Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería de Transporte y Logística

Sistemas Urbanos Inteligentes

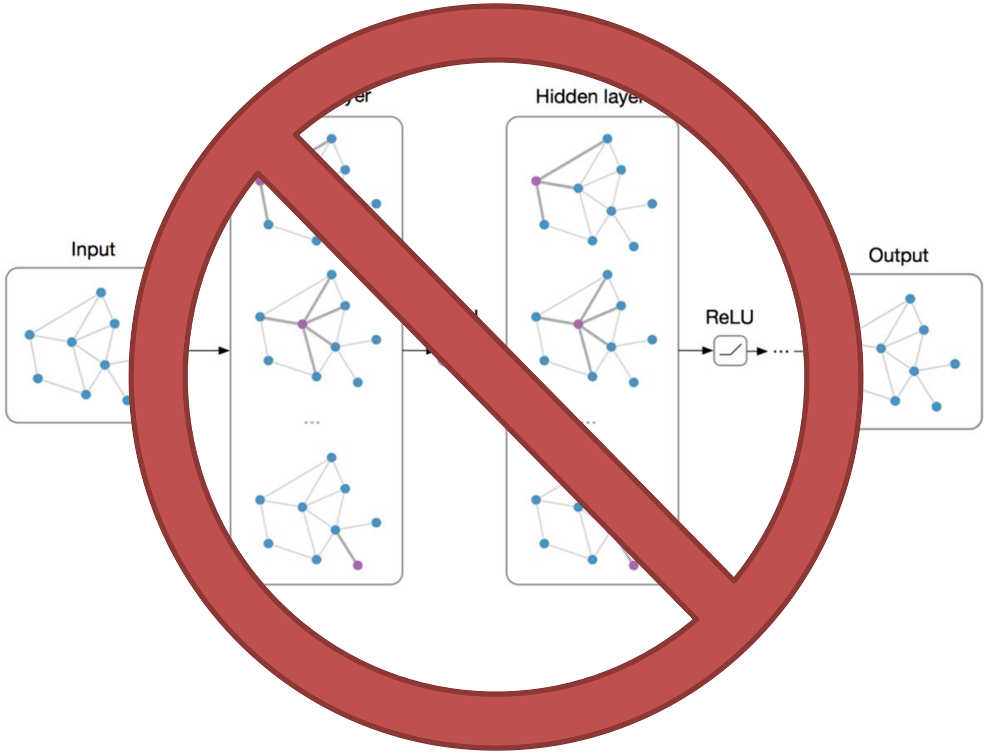
Node Embeddings

Hans Löbel
Dpto. Ingeniería de Transporte y Logística
Dpto. Ciencia de la Computación

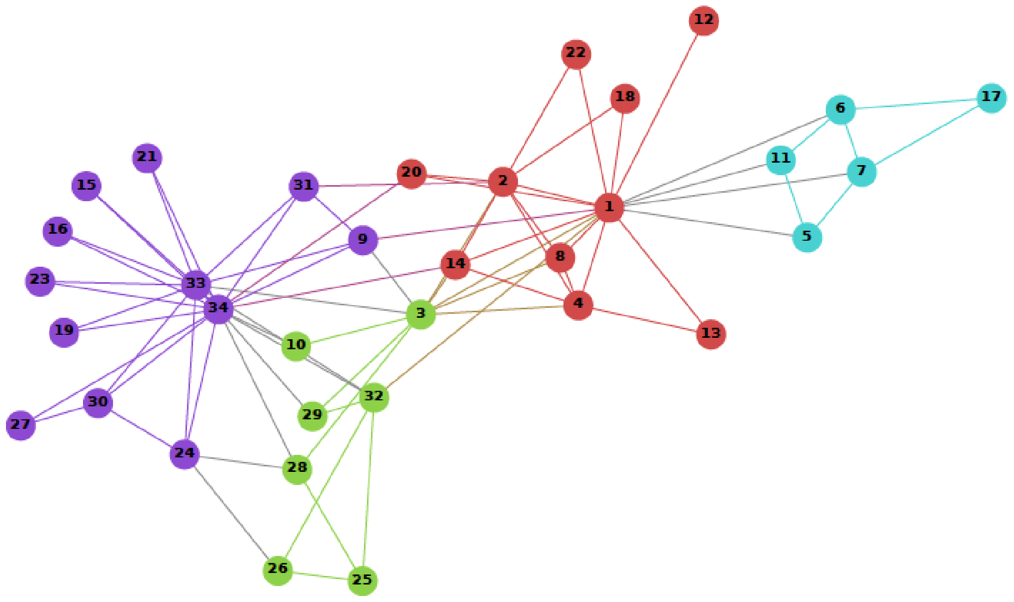
Muchas veces, entrenar una red no es lo que buscamos

