Pontificia Universidad
JAVERIANA
Colombia

Procesamiento de Lenguaje Natural

Tópicos Avanzados en Analítica
Maestría en Analítica para la Inteligencia de Negocios

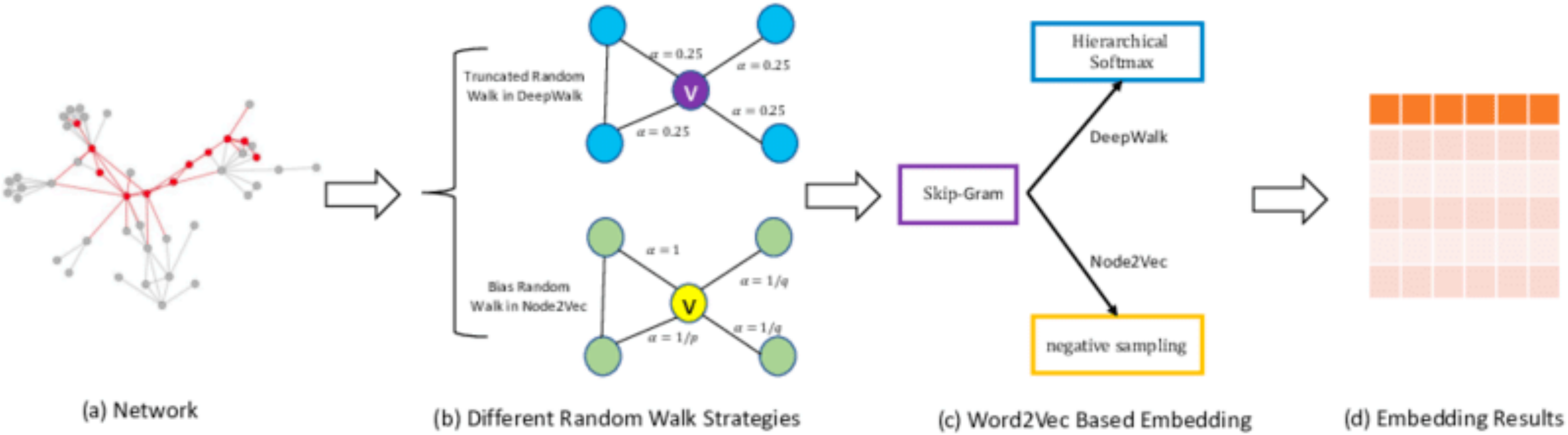
Sergio Alberto Mora Pardo - H2 2023

“Conocerás una palabra por la compañía que tiene”

JR Fiordo - 1957

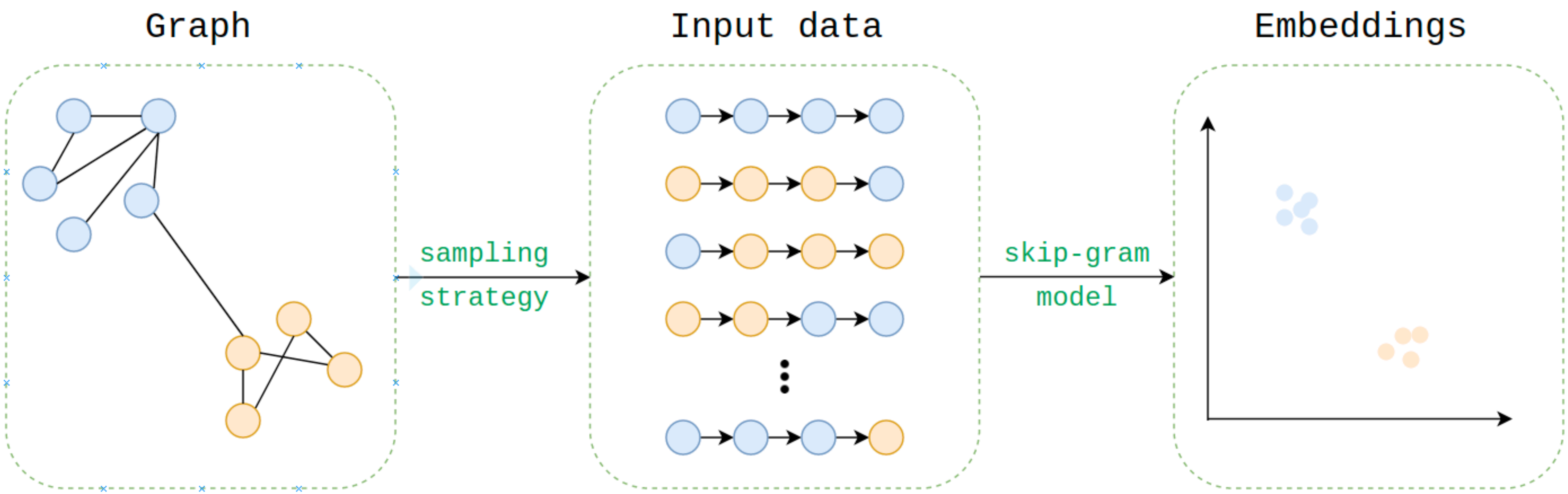
Graph Representation

Node2Vec



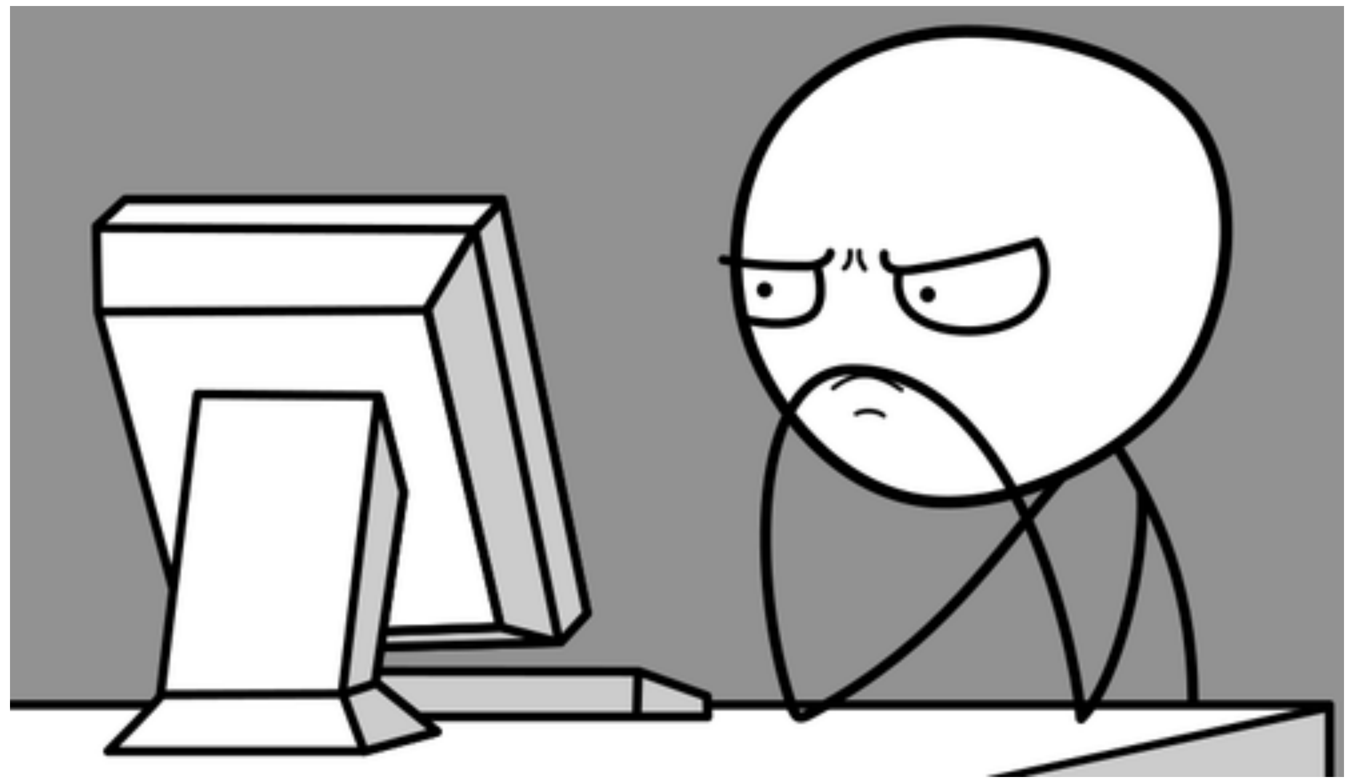
Graph Representation

Node2Vec



Text Representation

Embeddings

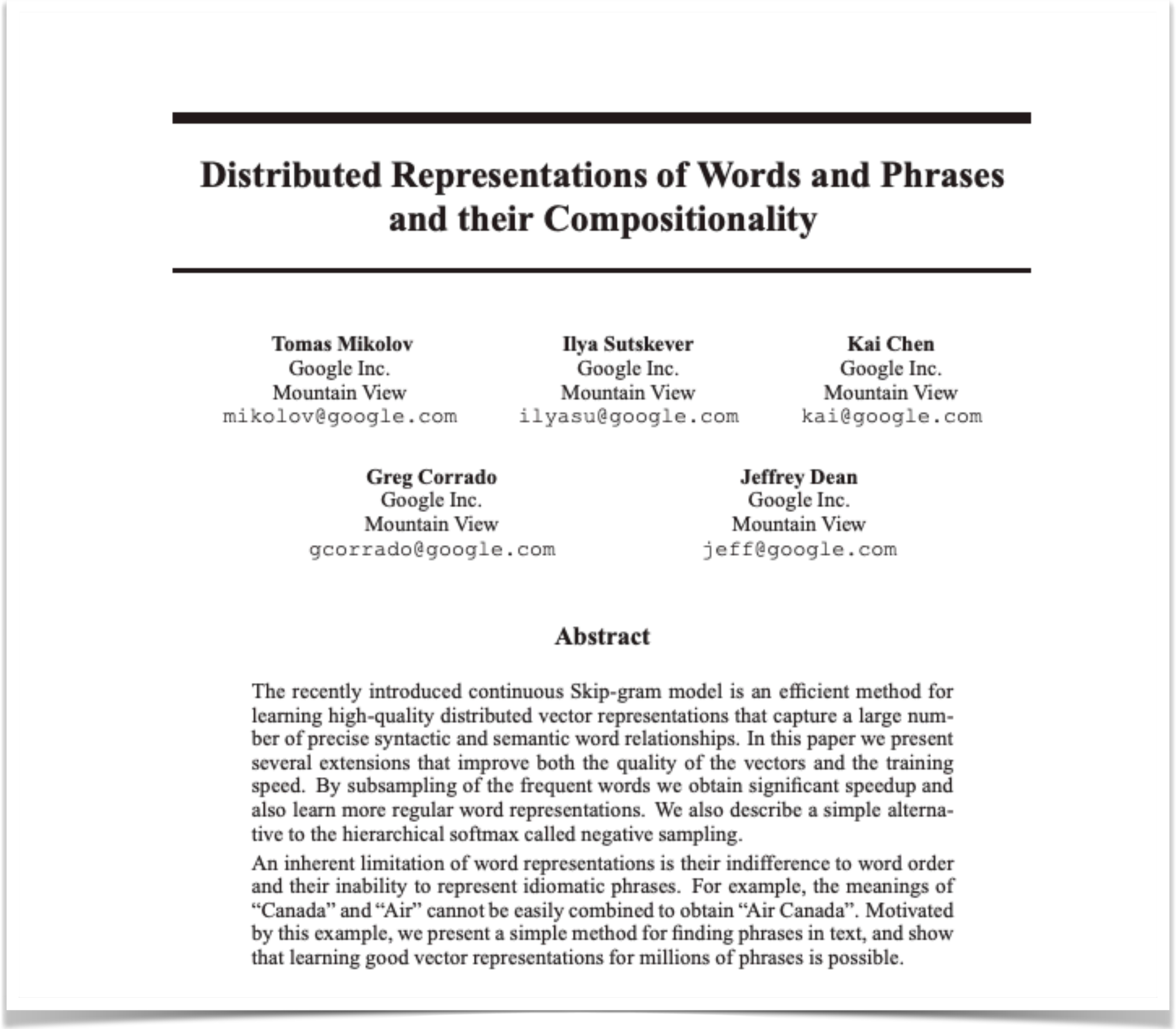


queen = king - he + she?

$$P(w_{i-m}, \dots, w_{i-1}, w_{i+1}, \dots, w_{i+m} | w_i) = \prod_{j \neq i \wedge j = i-m}^{i+m} P(w_j | w_i)$$

Text Representation

Embeddings



Distributed Representations of Words and Phrases and their Compositionality

Tomas Mikolov
Google Inc.
Mountain View
mikolov@google.com

Ilya Sutskever
Google Inc.
Mountain View
ilyasu@google.com

Kai Chen
Google Inc.
Mountain View
kai@google.com

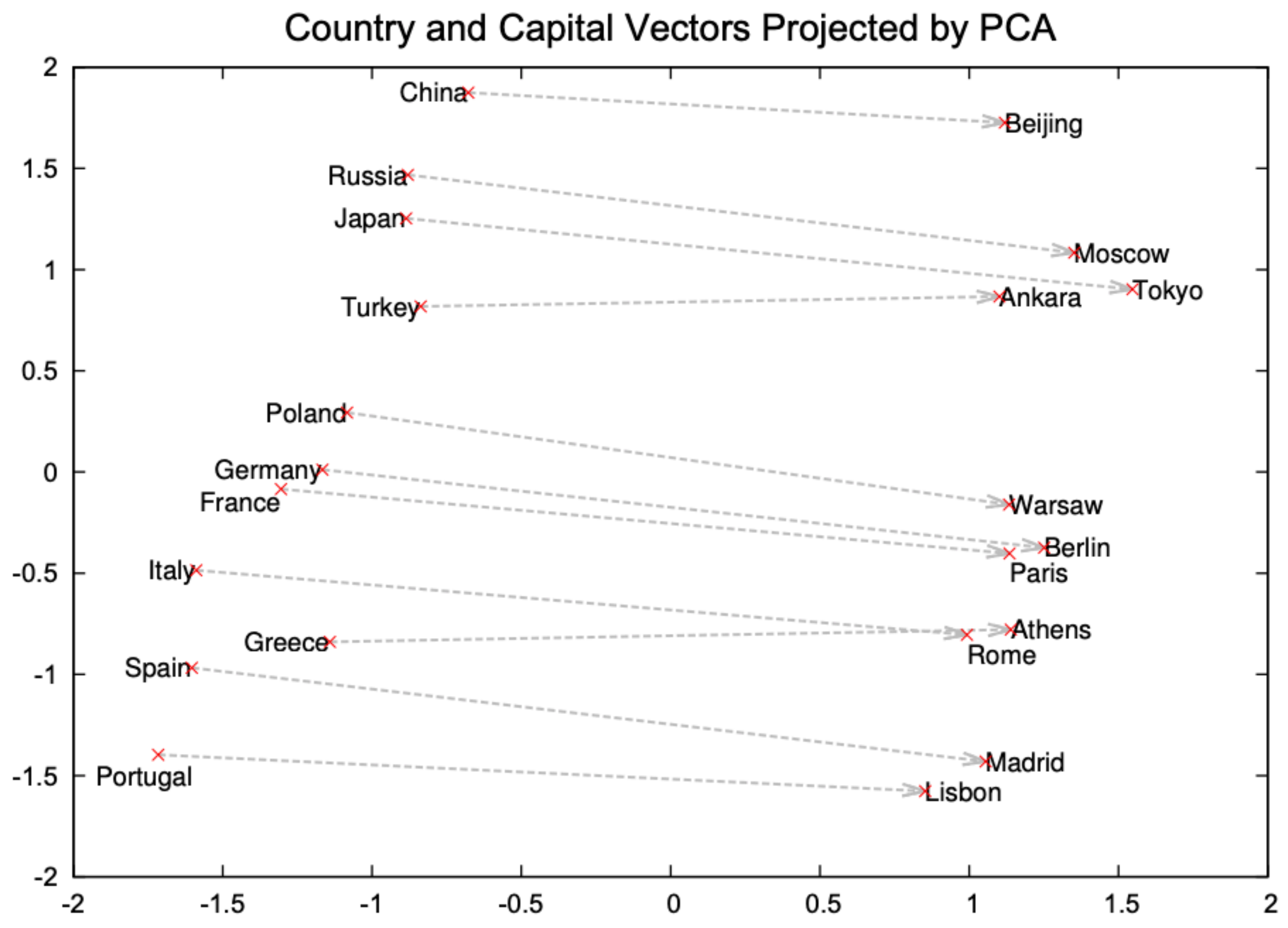
Greg Corrado
Google Inc.
Mountain View
gcorrado@google.com

Jeffrey Dean
Google Inc.
Mountain View
jeff@google.com

Abstract

The recently introduced continuous Skip-gram model is an efficient method for learning high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships. In this paper we present several extensions that improve both the quality of the vectors and the training speed. By subsampling of the frequent words we obtain significant speedup and also learn more regular word representations. We also describe a simple alternative to the hierarchical softmax called negative sampling.

An inherent limitation of word representations is their indifference to word order and their inability to represent idiomatic phrases. For example, the meanings of "Canada" and "Air" cannot be easily combined to obtain "Air Canada". Motivated by this example, we present a simple method for finding phrases in text, and show that learning good vector representations for millions of phrases is possible.



Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

Pronóstico con base en la palabra actual.

No pronostica basándose en el contesto (palabras anteriores y posteriores)

La palabra en la ranura verde sería la palabra de entrada, cada cuadro rosa sería una posible salida.

Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

Pronóstico con base en la palabra actual.

No pronostica basándose en el contesto (palabras anteriores y posteriores)

Jay was hit by a red bus in...

by	a	red	bus	in

input	output
red	by
red	a
red	bus
red	in

Esta ventana deslizante crea cuatro muestras separadas.

Text Representation

Word2Vec

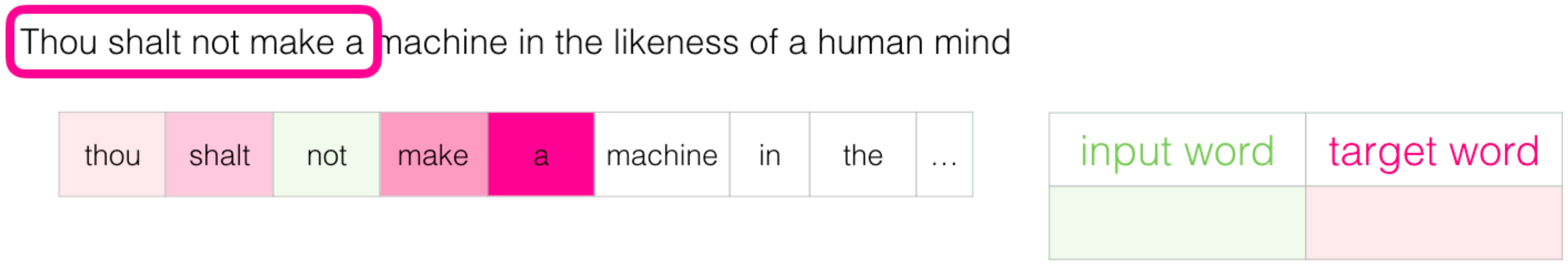
SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Ventana deslizante:



Esta ventana deslizante crea cuatro muestras separadas.

Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Ventana deslizante:

Thou shalt not make a machine in the likeness of a human mind

thou	shalt	not	make	a	machine	in	the	…

input word	target word
not	thou
not	shalt
not	make
not	a

Ahora a la siguiente posición…

Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Ventana deslizante:

Thou shalt not make a machine in the likeness of a human mind

thou	shalt	not	make	a	machine	in	the	…

thou	shalt	not	make	a	machine	in	the	…

input word	target word
not	thou
not	shalt
not	make
not	a

Lo que genera cuatro registros más…

Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Ventana deslizante:

Thou shalt not make a machine in the likeness of a human mind

thou	shalt	not	make	a	machine	in	the	…

thou	shalt	not	make	a	machine	in	the	…

input word	target word
not	thou
not	shalt
not	make
not	a
make	shalt
make	not
make	a
make	machine

Un par adelante veríamos algo como…

Text Representation

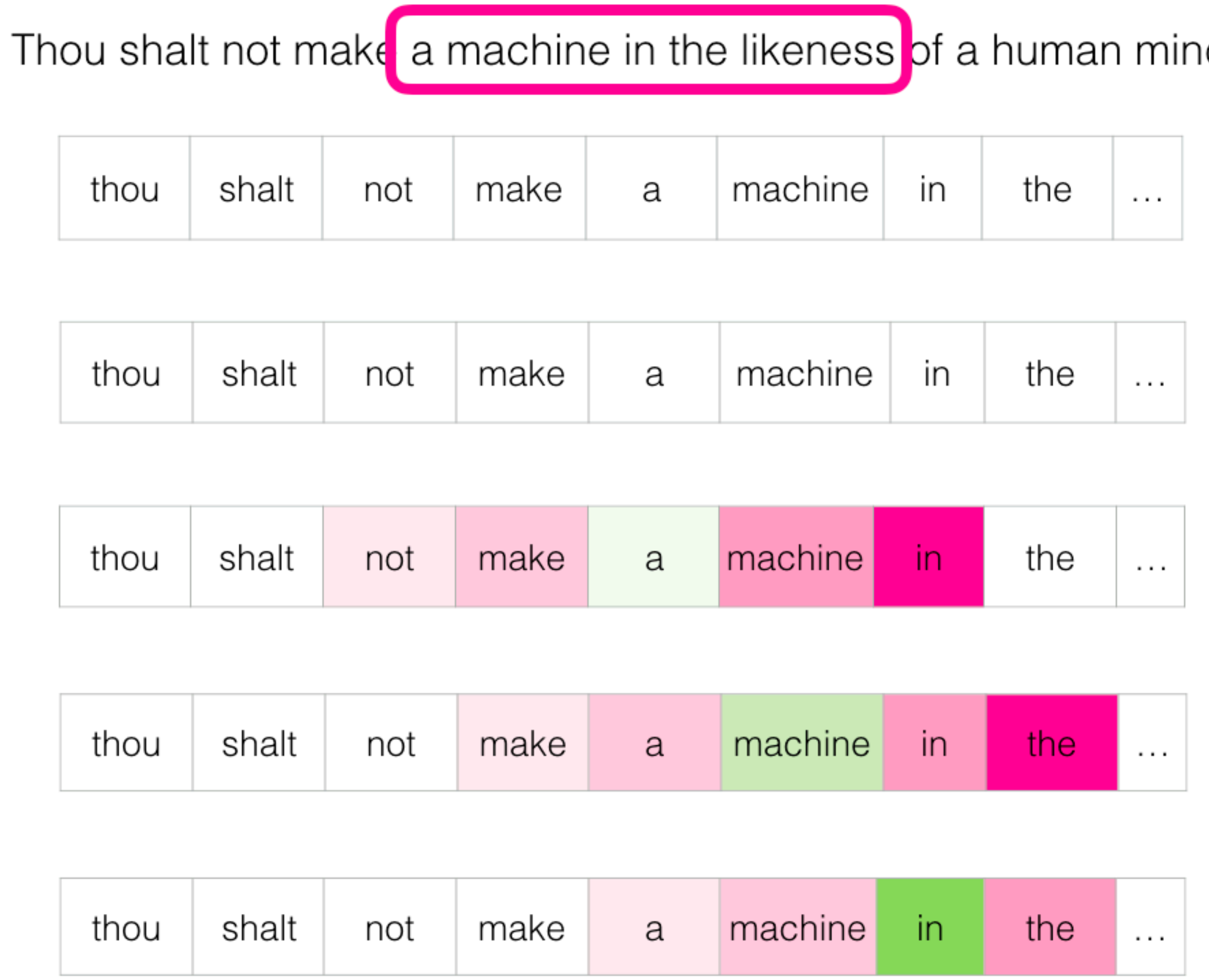
Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

Pronóstico con base en la palabra actual.

Ventana deslizante:

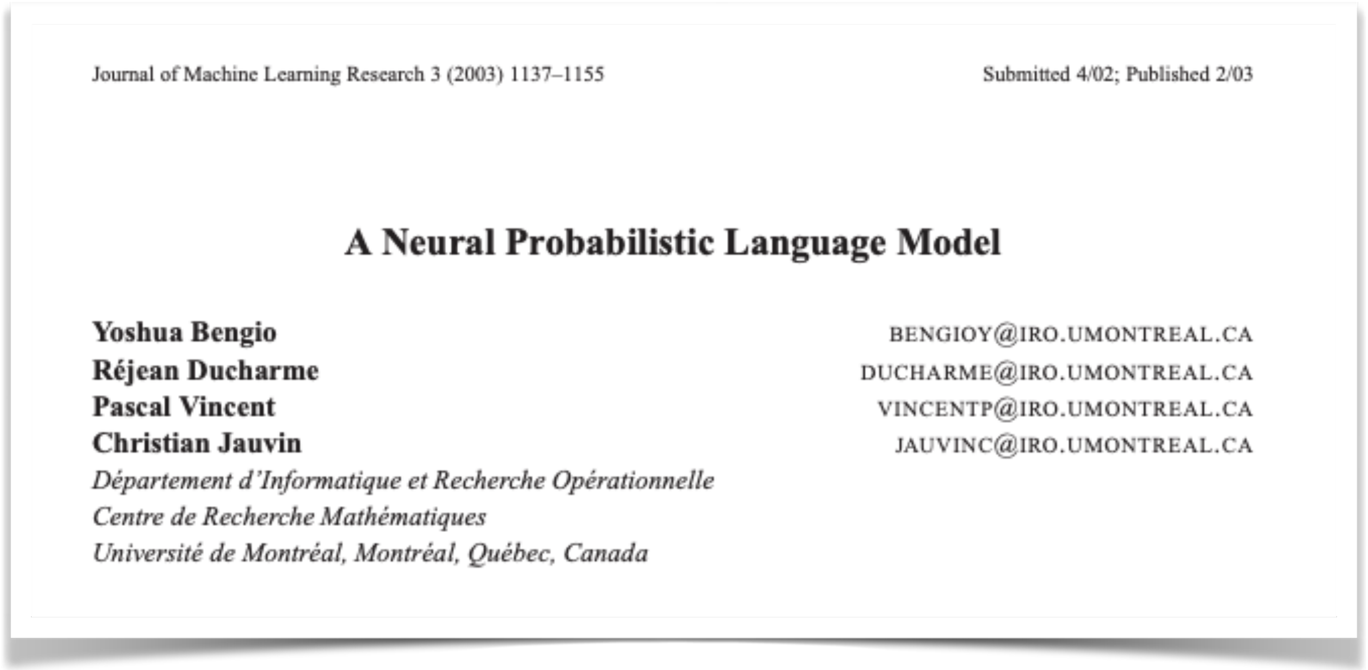


input word	target word
not	thou
not	shalt
not	make
not	a
make	shalt
make	not
make	a
make	machine
a	not
a	make
a	machine
a	in
machine	make
machine	a
machine	in
machine	the
in	a
in	machine
in	the
in	likeness

Text Representation

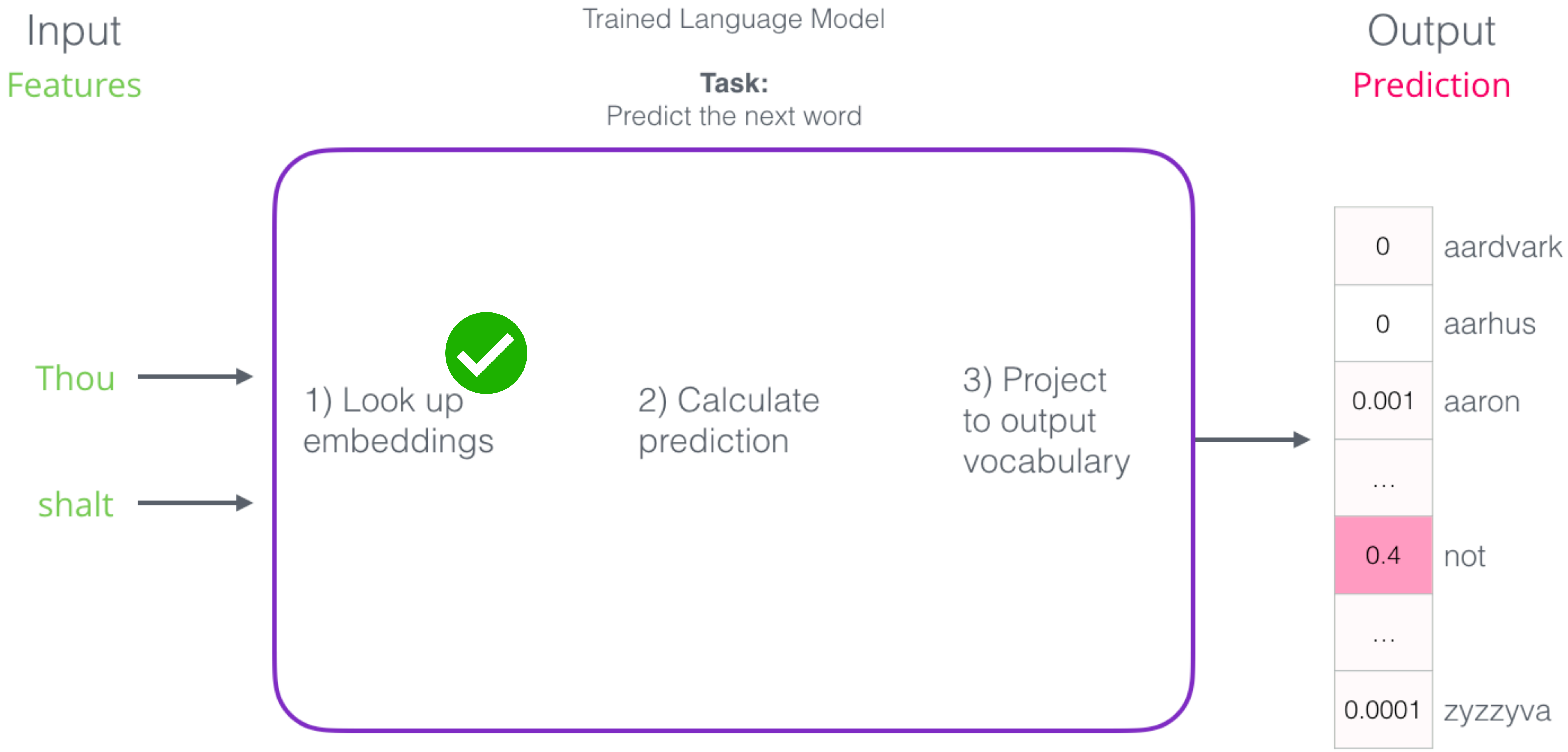
Word2Vec

Predicción en tres pasos, de acuerdo con Bengio 2003


Journal of Machine Learning Research 3 (2003) 1137–1155 Submitted 4/02; Published 2/03

A Neural Probabilistic Language Model

Yoshua Bengio BENGIOY@IRO.UMONTREAL.CA
Réjean Ducharme DUCHARME@IRO.UMONTREAL.CA
Pascal Vincent VINCENTP@IRO.UMONTREAL.CA
Christian Jauvin JAUVINC@IRO.UMONTREAL.CA
Département d'Informatique et Recherche Opérationnelle
Centre de Recherche Mathématiques
Université de Montréal, Montréal, Québec, Canada



Text Representation

Word2Vec

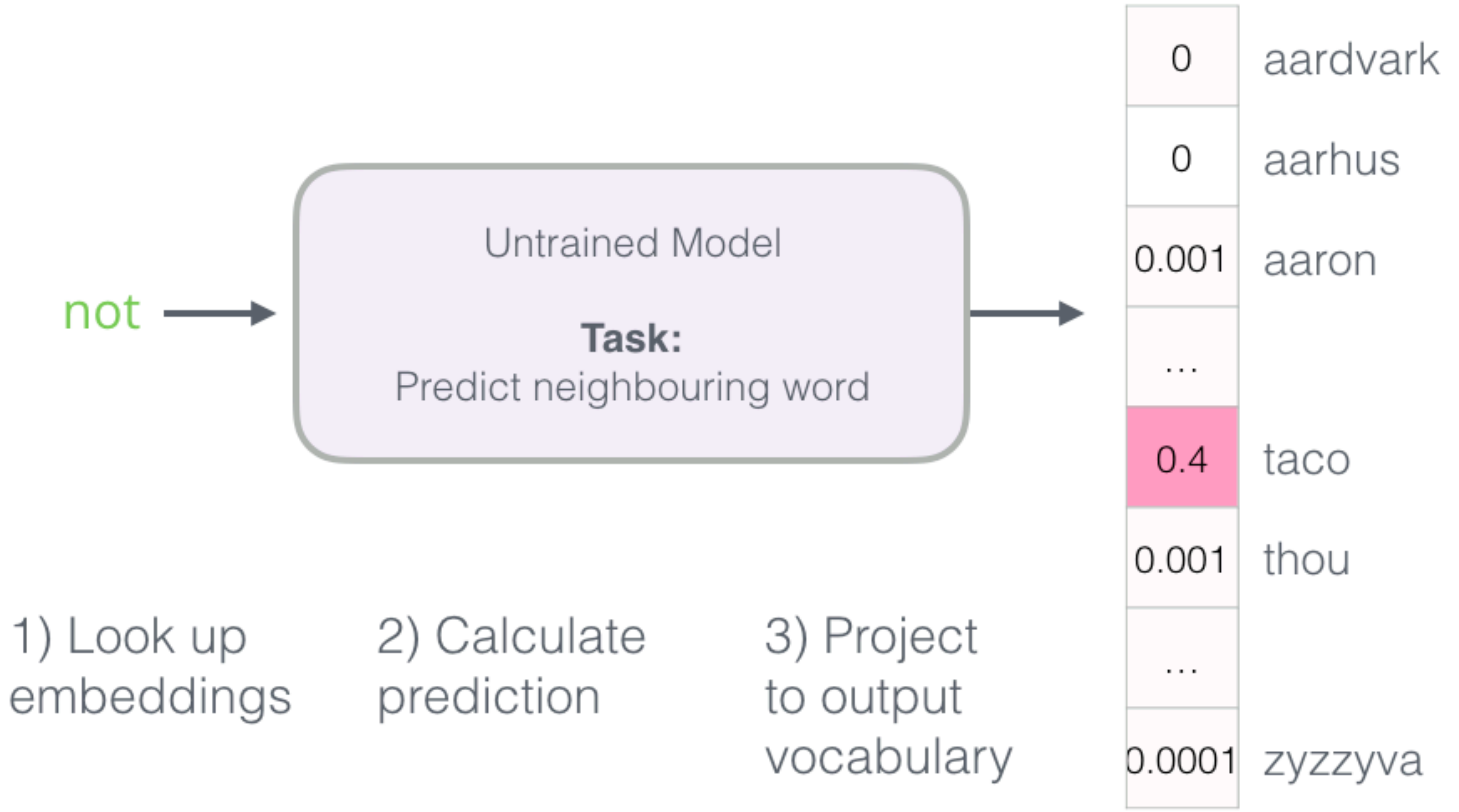
SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Entrenamiento:



Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Entrenamiento (iteración):

1. El modelo realiza los tres pasos y genera un vector de predicción (con una probabilidad asignada a cada palabra de su vocabulario).

2. Dado que el modelo no está entrenado, es seguro que su predicción será incorrecta en esta etapa.

3. Sabemos qué palabra debería haber adivinado: la etiqueta/celda de salida en la fila que estamos usando actualmente para entrenar el modelo

Text Representation

Word2Vec

SkipGram:
Language Modeling

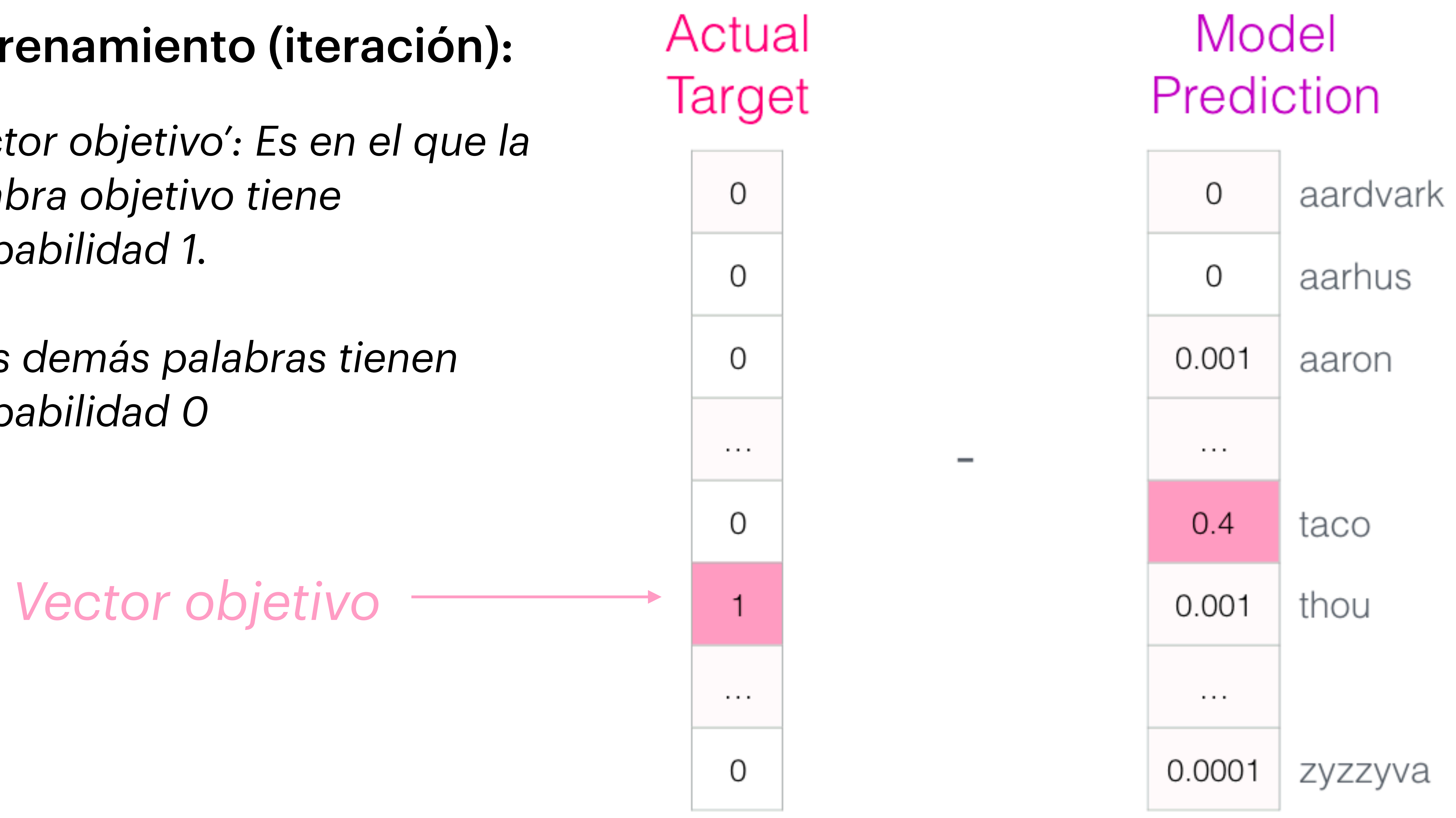
La predicción de la siguiente palabra.

Pronóstico con base en la palabra actual.

Entrenamiento (iteración):

'Vector objetivo': Es en el que la palabra objetivo tiene probabilidad 1.

