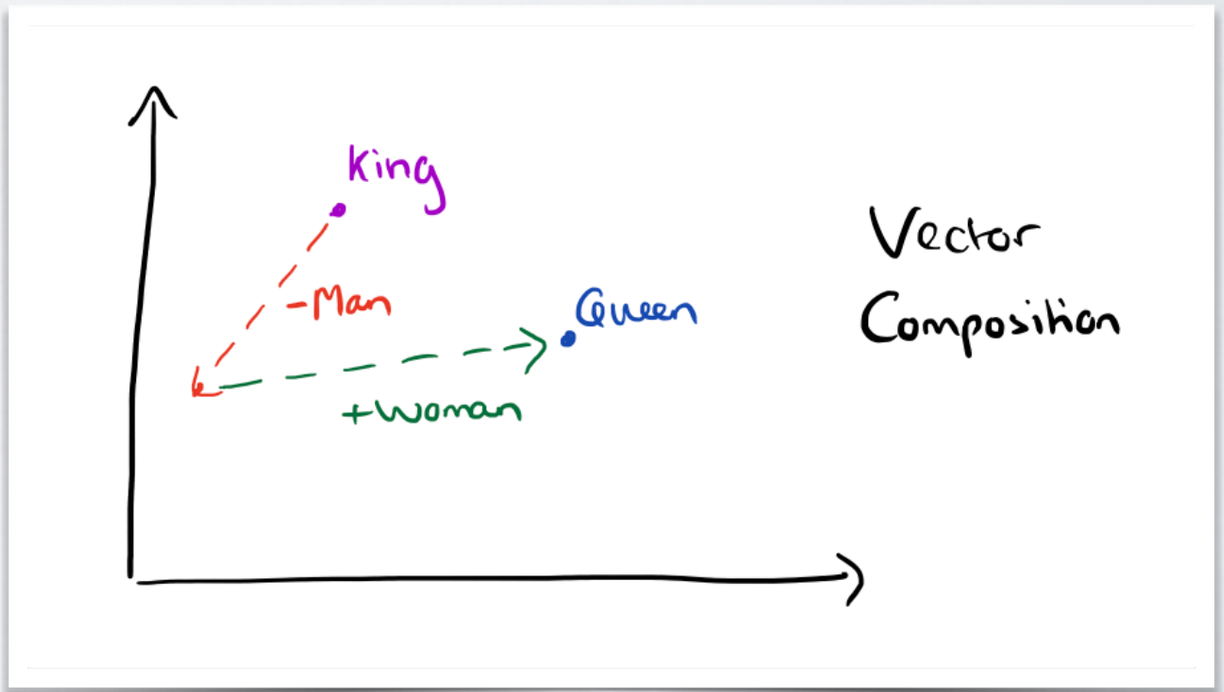


N=embedding size. **V**=vocabulary size

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

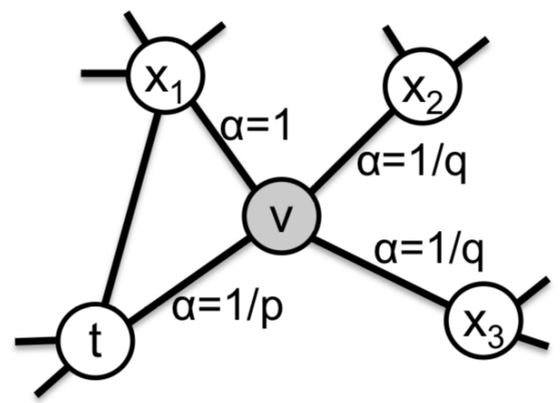


Figure 2: Illustration of the random walk procedure in *node2vec*. The walk just transitioned from t to v and is now evaluating its next step out of node v. Edge labels indicate search biases α.

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.

EMBEDDING ROLES

STRUC2VEC/ROLE2VEC

- In node2vec/Deepwalk, the context collected by RW contains the **labels** of encountered nodes

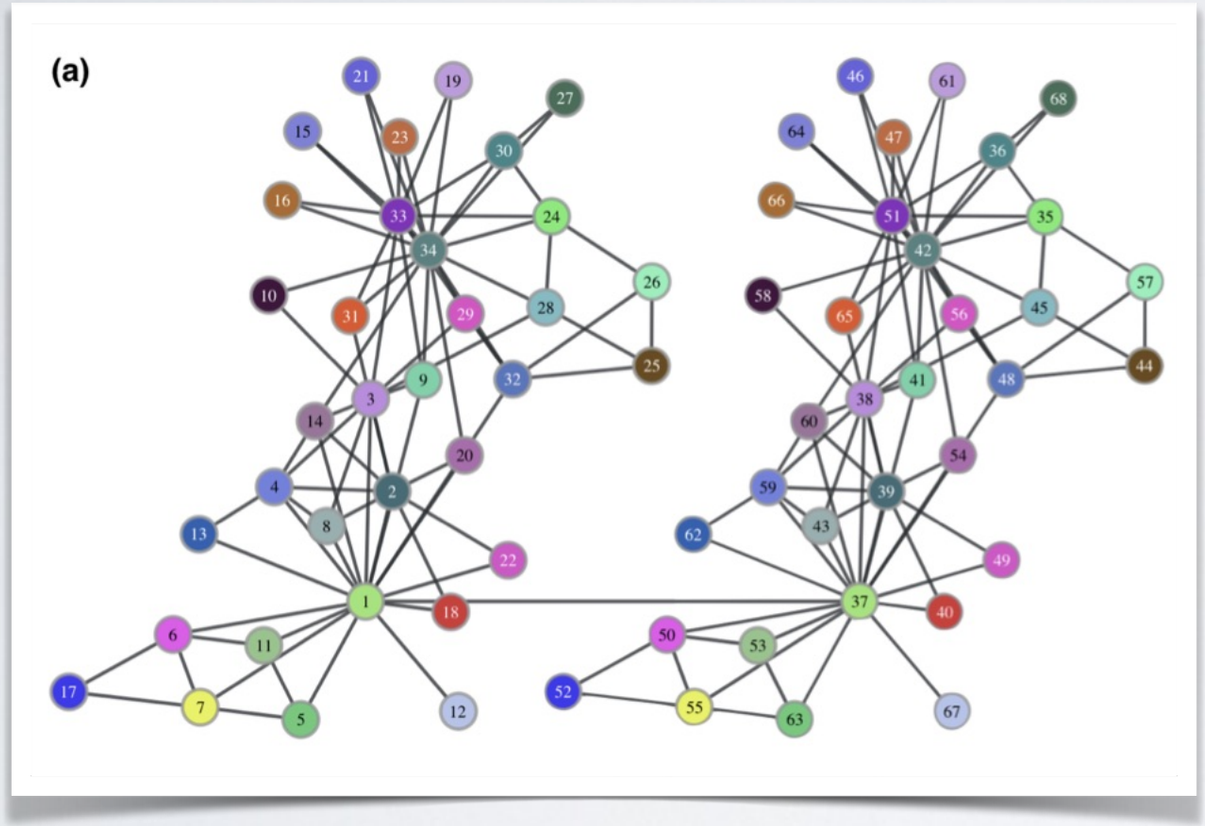
- Instead, we could memorize the **properties** of the nodes: attributes if available, or computed attributes (degrees, CC, …)