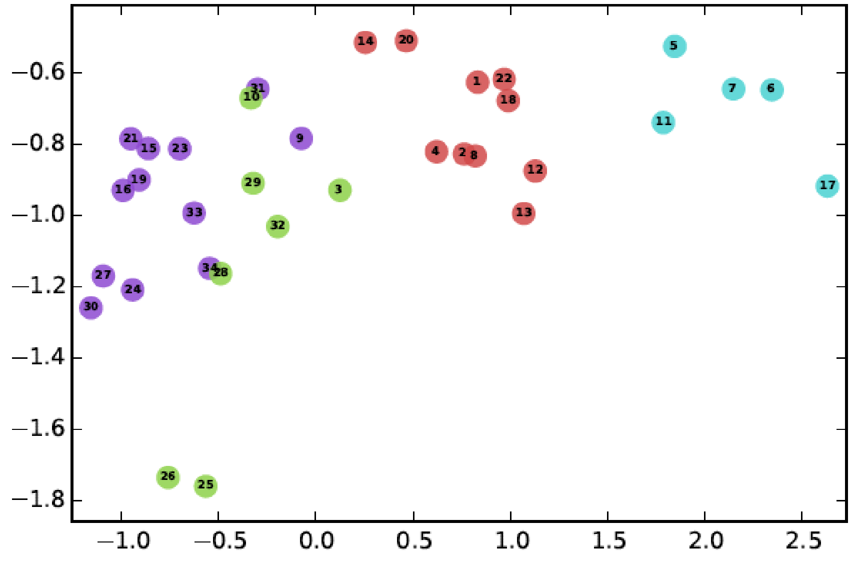
(no le digan a nadie que dije eso)

- Puede que no tengamos una tarea asociada al grafo.
- Quizá los nodos del grafo son un insumo para otro proceso.
- Recursos computacionales pueden ser demasiado altos



Sin embargo, los conceptos de ML igual tienen validez, en particular, el aprendizaje de representaciones

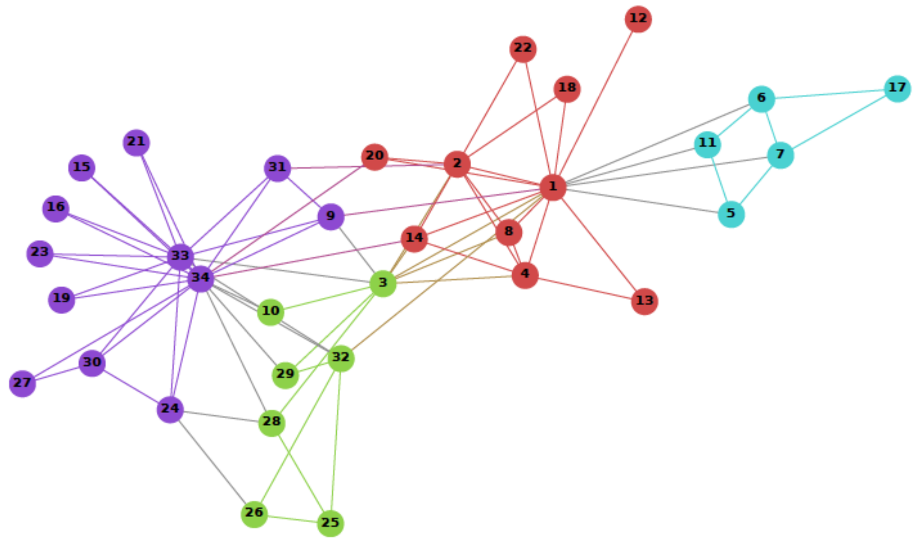




Por ejemplo, podemos interesarnos en un espacio de *embedding*, donde nodos "similares" se encuentren cerca.