Y las demás palabras tienen probabilidad 0



Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

Pronóstico con base en la palabra actual.

Entrenamiento (iteración): *¿A que distancia estaba el modelo?*

Actual Target		Model Prediction			Error
0		0	aardvark		0
0		0	aarhus		0
0		0.001	aaron		-0.001
…	-	…		=	…
0		0.4	taco		-0.4
1		0.001	thou		0.999
…		…			…
0		0.0001	zyzzyva		-0.0001

Text Representation

Word2Vec

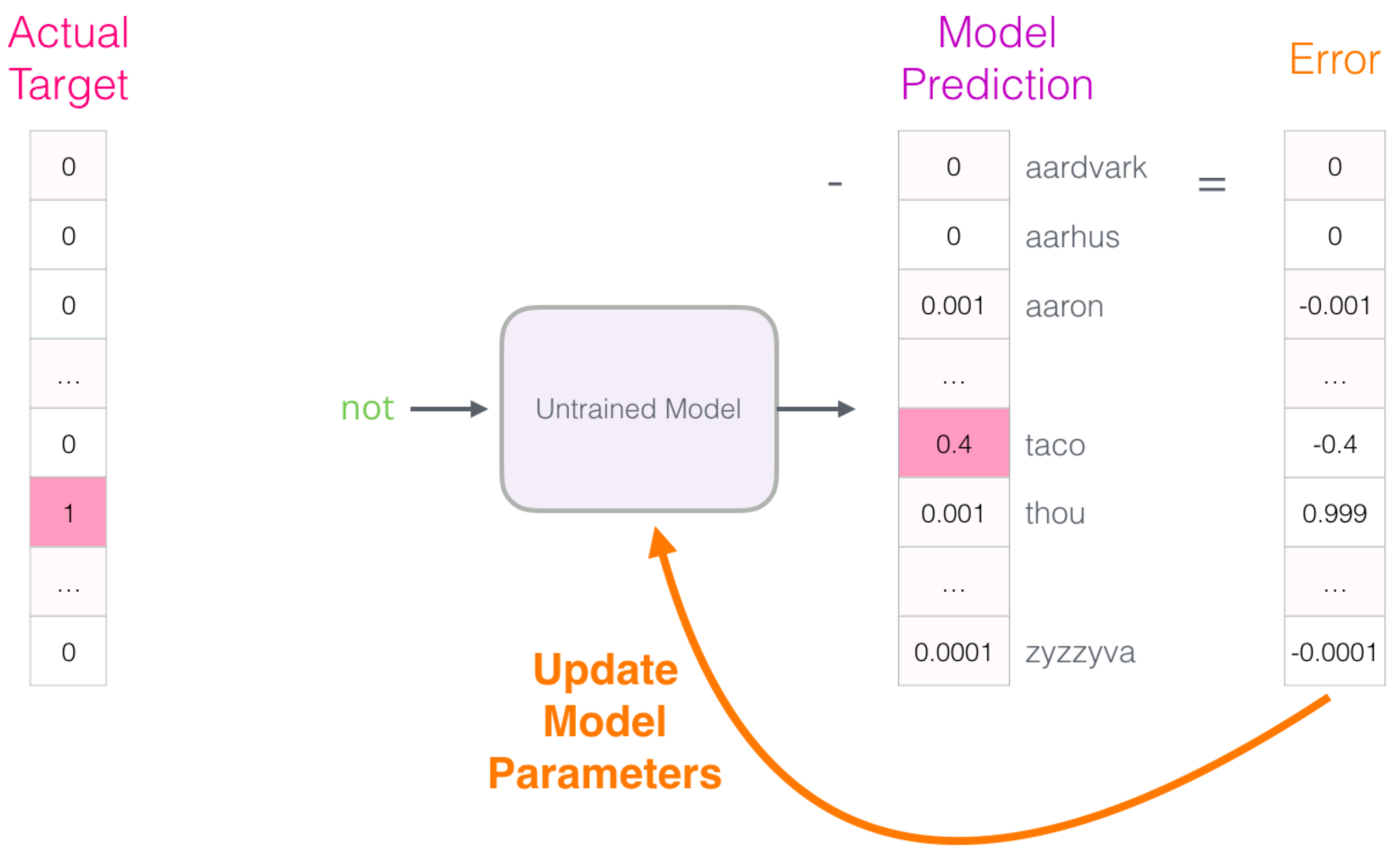
SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Entrenamiento (iteración):

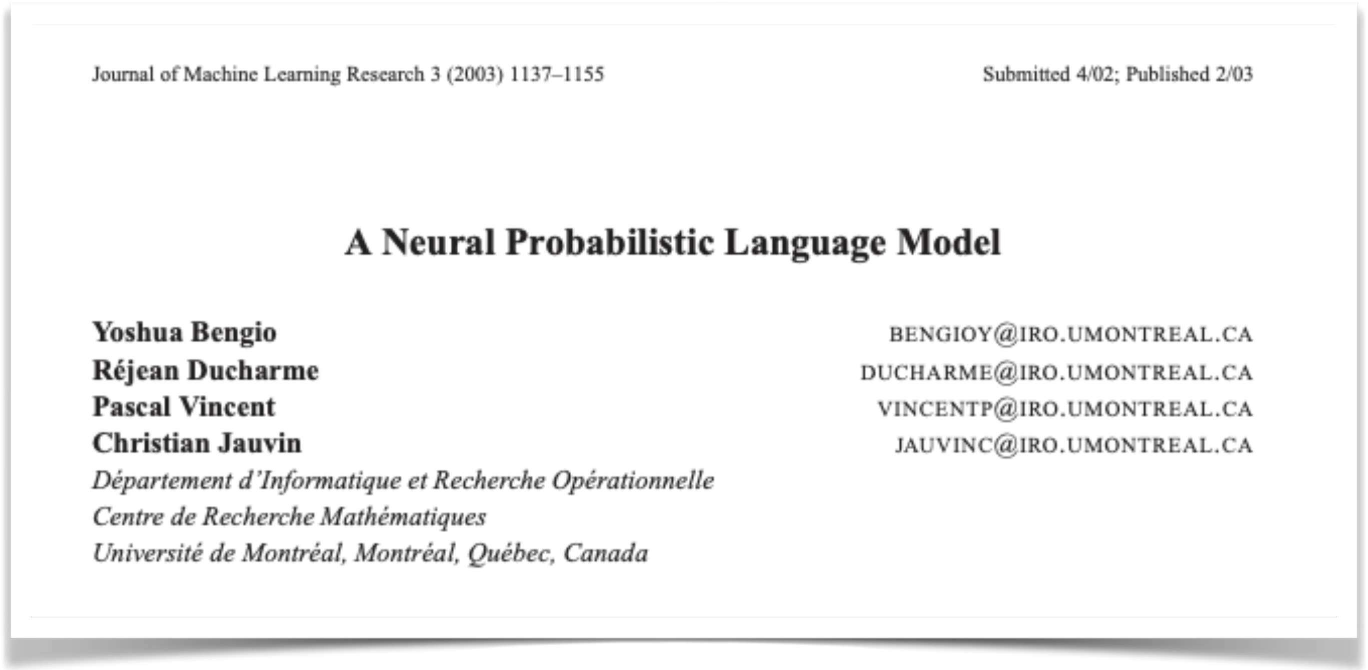


Actualizamos el modelo con el vector de error.

Text Representation

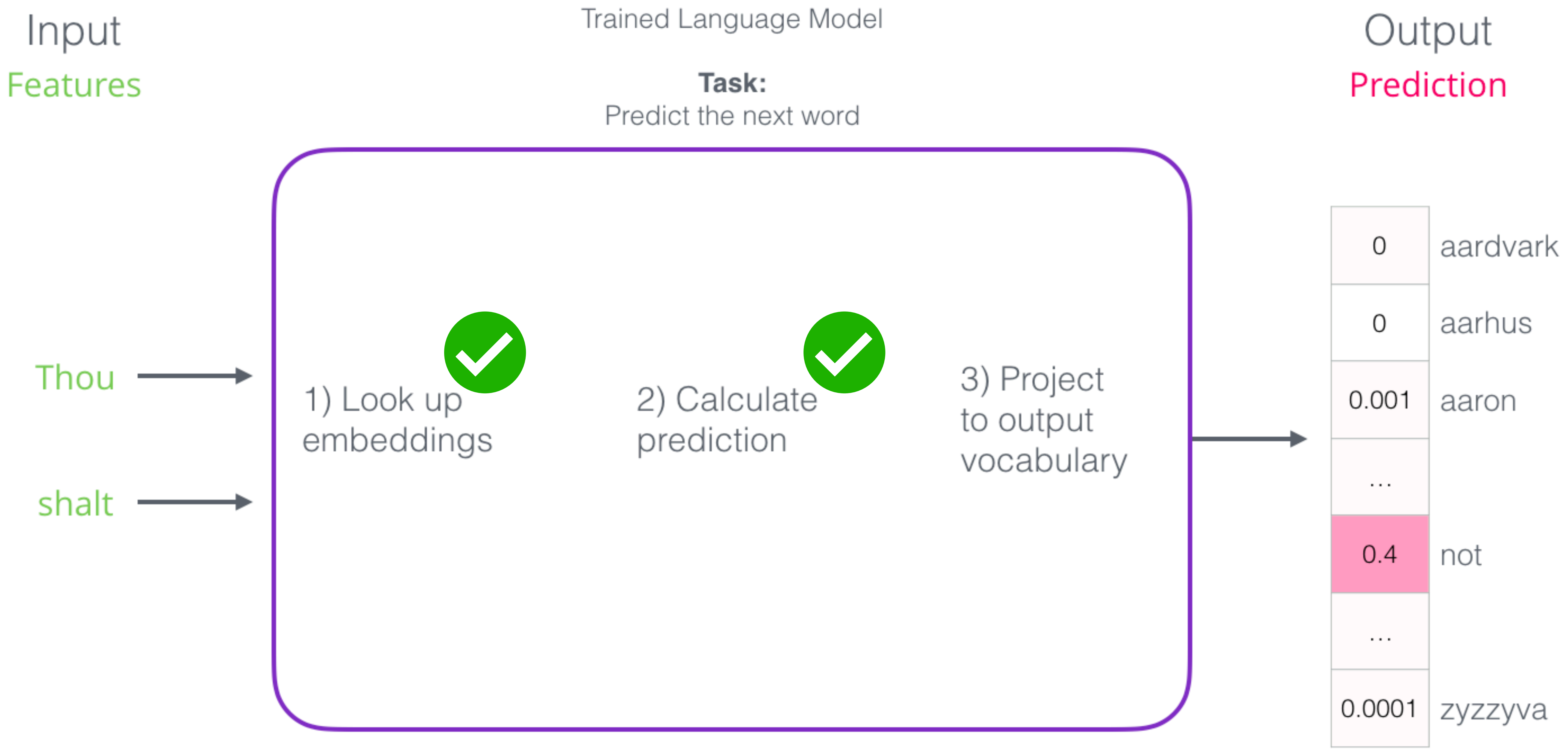
Word2Vec

Predicción en tres pasos, de acuerdo con Bengio 2003


Journal of Machine Learning Research 3 (2003) 1137–1155 Submitted 4/02; Published 2/03

A Neural Probabilistic Language Model

Yoshua Bengio BENGIOY@IRO.UMONTREAL.CA
Réjean Ducharme DUCHARME@IRO.UMONTREAL.CA
Pascal Vincent VINCENTP@IRO.UMONTREAL.CA
Christian Jauvin JAUVINC@IRO.UMONTREAL.CA
Département d'Informatique et Recherche Opérationnelle
Centre de Recherche Mathématiques
Université de Montréal, Montréal, Québec, Canada



Text Representation

Word2Vec

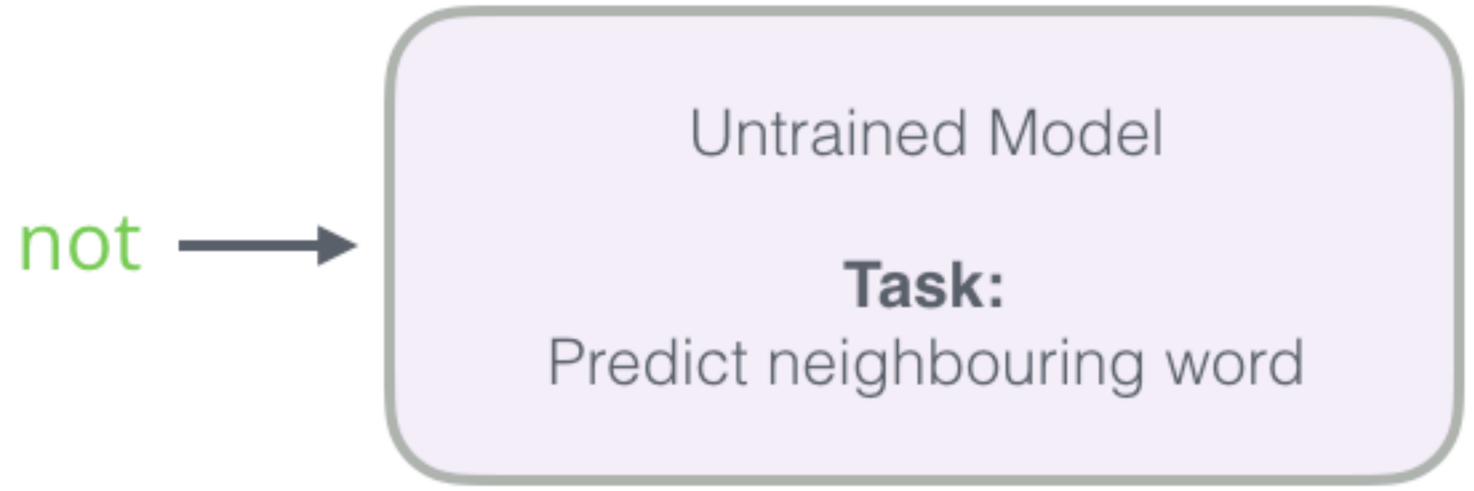
SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Project vocabulary:



1) Look up embeddings

2) Calculate prediction

3) Project to output vocabulary

[Computationally Intensive]

Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Project vocabulary:

Costoso computacionalmente

Una vez por cada muestra de entrenamiento en el conjunto de datos.

Decenas de millones de veces.

Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Project vocabulary:

1. Generar embeddings de palabras de alta calidad →	*No preocuparse por la predicción del siguiente pronóstico.*
2. Utilizar estas incorporaciones de alta calidad para entrenar un modelo de lenguaje →	*Realizar predicciones de la siguiente palabra.*

Text Representation

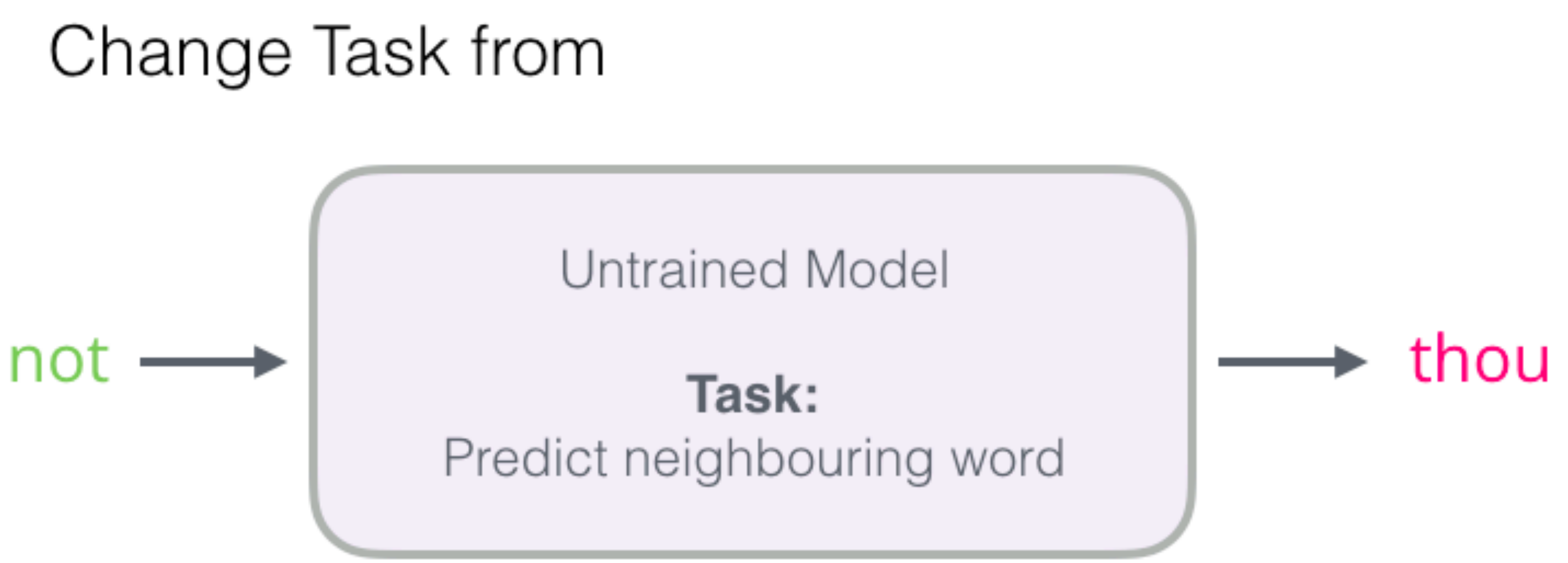
Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

Pronóstico con base en la palabra actual.