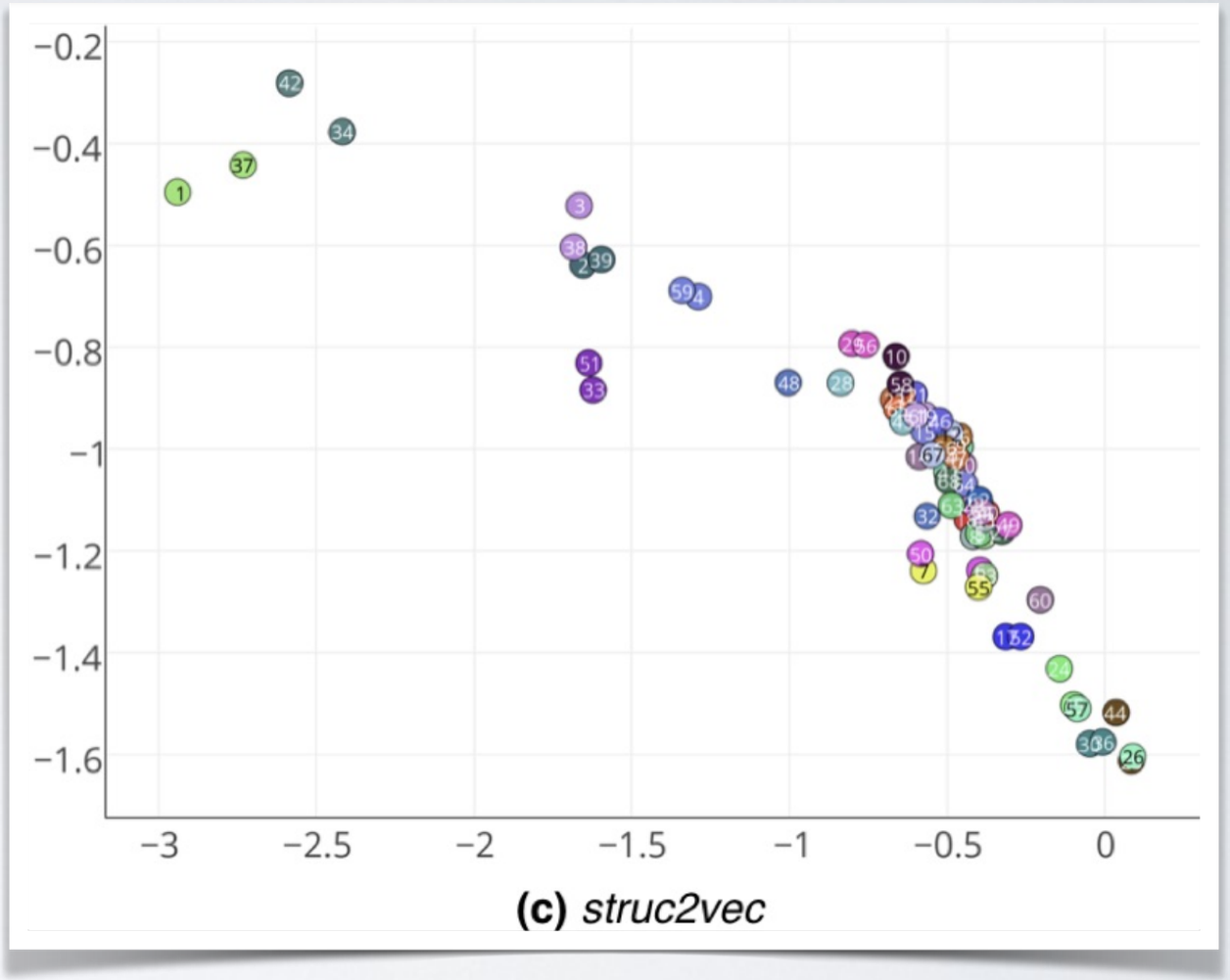
- =>Nodes with a same context will be nodes in a same "position" in the graph

- =>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





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.