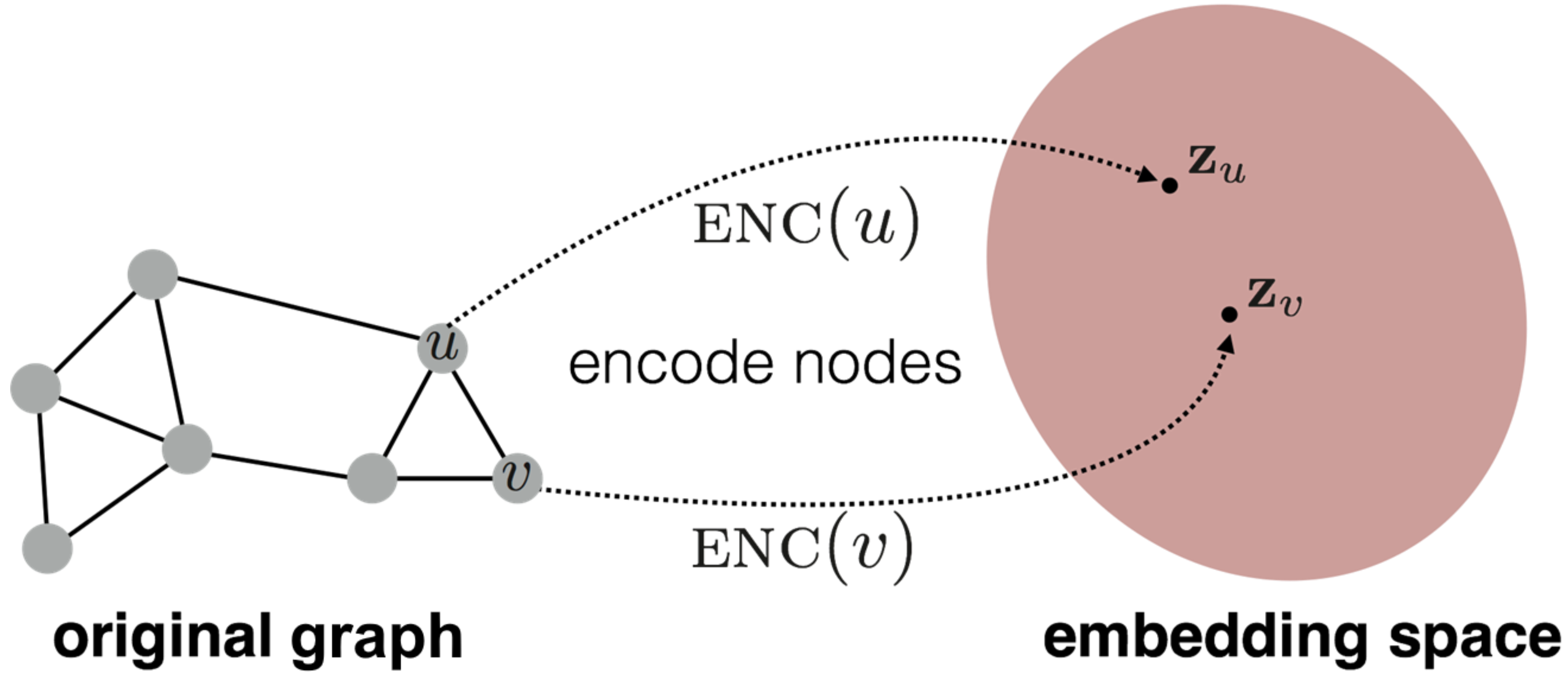
Comencemos formalizando esta idea

- Un grafo G:
- V es el conjunto de nodos
- A es la matriz de adyacencia
- Los nodos no tiene features