Project vocabulary: *Cambiar la tarea del modelo de predecir la palabra vecina*



Text Representation

Word2Vec

SkipGram:
Language Modeling

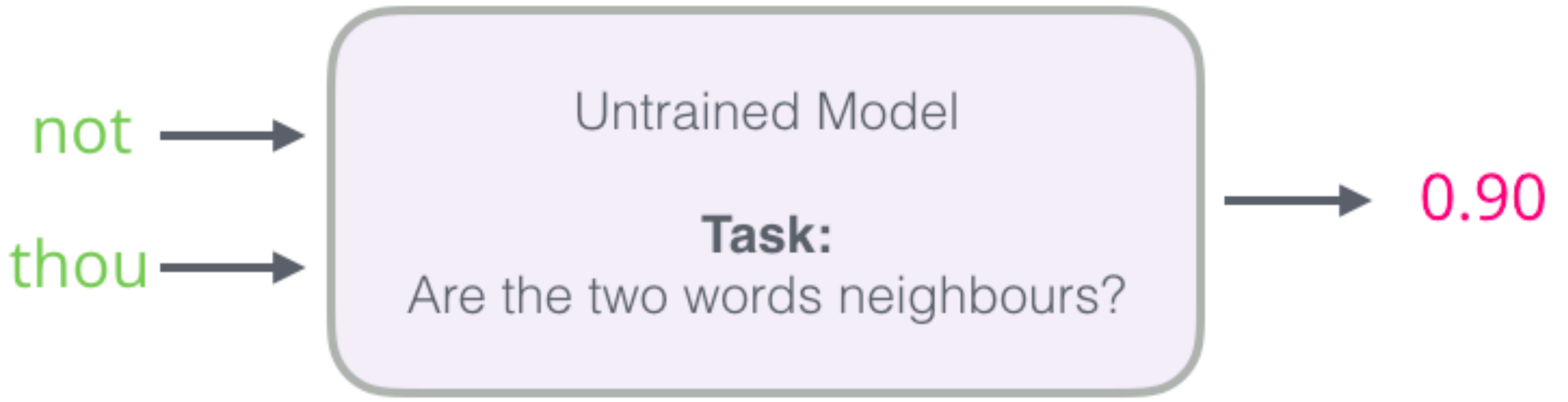
La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Project vocabulary: *Nueva tarea, decir si son vecinos o no. 1 para "vecinos".*

To:



Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Project vocabulary: *Cambio en nuestro conjunto de datos. Nueva etiqueta con valores 0 y 1.*

input word	target word
not	thou
not	shalt
not	make
not	a
make	shalt
make	not
make	a
make	machine

input word	output word	target
not	thou	1
not	shalt	1
not	make	1
not	a	1
make	shalt	1
make	not	1
make	a	1
make	machine	1

Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

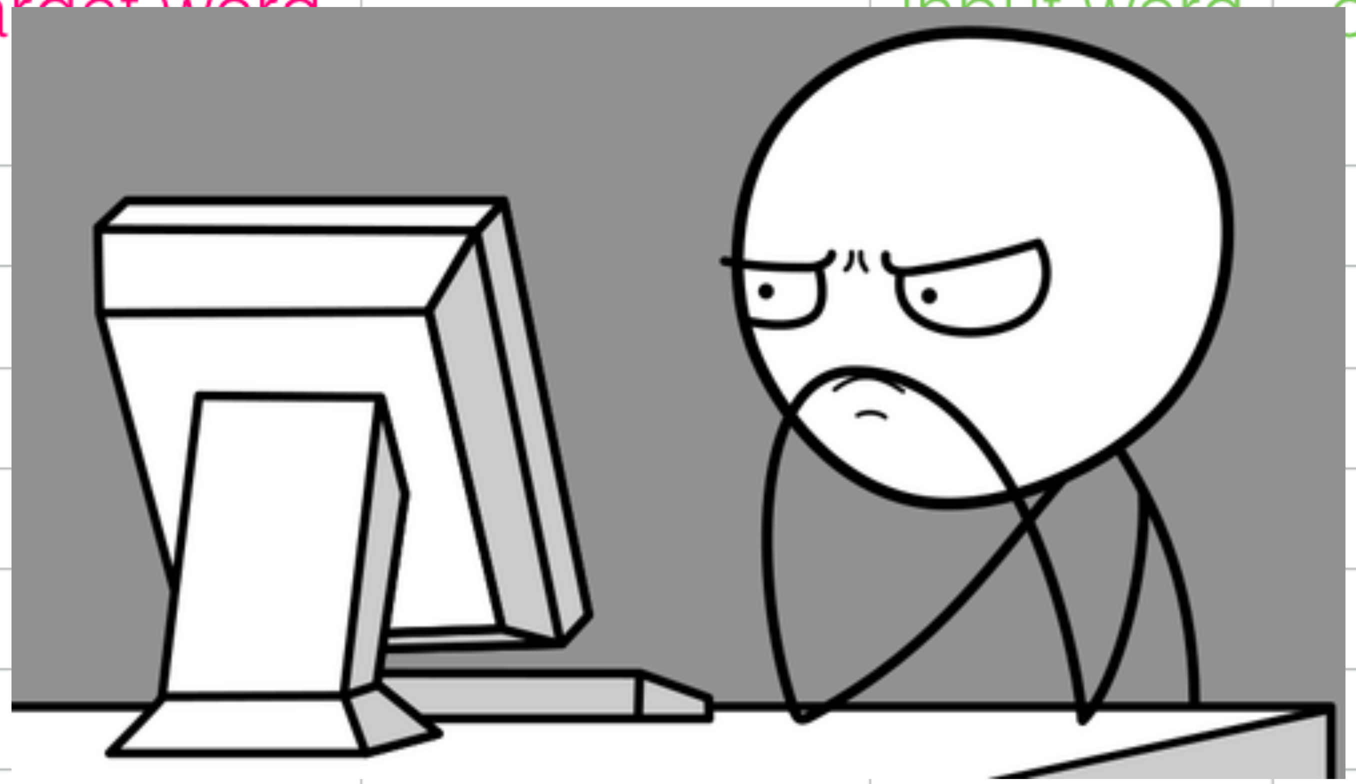
↓

Pronóstico con base en la palabra actual.

Project vocabulary: *Cambio en nuestro conjunto de datos. Nueva etiqueta con valores 0 y 1.*

input word	target word
not	
not	
not	
not	
make	
make	
make	a
ma	

input word	output word	target
	thou	1
	shalt	1
	make	1
	a	1
	shalt	1
	not	1
make	a	1
		1



¿Qué falla tiene el conjunto de datos?

Text Representation

Word2Vec

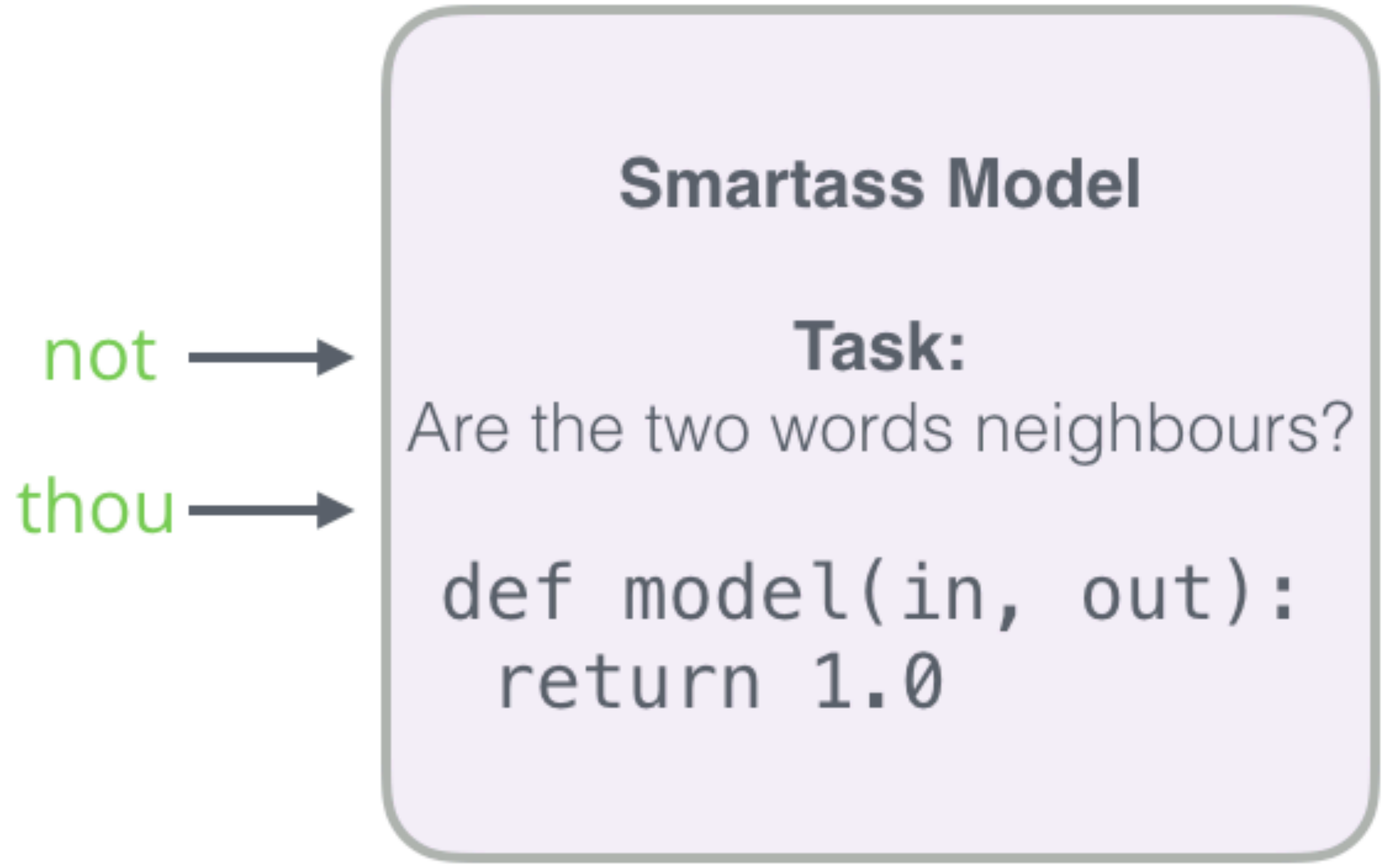
SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Project vocabulary: *Smartass Model Problem*



Todos nuestros objetivos son 1. El modelo puede devolver siempre 1, logrando una precisión del 100%. Pero no aprendió nada y genera embeddings basura.

Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

Pronóstico con base en la palabra actual.

Project vocabulary: *Introducir muestras negativas en el conjunto de datos*

input word	output word	target
not	thou	**1**
not		**0**
not		**0**
not	shalt	**1**
not	make	**1**

Negative examples

Palabras que no son vecinas.

Pontificia Universidad JAVERIANA Colombia

Text Representation

Word2Vec

SkipGram:
Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Project vocabulary: *Introducir muestras negativas en el conjunto de datos*

Pick randomly from vocabulary
(random sampling)

input word	output word	target
not	thou	1
not	aaron	0
not	taco	0
not	shalt	1
not	make	1

Word	Count	Probability
aardvark		
aarhus		
aaron		
taco		
thou		
zyzzyva		

Text Representation

Word2Vec

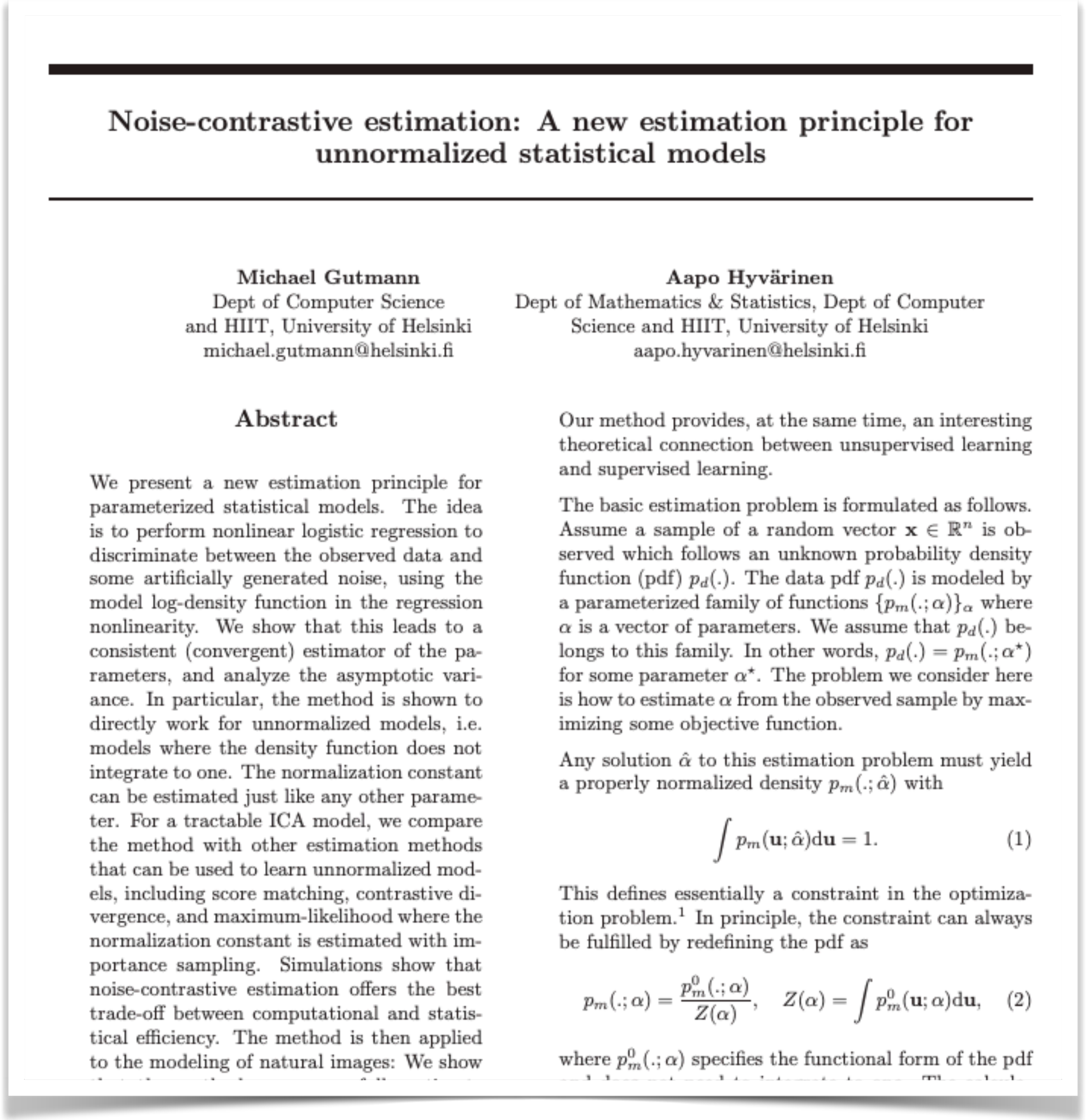
SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Noise-contrastive estimation: A new estimation principle for unnormalized statistical models



Noise-contrastive estimation: A new estimation principle for unnormalized statistical models

Michael Gutmann
Dept of Computer Science
and HIIT, University of Helsinki
michael.gutmann@helsinki.fi

Aapo Hyvärinen
Dept of Mathematics & Statistics, Dept of Computer Science and HIIT, University of Helsinki
aapo.hyvarinen@helsinki.fi

Abstract

We present a new estimation principle for parameterized statistical models. The idea is to perform nonlinear logistic regression to discriminate between the observed data and some artificially generated noise, using the model log-density function in the regression nonlinearity. We show that this leads to a consistent (convergent) estimator of the parameters, and analyze the asymptotic variance. In particular, the method is shown to directly work for unnormalized models, i.e. models where the density function does not integrate to one. The normalization constant can be estimated just like any other parameter. For a tractable ICA model, we compare the method with other estimation methods that can be used to learn unnormalized models, including score matching, contrastive divergence, and maximum-likelihood where the normalization constant is estimated with importance sampling. Simulations show that noise-contrastive estimation offers the best trade-off between computational and statistical efficiency. The method is then applied to the modeling of natural images: We show

Our method provides, at the same time, an interesting theoretical connection between unsupervised learning and supervised learning.

The basic estimation problem is formulated as follows. Assume a sample of a random vector $\mathbf{x} \in \mathbb{R}^n$ is observed which follows an unknown probability density function (pdf) $p_d(.)$. The data pdf $p_d(.)$ is modeled by a parameterized family of functions $\{p_m(.;\alpha)\}_\alpha$ where α is a vector of parameters. We assume that $p_d(.)$ belongs to this family. In other words, $p_d(.) = p_m(.;\alpha^\star)$ for some parameter $\alpha^\star$. The problem we consider here is how to estimate α from the observed sample by maximizing some objective function.

Any solution $\hat{\alpha}$ to this estimation problem must yield a properly normalized density $p_m(.;\hat{\alpha})$ with

$$\int p_m(\mathbf{u};\hat{\alpha})d\mathbf{u} = 1. \qquad (1)$$

This defines essentially a constraint in the optimization problem.[1] In principle, the constraint can always be fulfilled by redefining the pdf as

$$p_m(.;\alpha) = \frac{p_m^0(.;\alpha)}{Z(\alpha)}, \quad Z(\alpha) = \int p_m^0(\mathbf{u};\alpha)d\mathbf{u}, \qquad (2)$$

where $p_m^0(.;\alpha)$ specifies the functional form of the pdf

Text Representation

Word2Vec

SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Skipgram

shalt	not	make	a	machine

input	output
make	shalt
make	not
make	a
make	machine

Negative Sampling

input word	output word	target
make	shalt	1
make	aaron	0
make	taco	0

Text Representation

Word2Vec

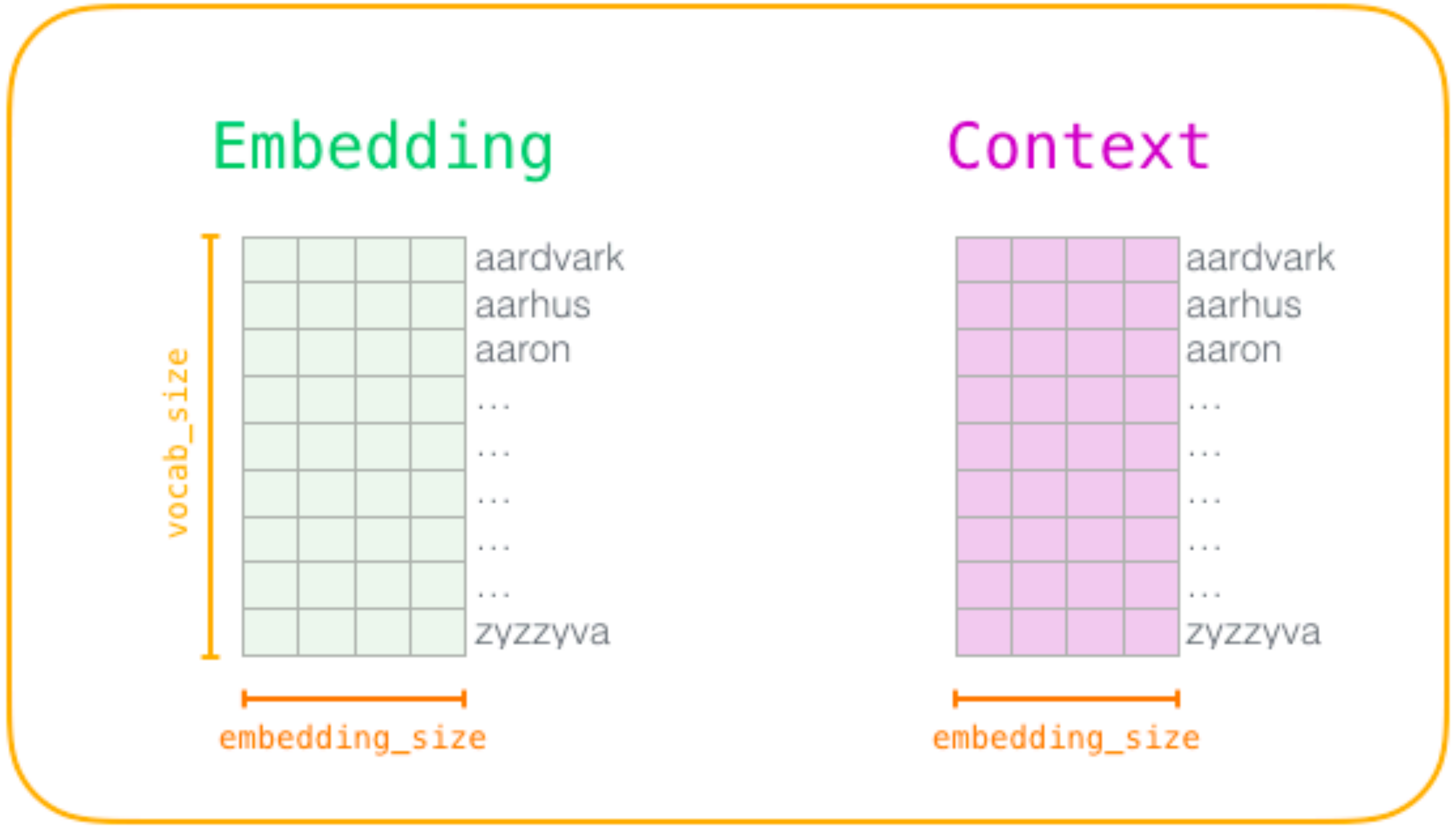
SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Training: *Definición de matrices con tamaño de vocabulario y tamaño de embedding.*



Tamaño de vocabulario: 10.000 palabras *Tamaño de embedding: 300 dimensiones*

Text Representation

Word2Vec

SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

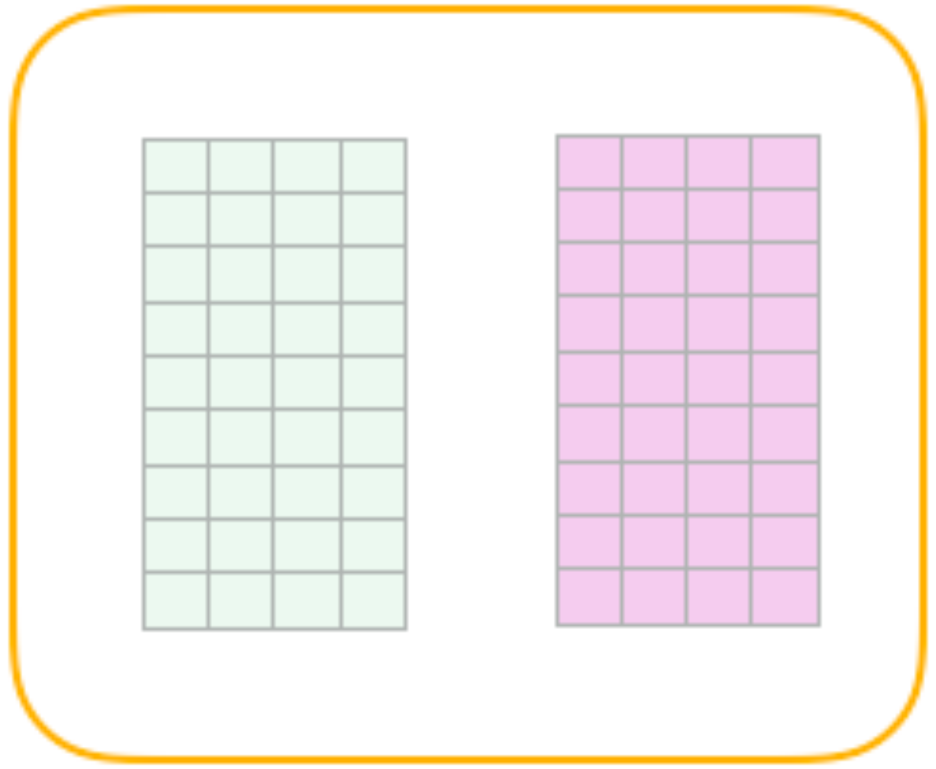
Pronóstico con base en la palabra actual.