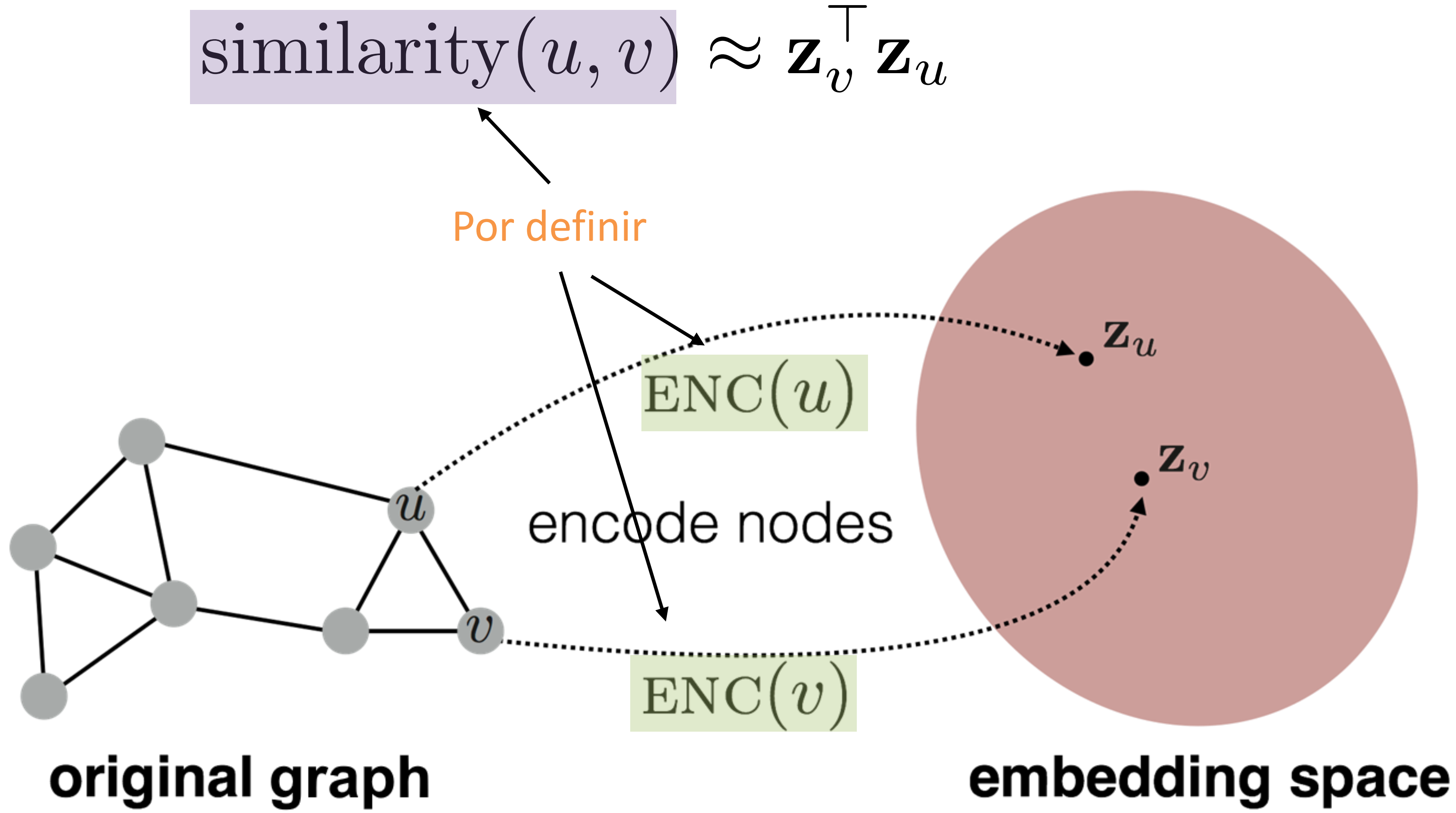


Comencemos formalizando esta idea

El objetivo es codificar nodos, de forma que la similitud en el espacio de *embedding* (medida como el producto punto) aproxima la similitud en el grafo original.



Node embeddings



Para aprender estos *embeddings*, seguimos el mismo enfoque que hemos usado todo el curso

1. Definimos una función codificadora (*encoder*) de nodos a embeddings
2. Definimos una función de similitud entre nodos en el grafo
3. Optimizamos los parámetros del *encoder*, de forma de obtener lo siguiente:

$$\text{similarity}(u, v) \approx \mathbf{z}_v^\top \mathbf{z}_u$$

Necesitamos entonces definir estos dos elementos

- Encoder mapea cada nodo a un vector de baja dimensionalidad:

embedding d-dimensional

$$\mathrm{ENC}(v) = \mathbf{z}_v$$

Nodo del grafo

- La función de similitud especifica como se mapean las relaciones del grafo original al espacio de embedding:

$$\mathrm{similarity}(u, v) \approx \mathbf{z}_v^{\top} \mathbf{z}_u$$

similitud entre *u* y *v* en el grafo

producto punto en espacio de embedding

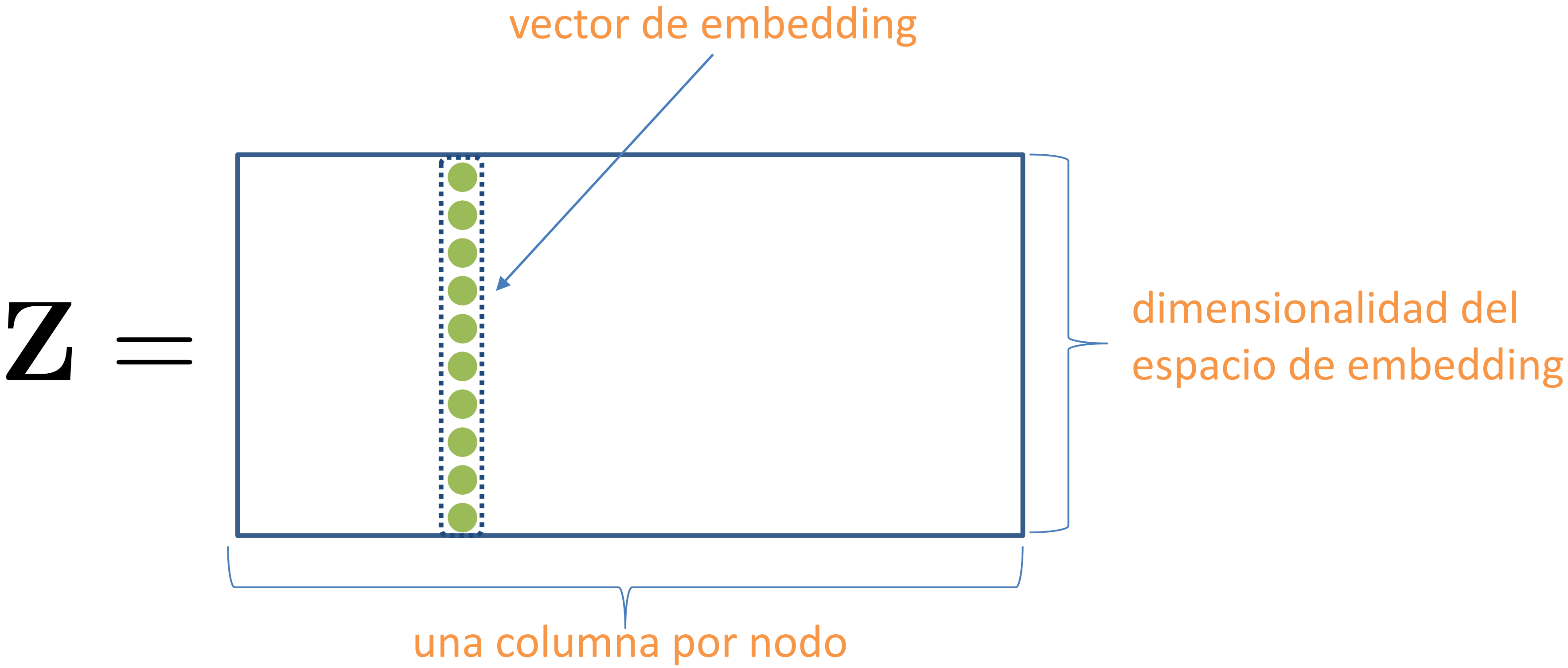
Tal como nuestros *embeddings* anteriores, el encoder será simplemente la indexación de un vector en una matriz

$$\mathrm{ENC}(v) = \mathbf{Z}\mathbf{v}$$

$\mathbf{Z} \in \mathbb{R}^{d\times|\mathcal{V}|}$ Matriz de embedding

$\mathbf{v} \in \mathbb{I}^{|\mathcal{V}|}$ vector one-hot que indica el índice del nodo

Tal como nuestros *embeddings* anteriores, el encoder será simplemente la indexación de un vector en una matriz



La función de similitud es más compleja

- ¿Cuándo deben considerarse como similares dos nodos?
  - ¿Si están conectados?
  - ¿Si comparte vecinos
  - ¿Si tienen funciones estructurales similares?
  - ¿…?

- En general, existen tres ideas principales:
  1. Basada en adyacencia
  2. Basada en múltiples *hops*
  3. Random walks

Hamilton et al. 2017, Representation Learning on Graphs: Methods and Applications. *IEEE Data Engineering Bulletin on Graph Systems*.

En la basada en adyacencia, la similitud es simplemente el peso del arco entre los nodos

$$\boxed{\mathcal{L}} = \boxed{\sum_{(u,v)\in V\times V}} \|\boxed{\mathbf{z}_u^\top \mathbf{z}_v} - \boxed{\mathbf{A}}_{u,v}\|^2$$

Función de pérdida

Suma sobre todos los pares de nodos

producto punto

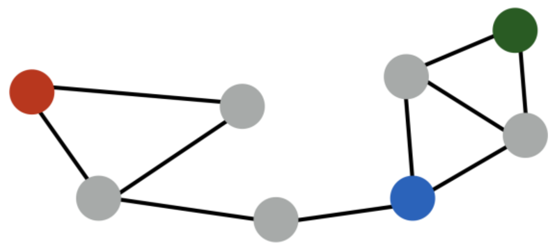
Matriz de adyacencia con pesos

En la basada en adyacencia, la similitud es simplemente el peso del arco entre los nodos

$$\mathcal{L} = \sum_{(u,v)\in V \times V} \| \mathbf{z}_u^\top \mathbf{z}_v - \mathbf{A}_{u,v} \|^2$$