Training: *Inicializar las matrices con valores aleatorios. Luego, se inicia el entrenamiento.*

dataset

input word	output word	target
not	thou	1
not	aaron	0
not	taco	0
not	shalt	1
not	mango	0
not	finglonger	0
not	make	1
not	plumbus	0
...	...	...

model



Text Representation

Word2Vec

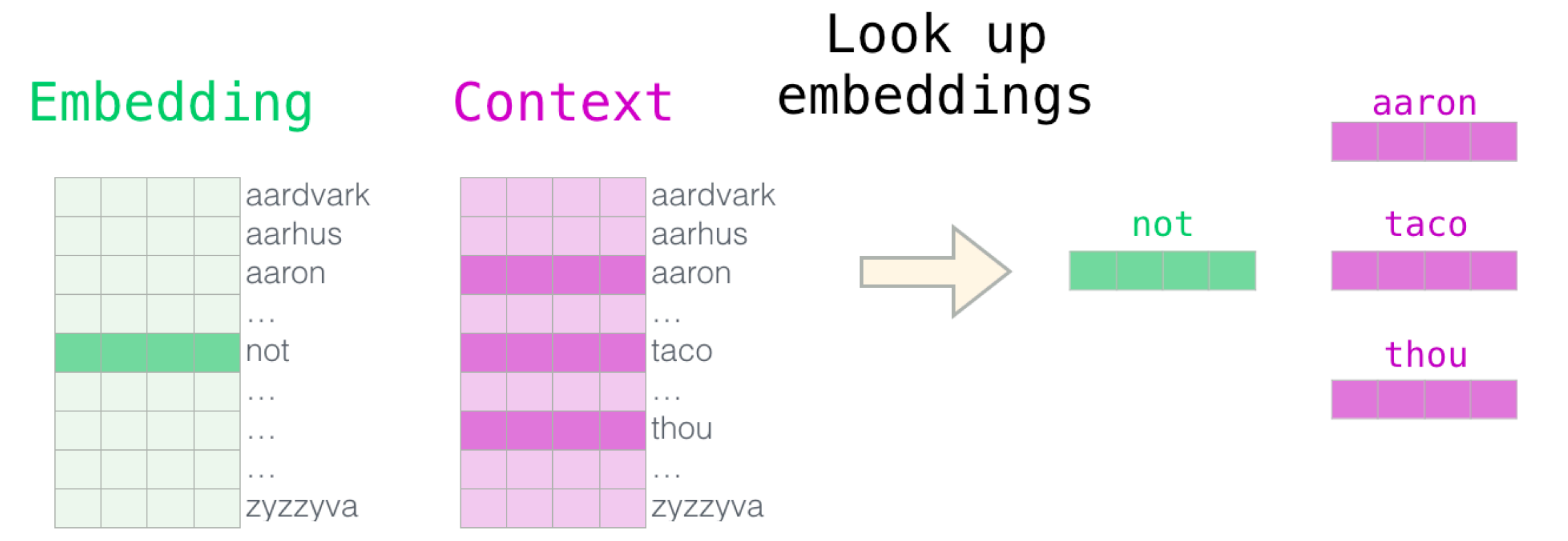
SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Training: *Buscar los embeddings en la matriz de embeddings y en la matriz de contexto.*



Text Representation

Word2Vec

SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Training: *Producto escalar de los embeddings de entrada con cada embedding de contexto.*

input word	output word	target	input • output
not	thou	1	0.2
not	aaron	0	-1.11
not	taco	0	0.74

Ahora debemos convertir estas puntuaciones en probabilidades. Positivas entre 0 y 1. Aplicamos una función sigmoide.

input word	output word	target	input • output	sigmoid()
not	thou	1	0.2	0.55
not	aaron	0	-1.11	0.25
not	taco	0	0.74	0.68

Text Representation

Word2Vec

SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Training: *Restamos las puntuaciones sigmoideas de las etiquetas objetivo/ target.*

input word	output word	target	input • output	sigmoid()	Error
not	thou	1	0.2	0.55	0.45
not	aaron	0	-1.11	0.25	-0.25
not	taco	0	0.74	0.68	-0.68

Text Representation

Word2Vec

SkipGram Negative Sampling: Language Modeling

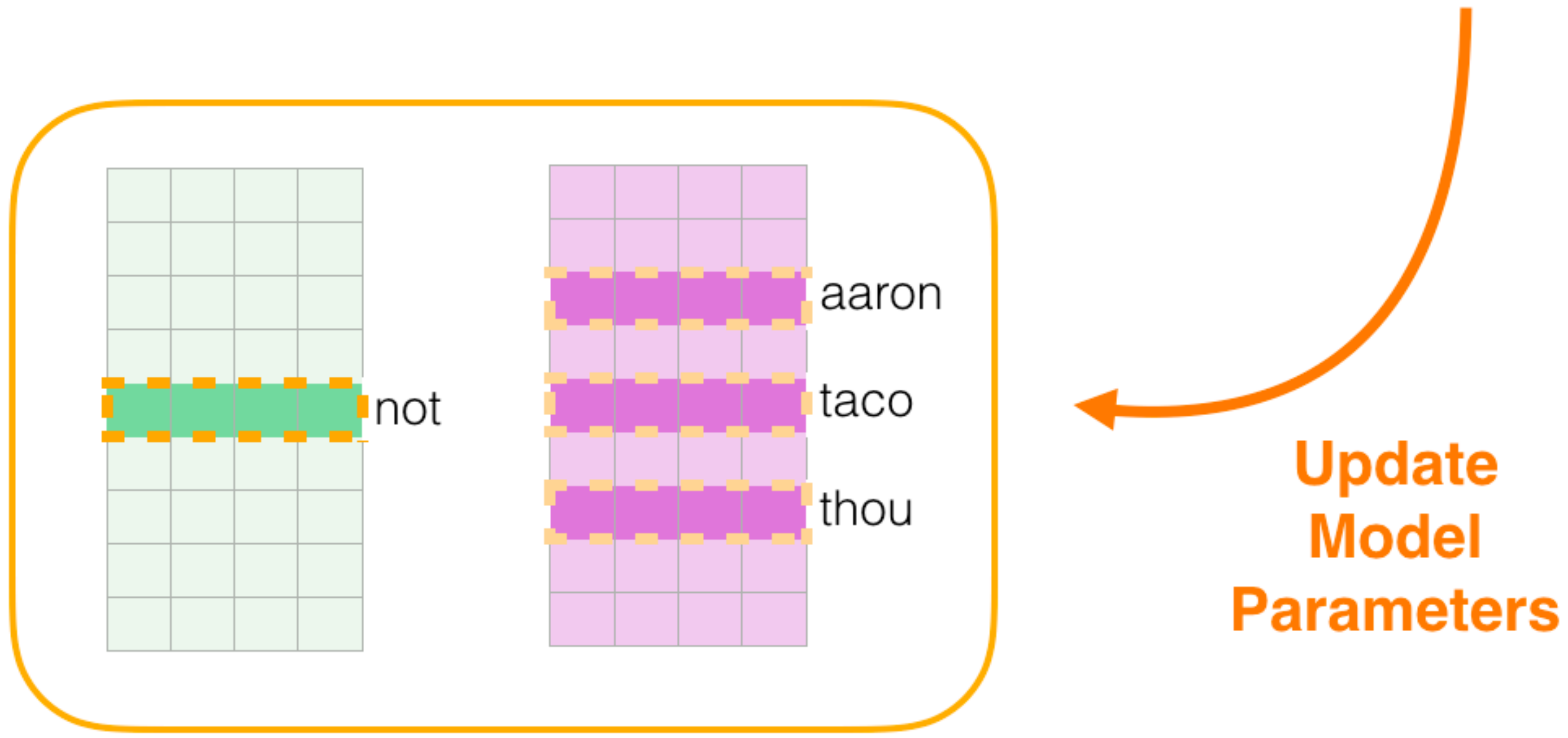
La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Training: *Actualizamos con los errores el modelo, proceso de "aprendizaje" en machine learning.*

input word	output word	target	input • output	sigmoid()	Error
not	thou	1	0.2	0.55	0.45
not	aaron	0	-1.11	0.25	-0.25
not	taco	0	0.74	0.68	-0.68



Text Representation

Word2Vec

SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

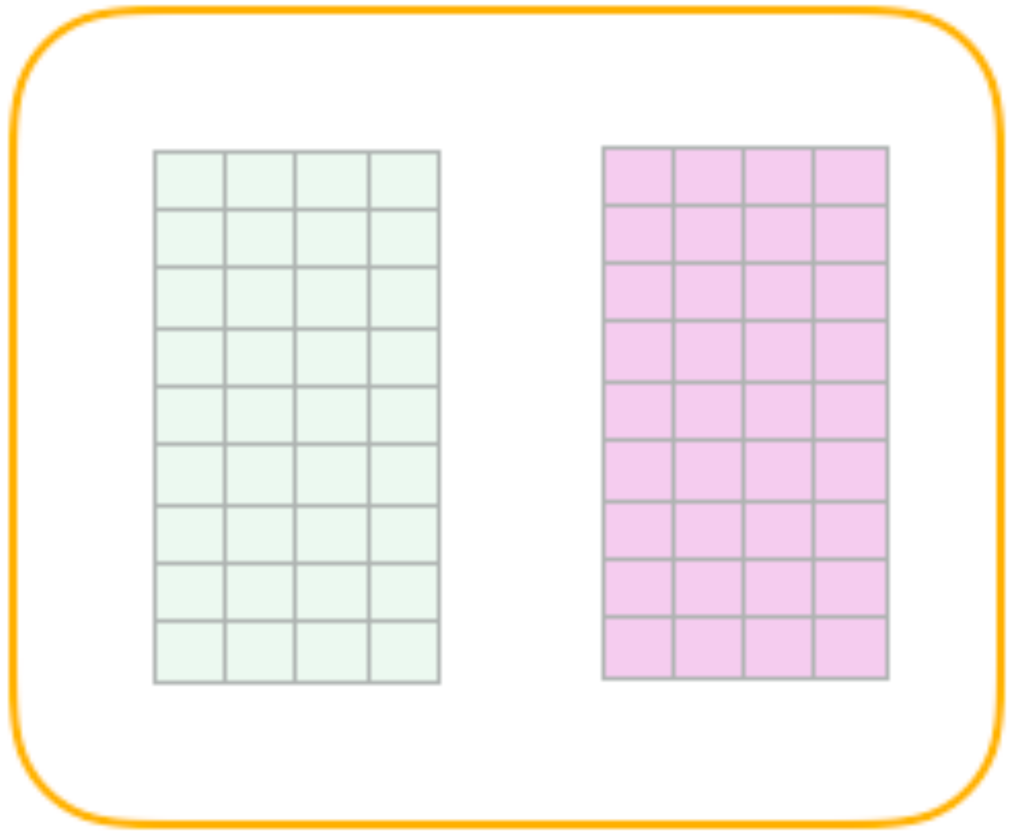
Pronóstico con base en la palabra actual.

Training: *Continuamos con la siguiente muestra de entrenamiento del conjunto de datos. Así, sucesivamente...*

dataset

input word	output word	target
not	thou	1
not	aaron	0
not	taco	0
not	shalt	1
not	mango	0
not	finglonger	0
not	make	1
not	plumbus	0
...	...	...

model



Text Representation

Word2Vec

SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Finalizando:

1. Las incorporaciones continúan mejorándose mientras recorremos todo nuestro conjunto de datos varias veces.

2. Al finalizar entrenamiento, debemos descartar la matriz de contexto y usar la matriz de embeddings (embeddings pre-entrenados)

Text Representation

Word2Vec

Hiperparámetros importantes

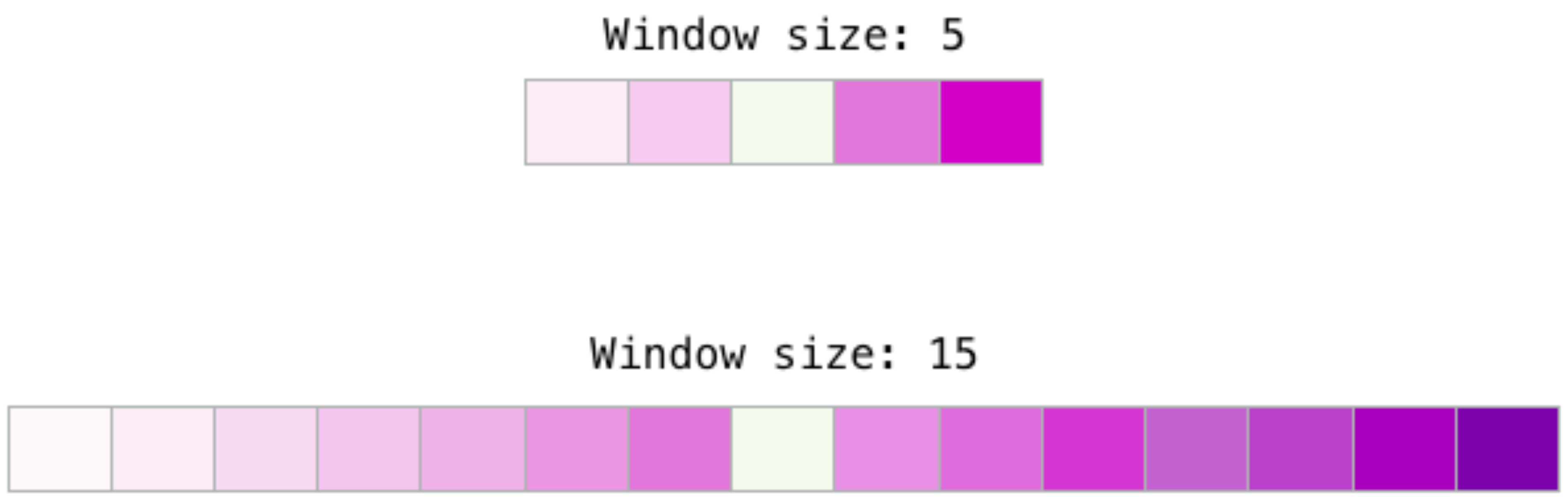
SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Tamaño de ventana:



1. Tamaño pequeños (2-15): score altos de similitud entre dos embeddings indica que las las palabras son intercambiables (antónimos: bueno y malo en contexto similares).

2. Tamaño más grande (15-50+): conducen a embeddings donde la similitud es más indicación de relación de las palabras.

Text Representation

Word2Vec

Hiperparámetros importantes

SkipGram Negative Sampling: Language Modeling

La predicción de la siguiente palabra.

↓

Pronóstico con base en la palabra actual.

Número de muestras negativas:

Negative samples: 2

input word	output word	target
make	shalt	1
make	aaron	0
make	taco	0

Negative samples: 5

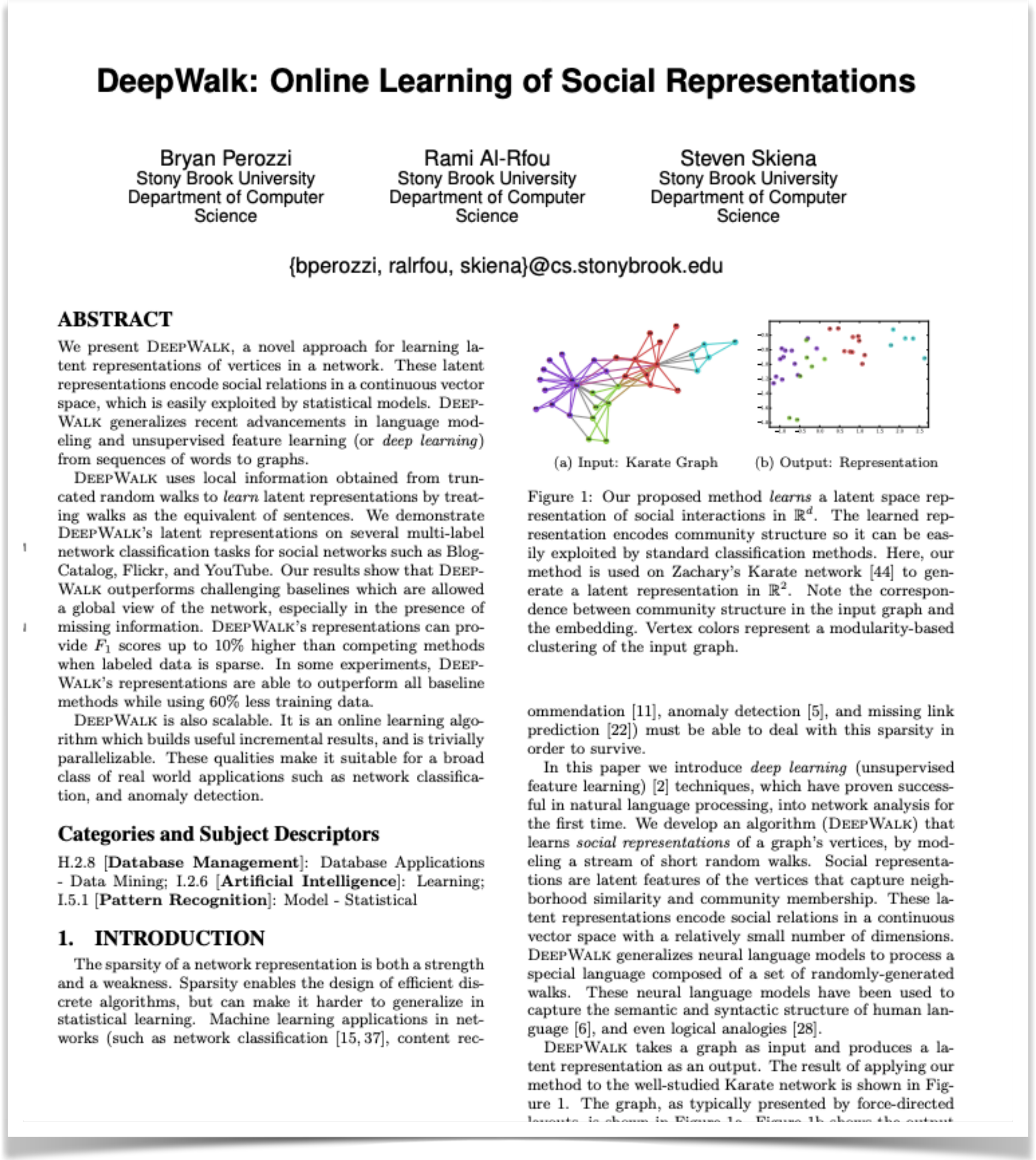
input word	output word	target
make	shalt	1
make	aaron	0
make	taco	0
make	finglonger	0
make	plumbus	0
make	mango	0

1. El artículo original prescribe que entre 5 y 20 son un buen número de muestras negativas.

2. También indica que 2-5 parece ser suficiente cuando se tiene un conjunto de datos lo suficientemente grande.

Graph Representation

Node2Vec



DeepWalk: Online Learning of Social Representations

Bryan Perozzi
Stony Brook University
Department of Computer Science

Rami Al-Rfou
Stony Brook University
Department of Computer Science

Steven Skiena
Stony Brook University
Department of Computer Science

{bperozzi, ralrfou, skiena}@cs.stonybrook.edu

ABSTRACT

We present DeepWalk, a novel approach for learning latent representations of vertices in a network. These latent representations encode social relations in a continuous vector space, which is easily exploited by statistical models. DeepWalk generalizes recent advancements in language modeling and unsupervised feature learning (or *deep learning*) from sequences of words to graphs.

DeepWalk uses local information obtained from truncated random walks to *learn* latent representations by treating walks as the equivalent of sentences. We demonstrate DeepWalk's latent representations on several multi-label network classification tasks for social networks such as BlogCatalog, Flickr, and YouTube. Our results show that DeepWalk outperforms challenging baselines which are allowed a global view of the network, especially in the presence of missing information. DeepWalk's representations can provide F_1 scores up to 10% higher than competing methods when labeled data is sparse. In some experiments, DeepWalk's representations are able to outperform all baseline methods while using 60% less training data.

DeepWalk is also scalable. It is an online learning algorithm which builds useful incremental results, and is trivially parallelizable. These qualities make it suitable for a broad class of real world applications such as network classification, and anomaly detection.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications - Data Mining; I.2.6 [**Artificial Intelligence**]: Learning; I.5.1 [**Pattern Recognition**]: Model - Statistical

1. INTRODUCTION

The sparsity of a network representation is both a strength and a weakness. Sparsity enables the design of efficient discrete algorithms, but can make it harder to generalize in statistical learning. Machine learning applications in networks (such as network classification [15, 37], content recommendation [11], anomaly detection [5], and missing link prediction [22]) must be able to deal with this sparsity in order to survive.

In this paper we introduce *deep learning* (unsupervised feature learning) [2] techniques, which have proven successful in natural language processing, into network analysis for the first time. We develop an algorithm (DeepWalk) that learns *social representations* of a graph's vertices, by modeling a stream of short random walks. Social representations are latent features of the vertices that capture neighborhood similarity and community membership. These latent representations encode social relations in a continuous vector space with a relatively small number of dimensions. DeepWalk generalizes neural language models to process a special language composed of a set of randomly-generated walks. These neural language models have been used to capture the semantic and syntactic structure of human language [6], and even logical analogies [28].

DeepWalk takes a graph as input and produces a latent representation as an output. The result of applying our method to the well-studied Karate network is shown in Figure 1. The graph, as typically presented by force-directed

(a) Input: Karate Graph (b) Output: Representation

Figure 1: Our proposed method *learns* a latent space representation of social interactions in $\mathbb{R}^d$. The learned representation encodes community structure so it can be easily exploited by standard classification methods. Here, our method is used on Zachary's Karate network [44] to generate a latent representation in $\mathbb{R}^2$. Note the correspondence between community structure in the input graph and the embedding. Vertex colors represent a modularity-based clustering of the input graph.

Deep Walk

producir representaciones de características de alta calidad de nodos.

Word2Vec → **Deep Walk + Bias** → **Node2Vec**

Graph Representation

Node2Vec

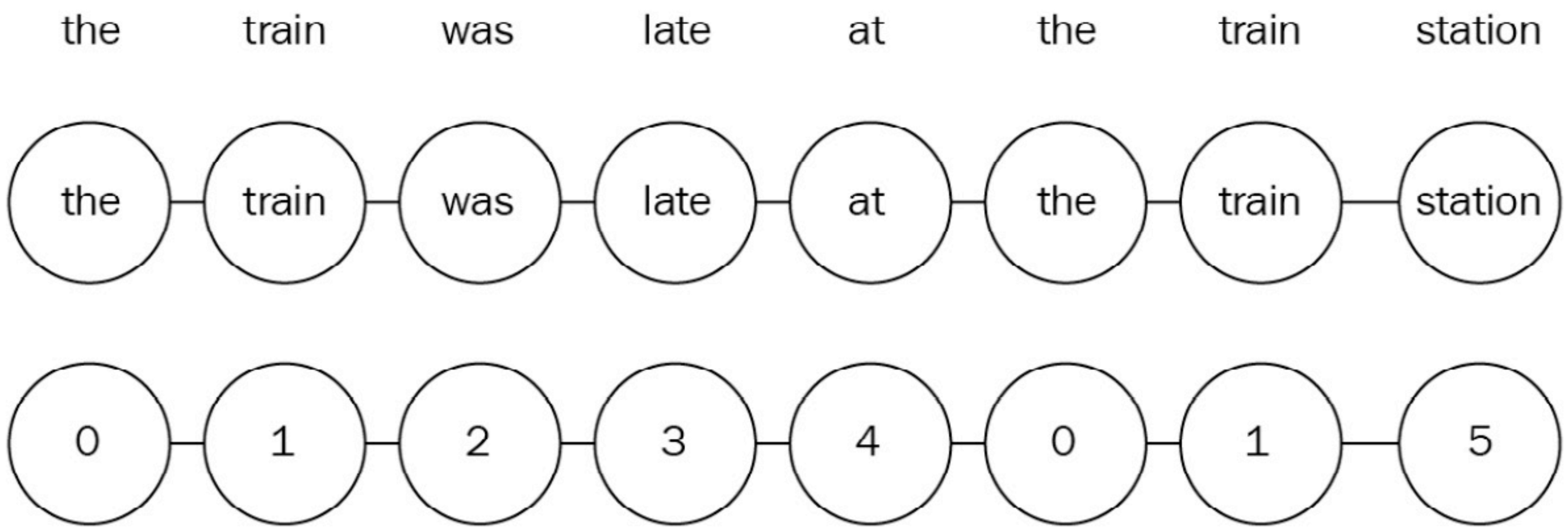
Deep Walk

producir representaciones de características de alta calidad de nodos.

Word2Vec

Node2Vec

Random Walks:



Graph Representation

Node2Vec

Deep Walk

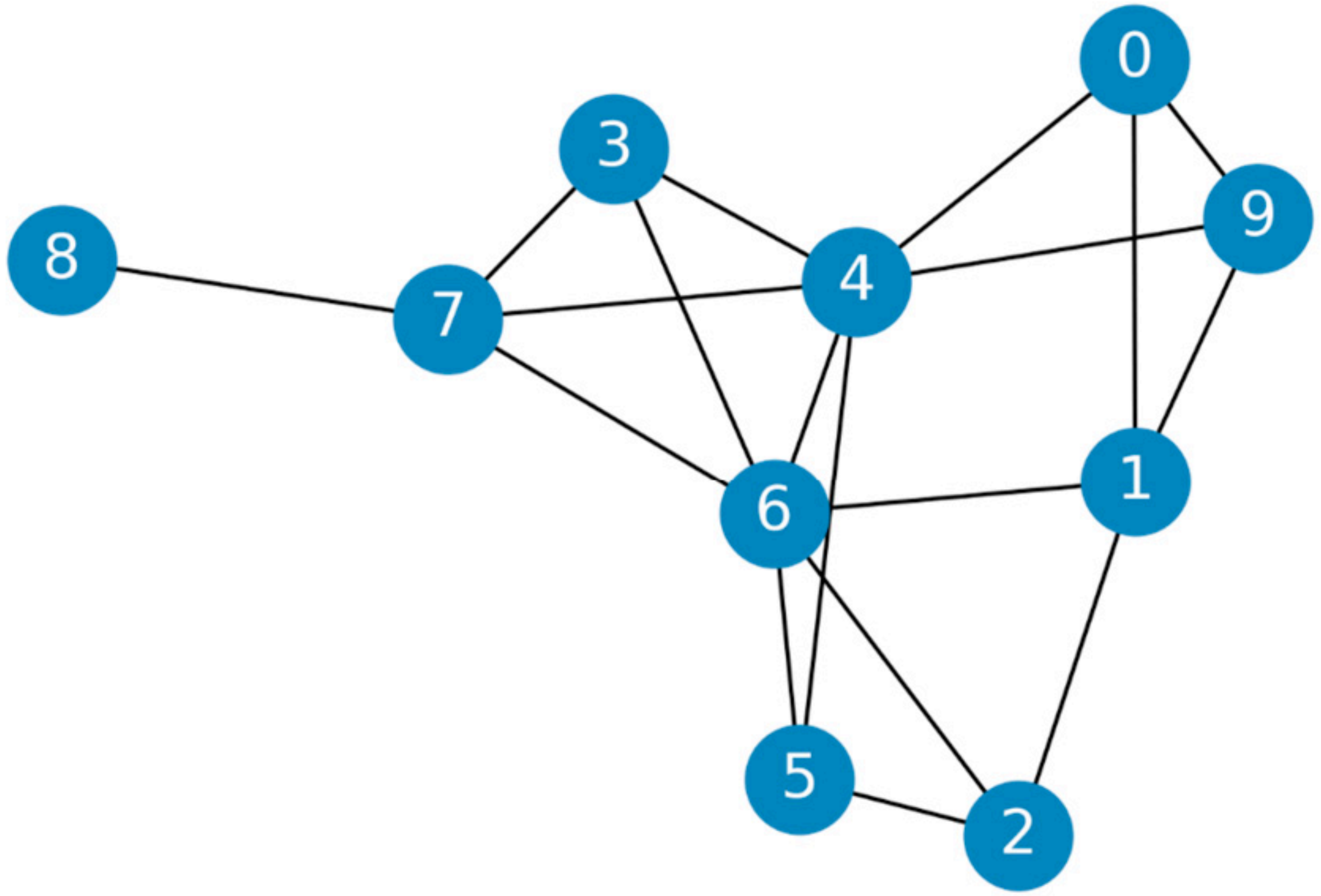
producir representaciones de características de alta calidad de nodos.

Word2Vec

↓

Node2Vec

Random Walks:



Hipótesis de homofilia:

Los nodos que están más cerca uno del otro son similares.

Graph Representation

Deep Walk and random walks

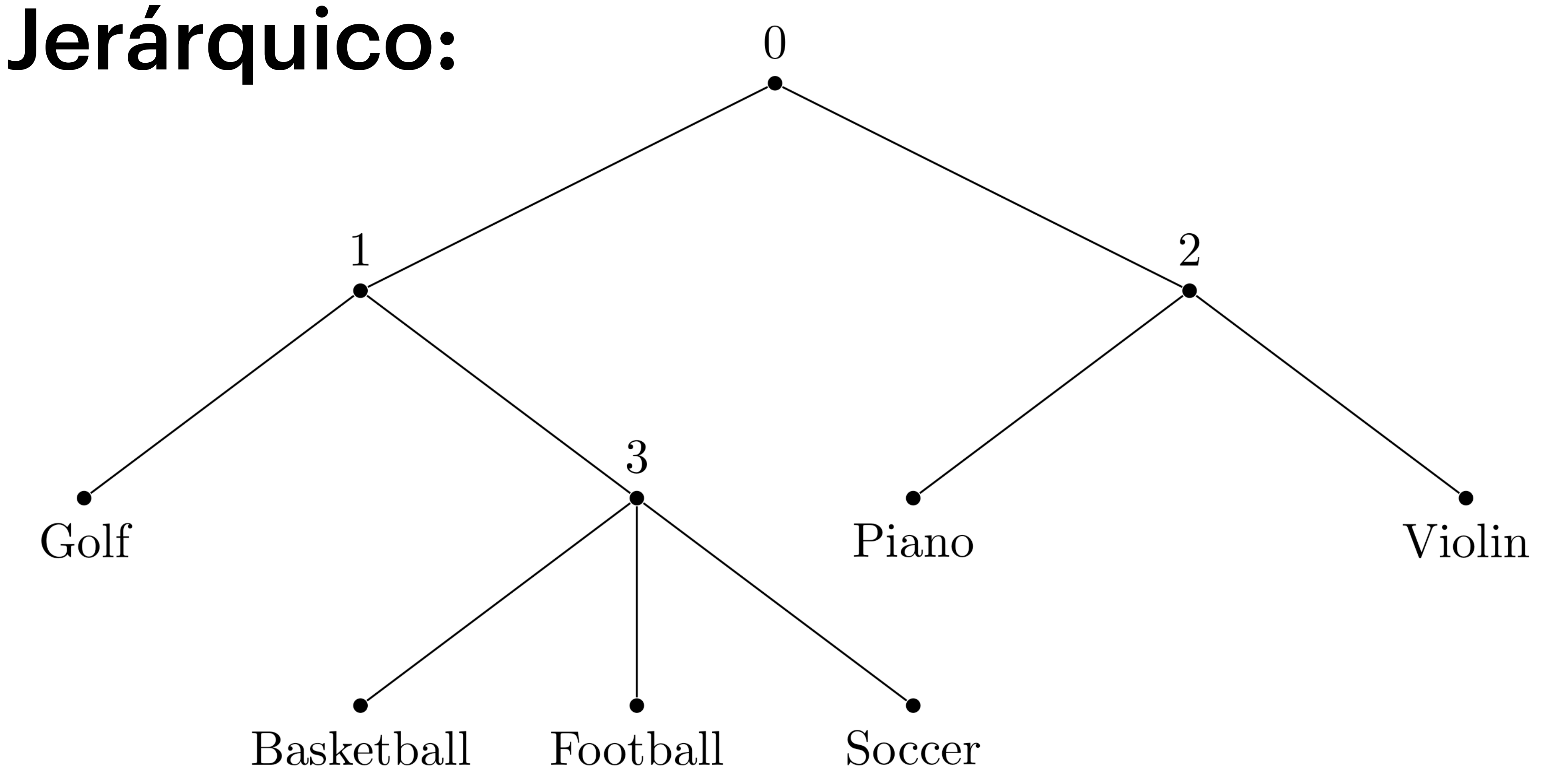
Deep Walk

producir representaciones de características de alta calidad de nodos.

Word2Vec

Node2Vec

Softmax Jerárquico:



$$P^h_\theta(w = \text{Football}) = P^h_\theta(0 \to 1)P^h_\theta(1 \to 3)P^h_\theta(3 \to \text{Football})$$

$$P^h_\theta(w) = \prod_{j=1}^{L(w)-1} P^h_{\theta_j}(n(w,j) \to n(w,j+1))$$

Graph Representation

Deep Walk and random walks

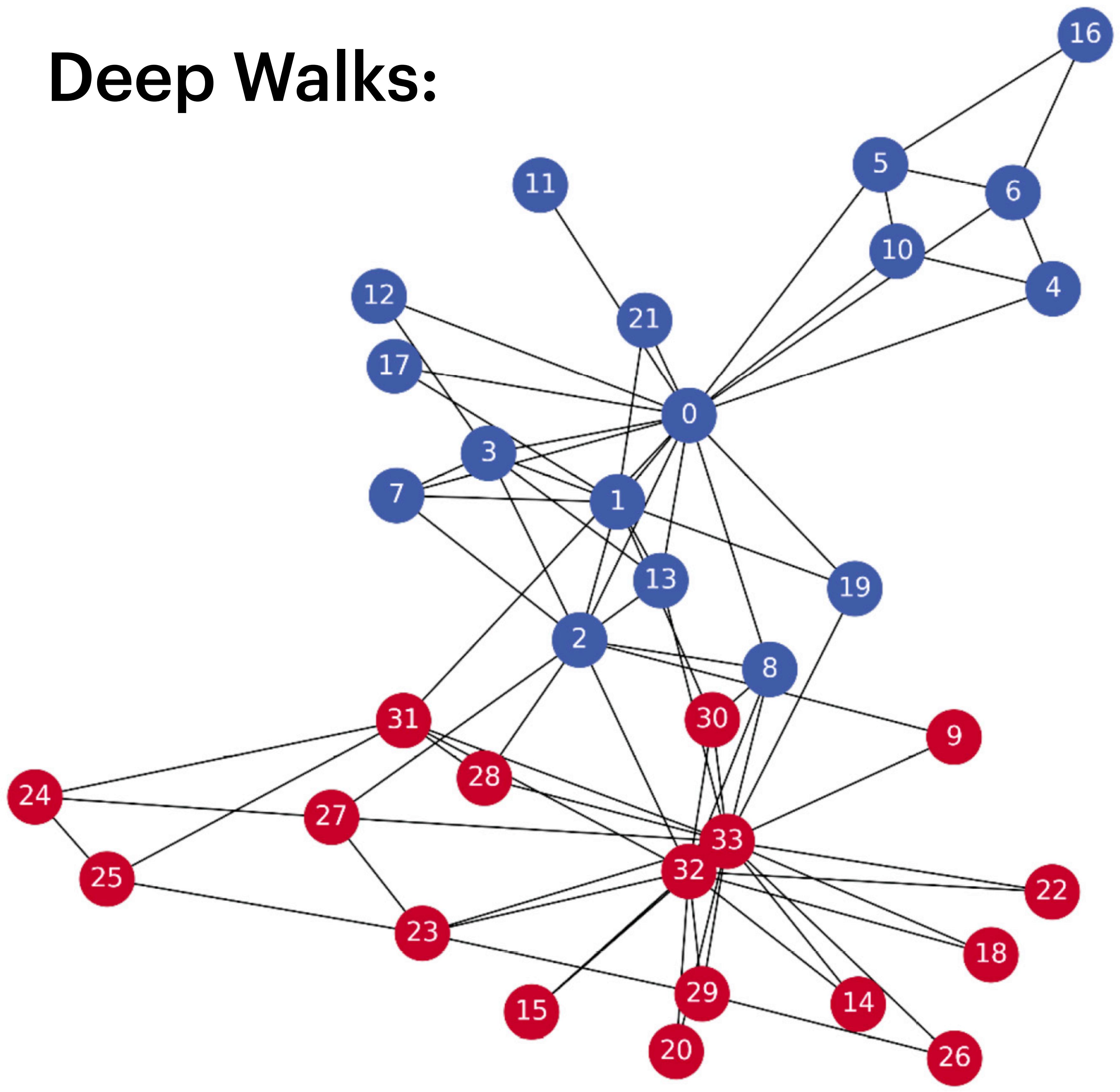
Deep Walk

producir representaciones de características de alta calidad de nodos.