Algunas desventajas

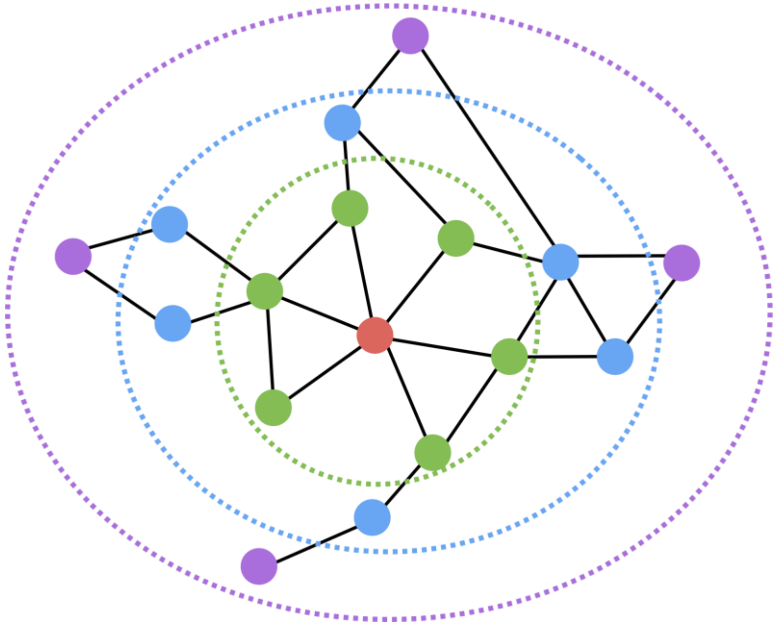
- $O(|V|^2)$: considera todos los pares
- $O(|V|)$ parámetros, un vector por nodo
- Solo considera conexiones directas



Nodo azul es más parecido al verde que el rojo, a pesar que ambos no están conectados a él.

En la basada en múltiples *hops*, la similitud considera vecindarios

$$\mathcal{L} = \sum_{(u,v)\in V\times V} \|\mathbf{z}_u^\top \mathbf{z}_v - \boxed{\mathbf{A}_{u,v}^k}\|^2$$



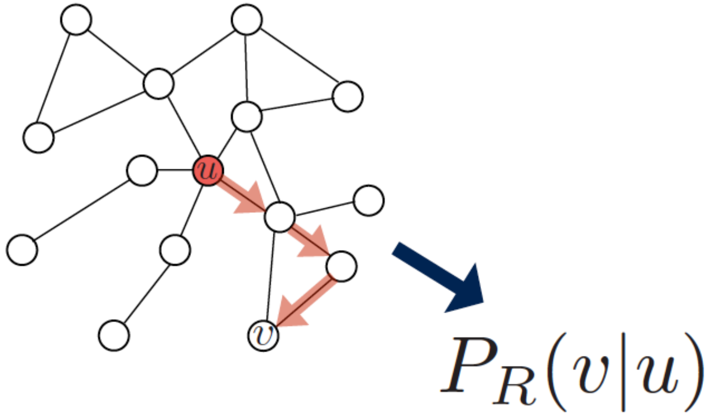
- Rojo: nodo objetivo
- Verde: vecindario de 1-hop (A)
- Azul: vecindario de 2-hop (A^2)
- Morado: vecindario de 3-hop (A^3)

Similitud de *random walks* es la más exitosa actualmente

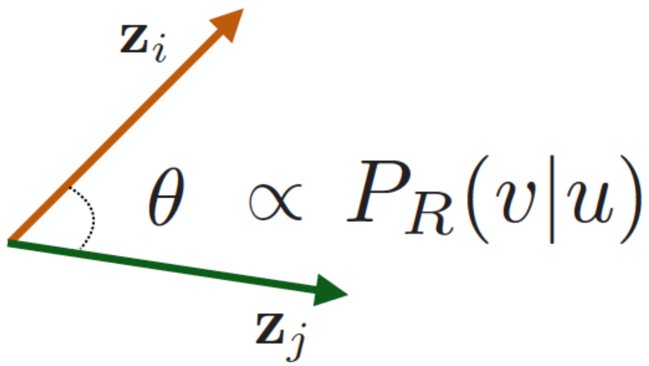
$$\mathbf{z}_u^\top \mathbf{z}_v \approx$$ Probabilidad que *u* y *v* co-ocurran en un *random walk* sobre el grafo

Similitud de *random walks* es la más exitosa actualmente

1. Estimar la probabilidad de visitar al nodo *v* en un *random walk* partiendo de *u,* una estrategia de recorrido *R*.



2. Optimizar *embedding* para capturar probabilidades estimadas



¿Por qué la similitud de *random walks* es la más exitosa actualmente?

- Expresividad: definición flexible y estocástica de similitud que permite incorporar información de interacciones locales y de mayor orden.
- Eficiencia: no necesita considerar todos los pares de nodos, solo aquellos que co-ocurren en los recorridos.

¿Cómo funciona en la práctica?

1. Realizar recorridos cortos partiendo de cada nodos del grafo, usando alguna estrategia R.
2. Para cada nodo *u,* recolectar $N_R(u)$, el conjunto de nodos visitados en el recorrido desde *u* (posiblemente con nodos repetidos).
3. Optimizar *embeddings*:

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log(P(v|\mathbf{z}_u))$$

¿Cómo funciona en la práctica?

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log(P(v|\mathbf{z}_u))$$

Intuición: optimizar embeddings para maximizar la probabilidad de co-ocurrencia en un recorrido, para lo que parametrizamos $P(v|z_u)$ usando un *softmax*:

$$P(v|\mathbf{z}_u) = \frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)}$$

¿Cómo funciona en la práctica?

$$\mathcal{L} = \underbrace{\boxed{\sum_{u \in V}}}_{\text{suma sobre todos los nodos u}} \underbrace{\boxed{\sum_{v \in N_R(u)}}}_{\text{suma sobre todos los nodos v vistos en un recorrido comenzado en u}} - \log\left(\underbrace{\boxed{\frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)}}}_{\text{probabilidad de co-ocurrencia}} \right)$$

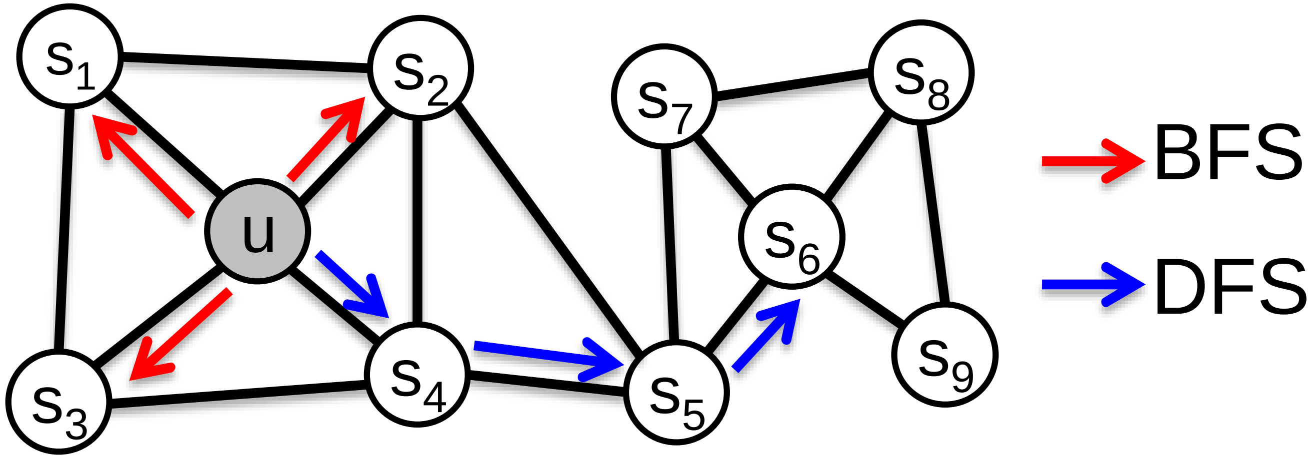
suma sobre todos los nodos u

suma sobre todos los nodos v vistos en un recorrido comenzado en u

probabilidad de co-ocurrencia

Node2vec es uno de los más utilizados

Se basa en la idea de usar random walks sesgados, mezclando vistas locales (BFS) y globales (DFS) del grafo.



Node2vec es uno de los más utilizados

On Network Embedding for Machine Learning on Road Networks: A Case Study on the Danish Road Network

Tobias Skovgaard Jepsen, Christian S. Jensen, Thomas Dyhre Nielsen, Kristian Torp
Department of Computer Science
Aalborg University, Aalborg, Denmark
{tsj, csj, tdn, torp}@cs.aau.dk

***Abstract*—Road networks are a type of spatial network, where edges may be associated with qualitative information such as road type and speed limit. Unfortunately, such information is often incomplete; for instance, OpenStreetMap only has speed limits for 13% of all Danish road segments. This is problematic for analysis tasks that rely on such information for machine learning. To enable machine learning in such circumstances, one may consider the application of network embedding methods to extract structural information from the network. However, these methods have so far mostly been used in the context of social networks, which differ significantly from road networks in terms of, e.g., node degree and level of homophily (which are key to the performance of many network embedding methods).**

We analyze the use of network embedding methods, specifically node2vec, for learning road segment embeddings in road networks. Due to the often limited availability of information on other relevant road characteristics, the analysis focuses on leveraging the spatial network structure. Our results suggest that network embedding methods can indeed be used for deriving relevant network features (that may, e.g, be used for predicting speed limits), but that the qualities of the embeddings differ from embeddings for social networks.