Word2Vec

Node2Vec

Deep Walks:



- *Random walks*
- *Softmax Jerárquico*
- *Skipgram (Word2Vec)*

Graph Representation

Deep Walk and random walks

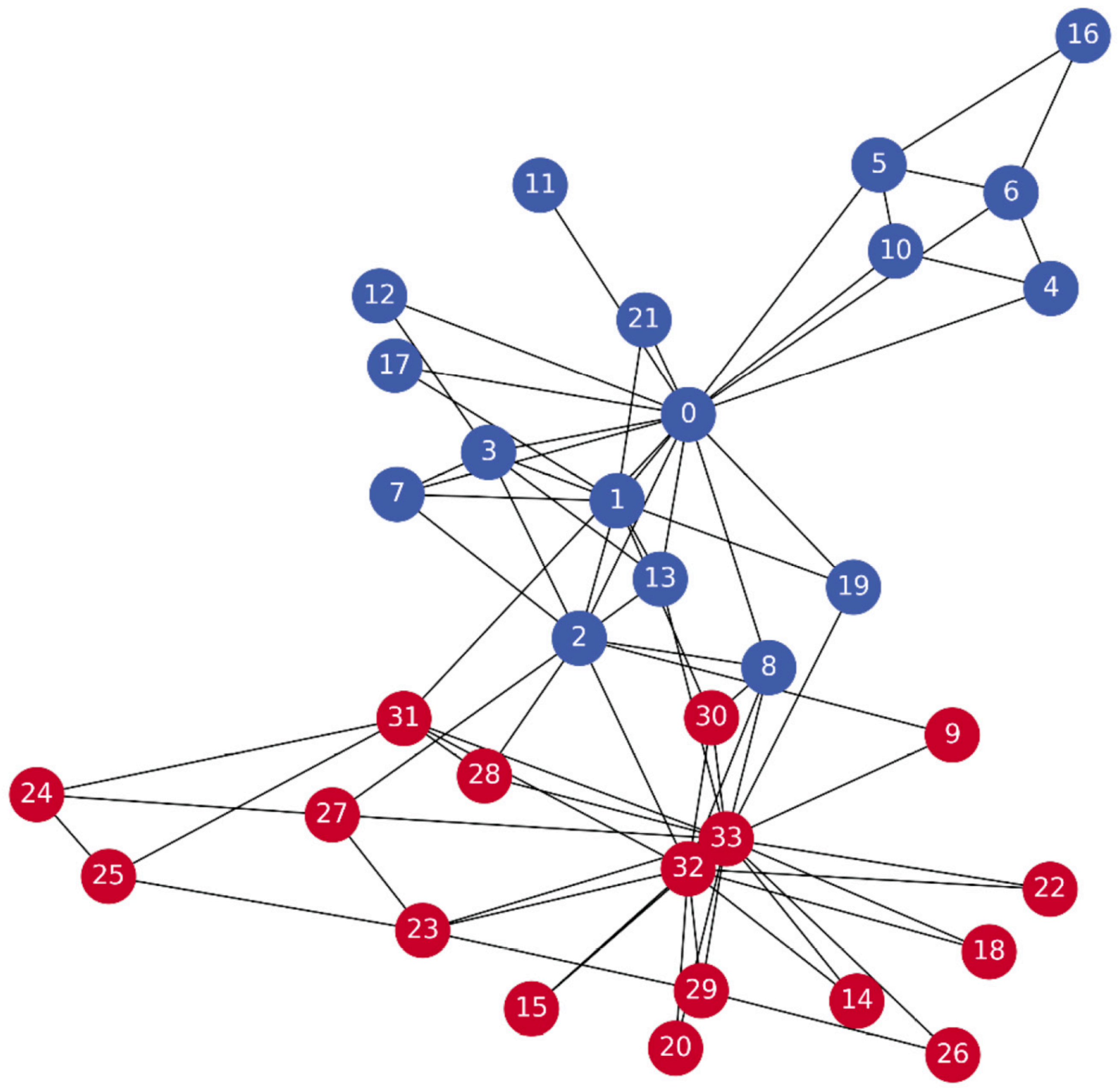
Deep Walk

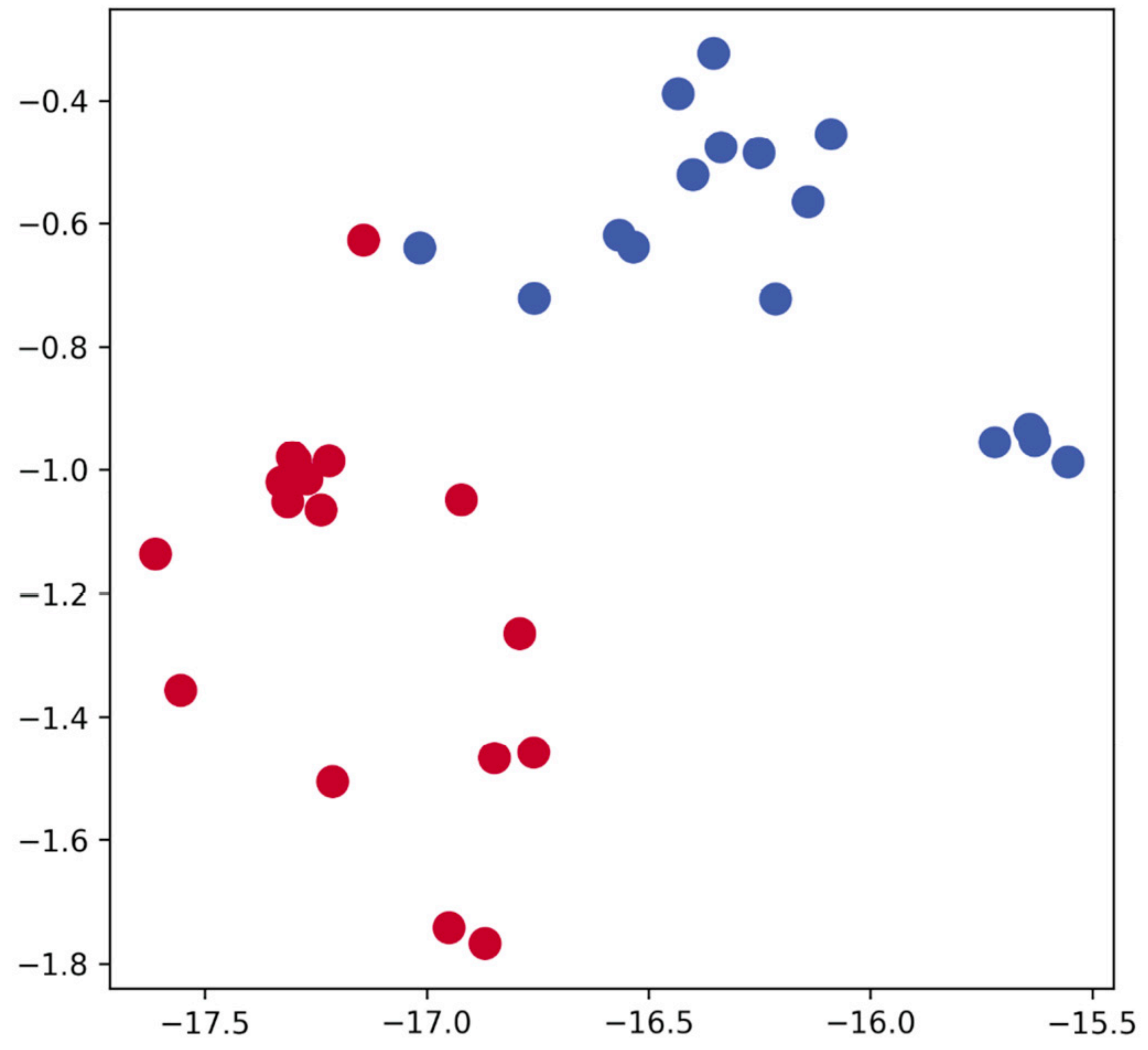
producir representaciones de características de alta calidad de nodos.

Word2Vec

Node2Vec

Deep Walks:





Graph Representation

Node2Vec

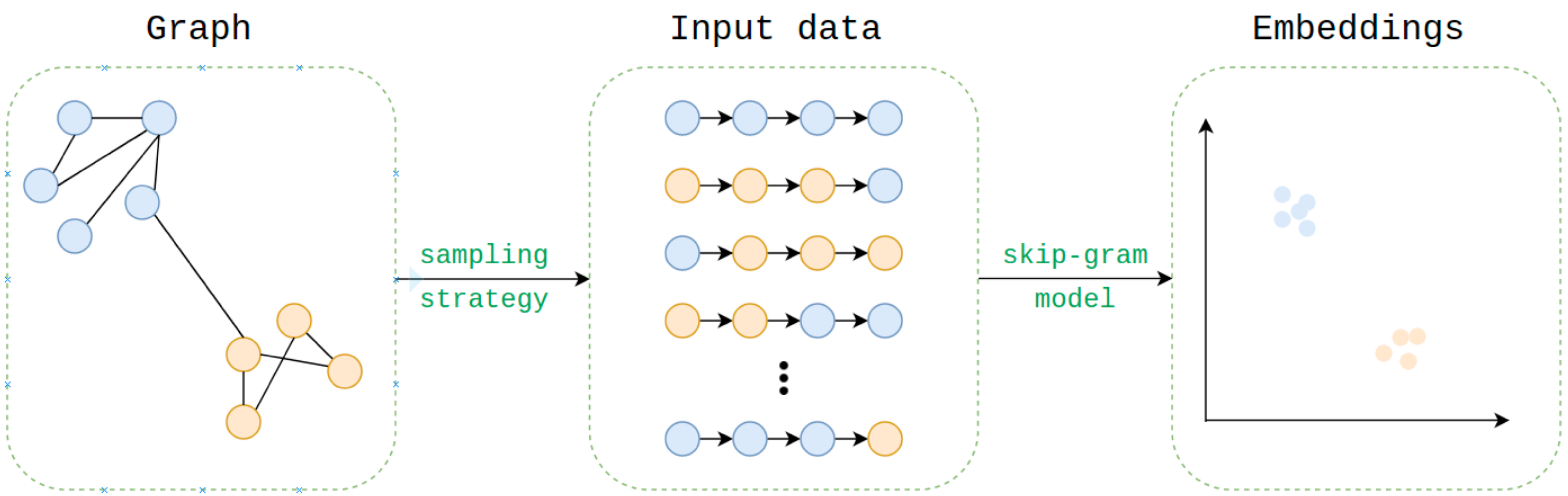
Deep Walk

producir representaciones de características de alta calidad de nodos.

Word2Vec

Node2Vec

Process:



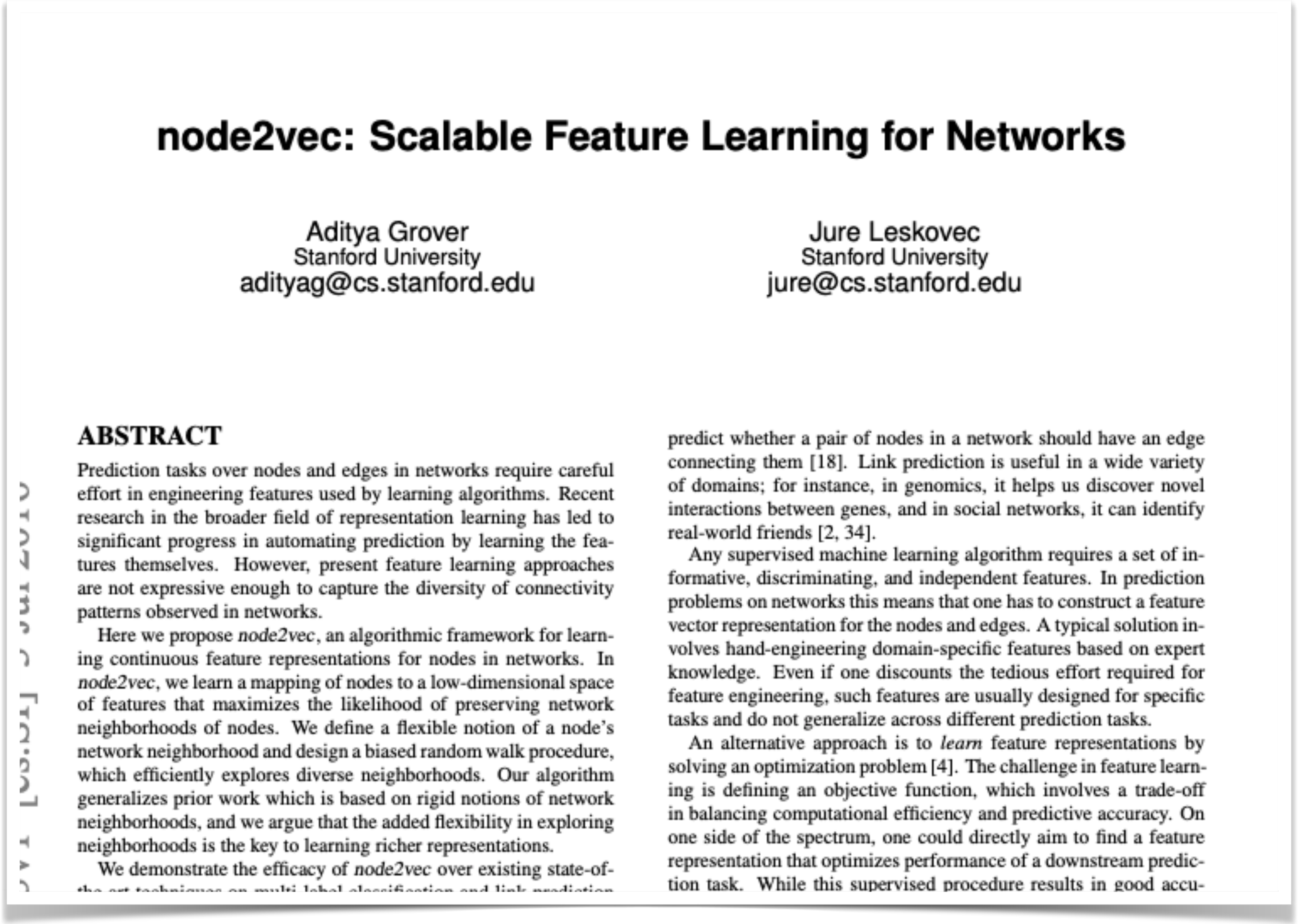
Graph Representation

Node2Vec

Word2Vec

Node2Vec

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.



node2vec: Scalable Feature Learning for Networks

Aditya Grover
Stanford University
adityag@cs.stanford.edu

Jure Leskovec
Stanford University
jure@cs.stanford.edu

ABSTRACT

Prediction tasks over nodes and edges in networks require careful effort in engineering features used by learning algorithms. Recent research in the broader field of representation learning has led to significant progress in automating prediction by learning the features themselves. However, present feature learning approaches are not expressive enough to capture the diversity of connectivity patterns observed in networks.

Here we propose *node2vec*, an algorithmic framework for learning continuous feature representations for nodes in networks. In *node2vec*, we learn a mapping of nodes to a low-dimensional space of features that maximizes the likelihood of preserving network neighborhoods of nodes. We define a flexible notion of a node's network neighborhood and design a biased random walk procedure, which efficiently explores diverse neighborhoods. Our algorithm generalizes prior work which is based on rigid notions of network neighborhoods, and we argue that the added flexibility in exploring neighborhoods is the key to learning richer representations.

We demonstrate the efficacy of *node2vec* over existing state-of-

predict whether a pair of nodes in a network should have an edge connecting them [18]. Link prediction is useful in a wide variety of domains; for instance, in genomics, it helps us discover novel interactions between genes, and in social networks, it can identify real-world friends [2, 34].

Any supervised machine learning algorithm requires a set of informative, discriminating, and independent features. In prediction problems on networks this means that one has to construct a feature vector representation for the nodes and edges. A typical solution involves hand-engineering domain-specific features based on expert knowledge. Even if one discounts the tedious effort required for feature engineering, such features are usually designed for specific tasks and do not generalize across different prediction tasks.

An alternative approach is to *learn* feature representations by solving an optimization problem [4]. The challenge in feature learning is defining an objective function, which involves a trade-off in balancing computational efficiency and predictive accuracy. On one side of the spectrum, one could directly aim to find a feature representation that optimizes performance of a downstream prediction task. While this supervised procedure results in good accu-

A. Grover y J. Leskovec, node2vec: Aprendizaje de funciones escalables para redes . arXiv, 2016. DOI: 10.48550/ARXIV.1607.00653.

Graph Representation

Node2Vec

Step 1:

Definir un vecindario.

Step 2:

Introducción de sesgos en paseos aleatorios.

Graph Representation

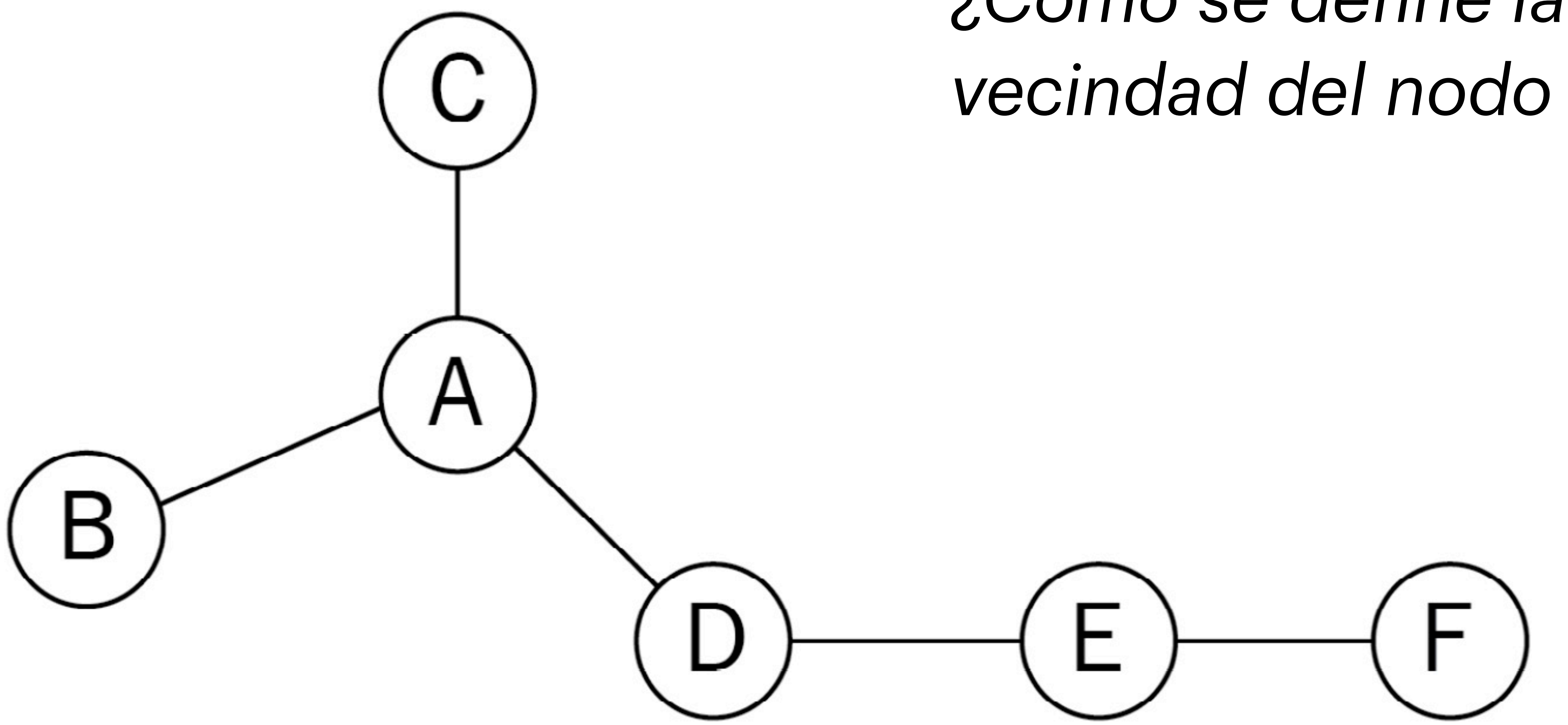
Node2Vec

Neighborhood

Queremos explorar tres nodos en la vecindad del nodo A.

↓

Estrategia de muestreo.



¿Cómo se define la vecindad del nodo A?