***Keywords*—road network, machine learning, feature learning, network embedding**

I. Introduction

Road networks represent an important class of spatial networks and are an essential component of modern societal infrastructure. Road networks are associated with many important analysis tasks such as traffic flow and travel pattern analyses. In particular, many important road network tasks are supported by machine learning algorithms, including travel-time estimation [1], [2], traffic forecasting [3], and k nearest points-of-interest queries [4], [5], that require set of informative features to describe, e.g., the different road segments.

Solving road network analysis tasks is difficult since there is often little information available beyond the network structure itself. For instance, the Danish road network from OpenStreetMap (OSM) [6] contains only the network structure and up to two attributes characterizing each road segments: road category and speed limit. In addition, only 13% of the road segments have a speed limit label, even when augmented with data from Danish municipalities. This information sparsity makes it difficult to derive the features necessary for solving many road network analysis tasks. The road network structure is a potentially rich source of information, but it is not straightforward to capture and utilize this often highly complex structure. For road network analyses, this typically involves explicit modeling of spatial correlations between adjacent road segments based on domain knowledge [1], [7], [8].

A road network is commonly modeled as a directed graph $G = (V, E)$, where each node $v \in V$ represents an intersection or the end of a road and each edge $(u, v) \in E$ represents a directed road segment that allows travel from u to v. Such graph representations makes *network embedding methods*—a class of feature learning methods for graphs—directly applicable for extracting structural information from road networks.

In network embedding, the goal is to learn a mapping (an *embedding*) that embeds nodes in networks into a d-dimensional vector space s.t. the node neighborhoods are preserved in the embedding space [9]. In other words, nodes are mapped to feature vectors that encode the structural information of the graph s.t. nearby nodes in the network are mapped to vectors that are near each other in the embedding space. For instance, Fig. 1b shows that road segments north and south of the bridge in Fig. 1a tend to cluster with other road segments from the same region. The road segments representing the bridge are somewhere in-between. Network embedding methods can extract the structural information in networks to supplement or replace attribute information if such information is low-quality, sparse, or unavailable.

The research in network embedding has thus far focused primarily on social, biological, and information networks [10]–[17]. Such networks differ significantly from road networks in terms of, e.g., structure, semantics, size, node degree, network diameter, and the amount of attribute information available. In addition, road networks may be disconnected due to inaccuracies in their spatial representation or the presence of islands, whereas, e.g., social networks are strongly connected. The effect of this disconnectedness on the embeddings is not obvious.

The differences between the types of networks studied in the network embedding literature, e.g., social networks, and road

Graph Embeddings for Street Network Analysis

Patrick DeMichele, Pablo Santos, and Isaac Scheinfeld

December 2019

Abstract

The field of street network analysis has not yet benefited from much of the recent work on graph machine learning. We extend a framework for large-scale analysis of OpenStreetMap data with recently released traffic data from Uber, and apply node embedding techniques to study the street networks of New York City and San Francisco. We present results for unsupervised clustering-based role discovery and supervised models for predicting speeds and a proxy for vulnerability to congestion.

1 Introduction

In this paper, we model the street networks of New York and San Francisco with data from OpenStreetMap (OSM) [19], an open source collection of real world street data. This allowed us to model street networks as graphs with nodes representing street intersections and edges representing streets. Using features from OSM, we construct embedding vectors for streets which can then be used in a variety of unsupervised and supervised models.

We examine the results of unsupervised role discovery on these embeddings as a first approach to interpreting what information they encode. Local, recursively generated features seem to cluster according to road type, while random-walk based node2vec features are similar within neighborhoods.

Modeling traffic flow in cities such as New York and San Francisco requires congestion-aware models since both cities suffer from frequent traffic jams. For each of these two networks, we experiment with different models to predict the mean speed – provided by Uber Movement [2] in a recently released dataset – as well as the "congestion" (according to a metric we define) of a given street. The relative success of these models demonstrates the power of graphical properties alone in explaining congestion and mean speed, indeed many of the models we train do not use any features besides network topology.

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería de Transporte y Logística

Sistemas Urbanos Inteligentes

Node Embeddings

Hans Löbel
Dpto. Ingeniería de Transporte y Logística
Dpto. Ciencia de la Computación