Graph Representation

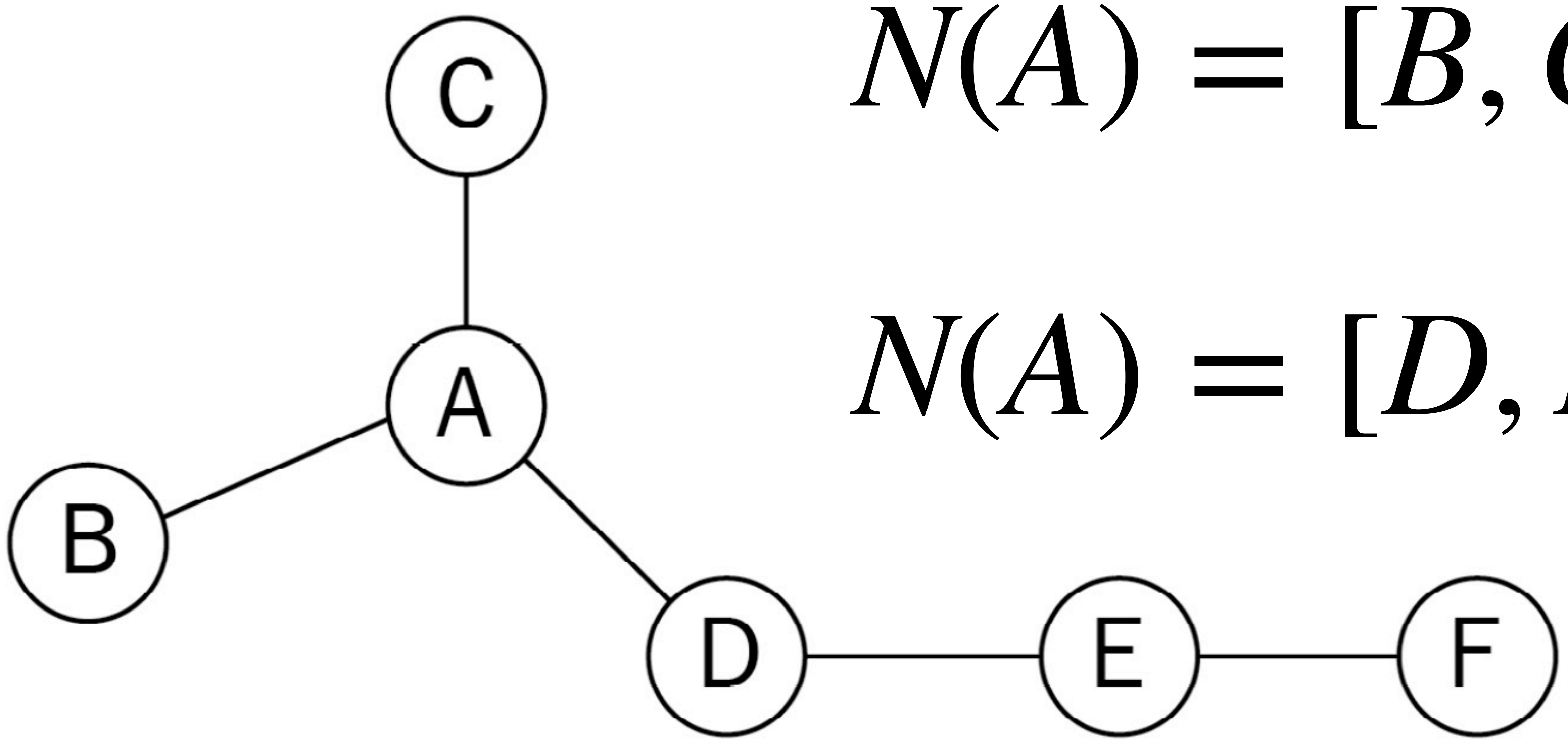
Node2Vec

Neighborhood

Queremos explorar tres nodos en la vecindad del nodo A.

↓

Estrategia de muestreo.



$$N(A) = [B, C, D]$$

$$N(A) = [D, E, F]$$

Graph Representation

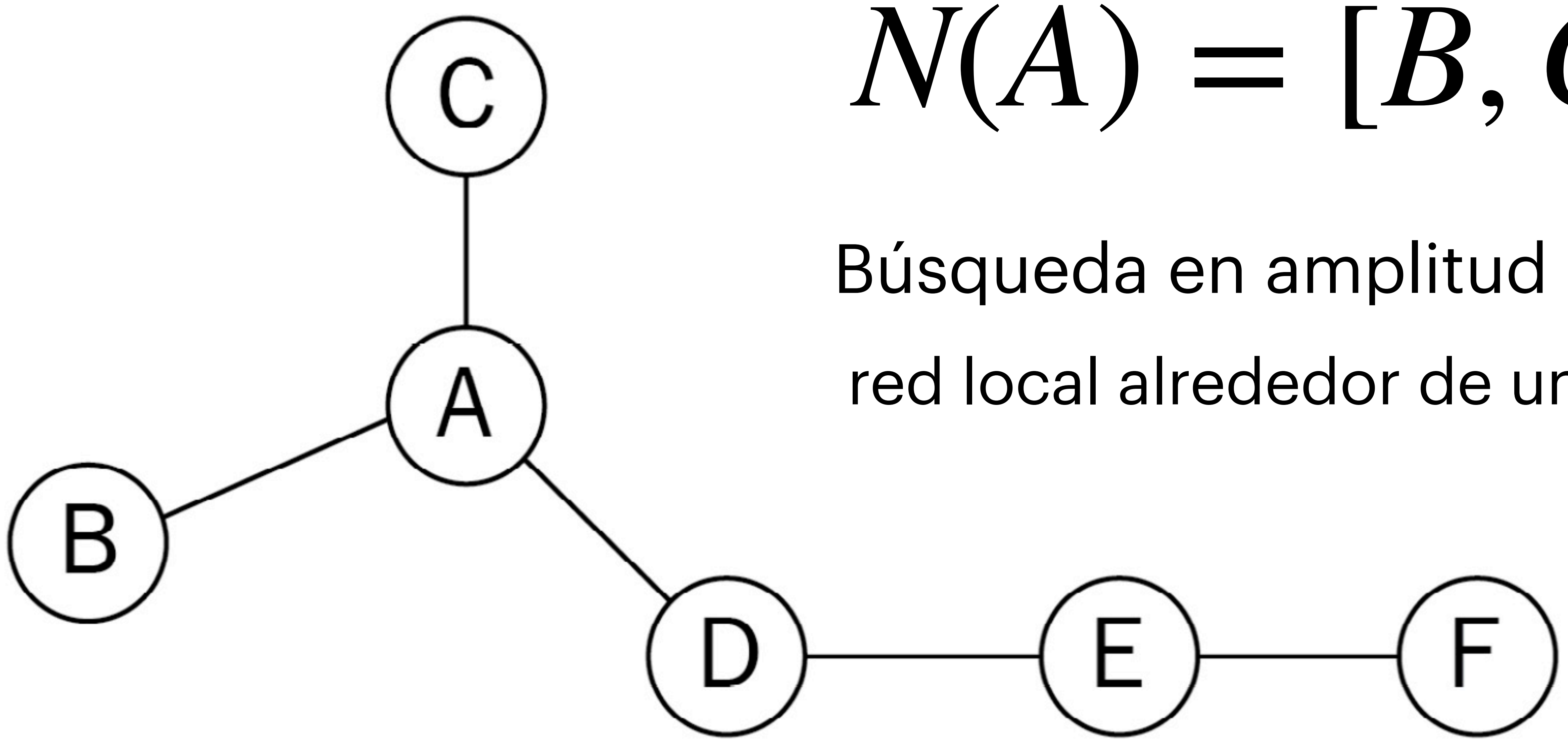
Node2Vec

Neighborhood

Queremos explorar tres nodos en la vecindad del nodo A.

↓

Estrategia de muestreo.



$$N(A) = [B, C, D]$$

Búsqueda en amplitud (BFS)

red local alrededor de un nodo

Graph Representation

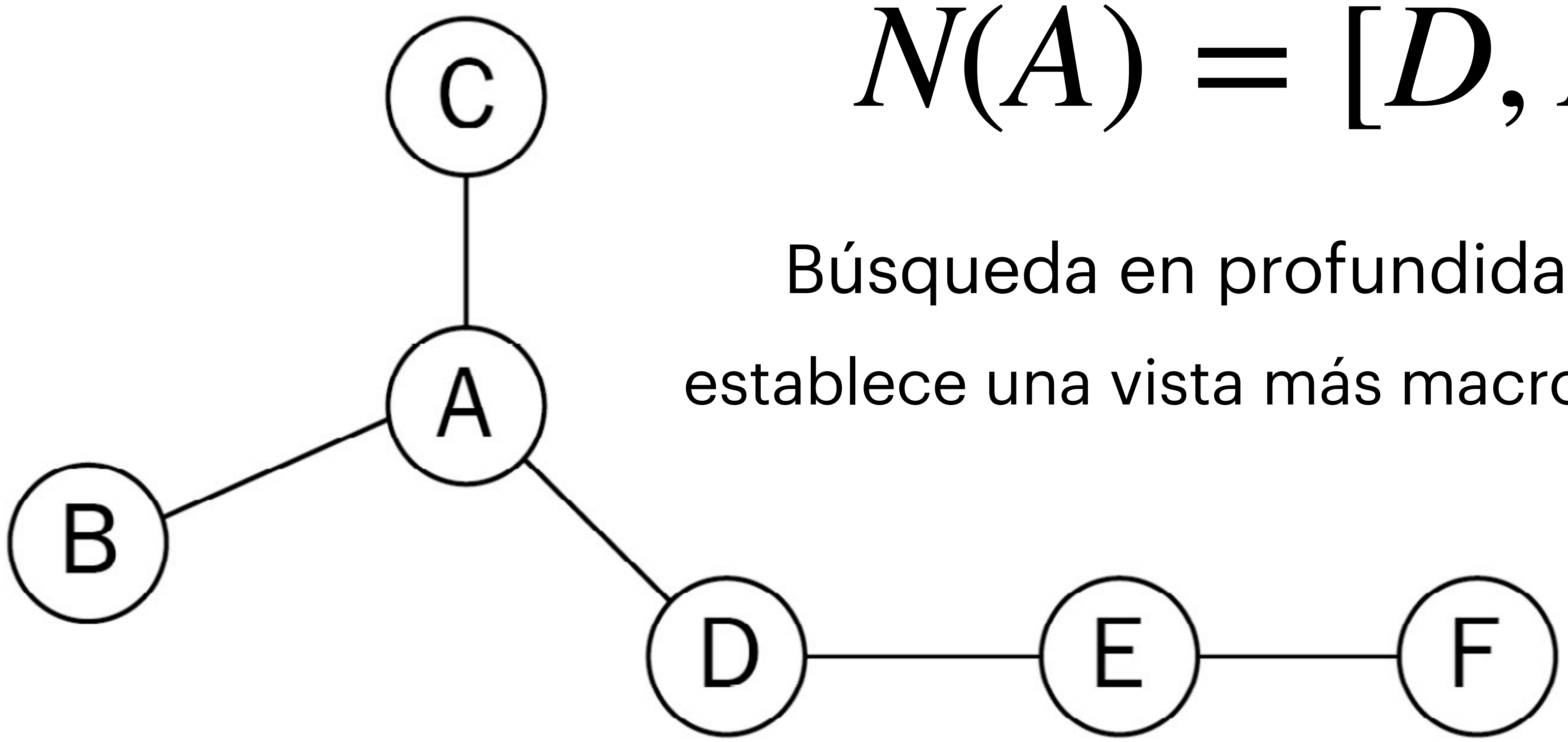
Node2Vec

Neighborhood

Queremos explorar tres nodos en la vecindad del nodo A.

↓

Estrategia de muestreo.



$$N(A) = [D, E, F]$$

Búsqueda en profundidad (DFS)

establece una vista más macro del gráfico

Graph Representation

Node2Vec

Neighborhood

Queremos explorar tres nodos en la vecindad del nodo A.

Estrategia de muestreo.

- Búsqueda en amplitud (BFS)

Solo mira nodos vecinos. En paseos aleatorios, los suelen repetirse y permanecer cerca.

1. Equivalencia estructural
2. La homofilia

- Búsqueda en profundidad (DFS)

Estos paseos aleatorios pueden muestrear nodos que están lejos de la fuente y, por lo tanto, volverse menos representativos.

Graph Representation

Node2Vec

Step 1:

Definir un vecindario.

Step 2:

Introducción de sesgos en paseos aleatorios.

Graph Representation

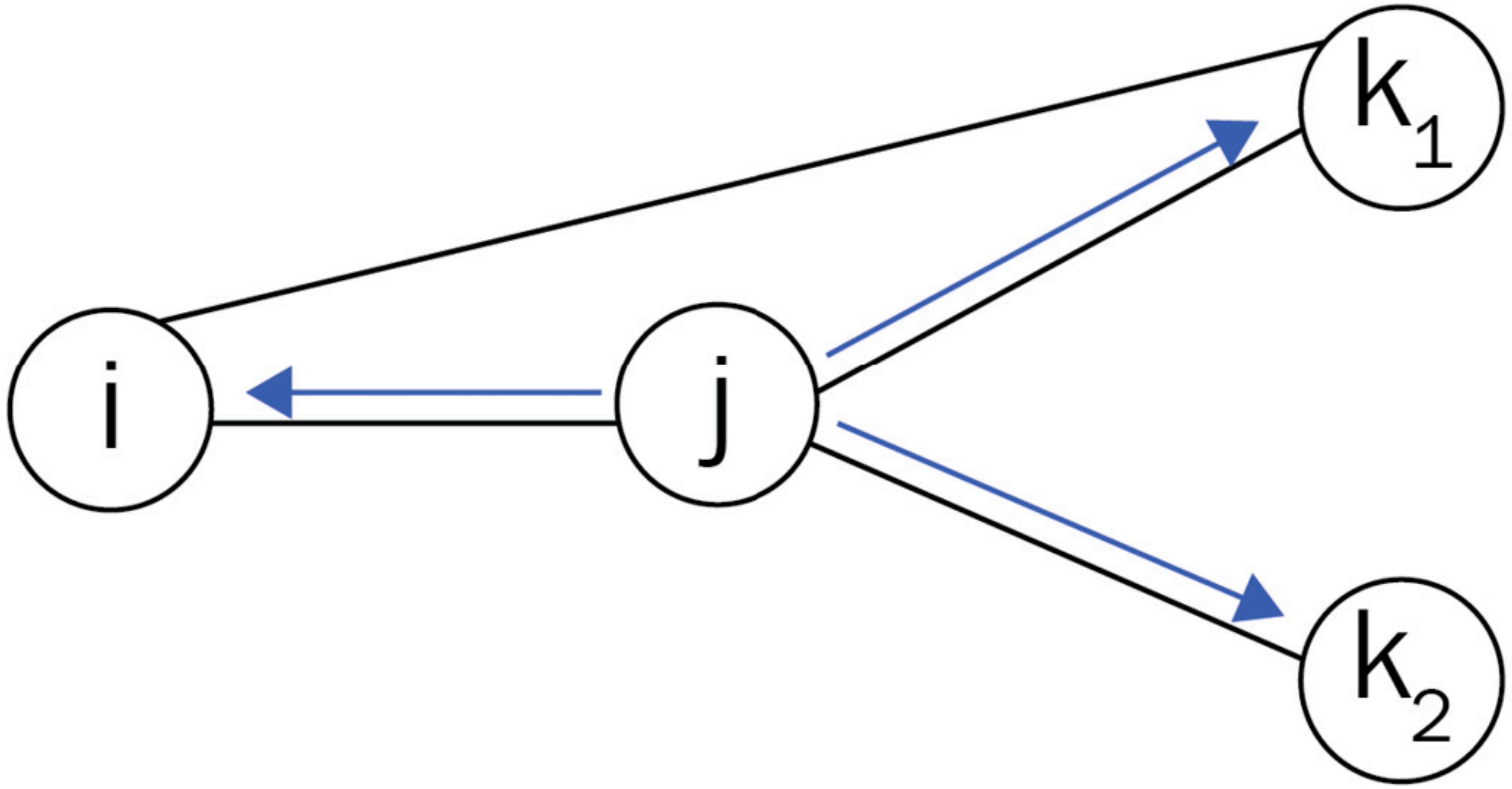
Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.



- Promocionar nodos que no están conectados al anterior (similar a DFS)
- Promoción de nodos cercanos al nodo anterior (similar a BFS)

Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

↓

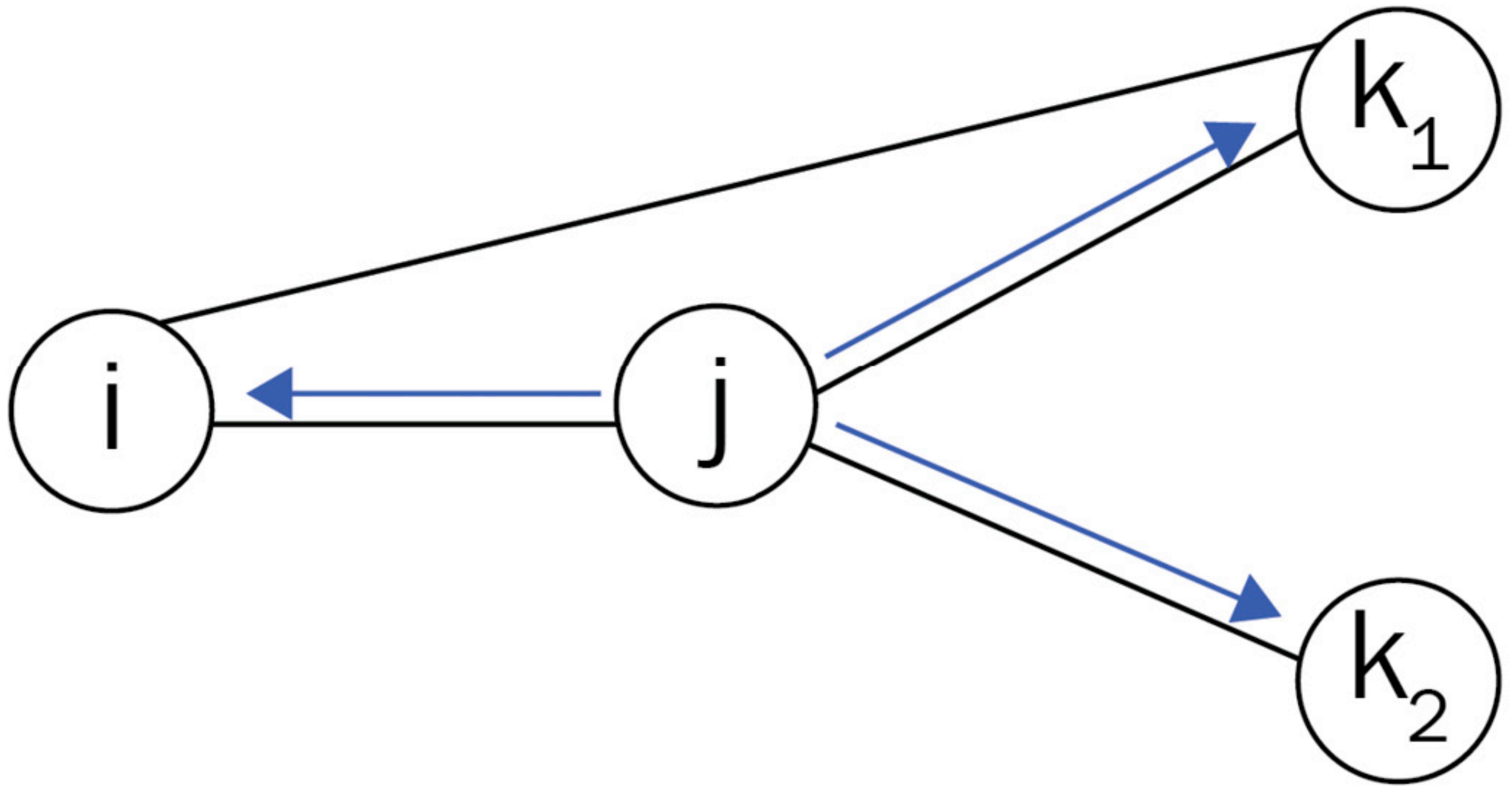
Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

i *Nodo actual*

j *Nodo anterior*

k *Nodo futuro*

π_{jk} *Probabilidad de transición no normalizada de un nodo* j *al nodo* k



Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

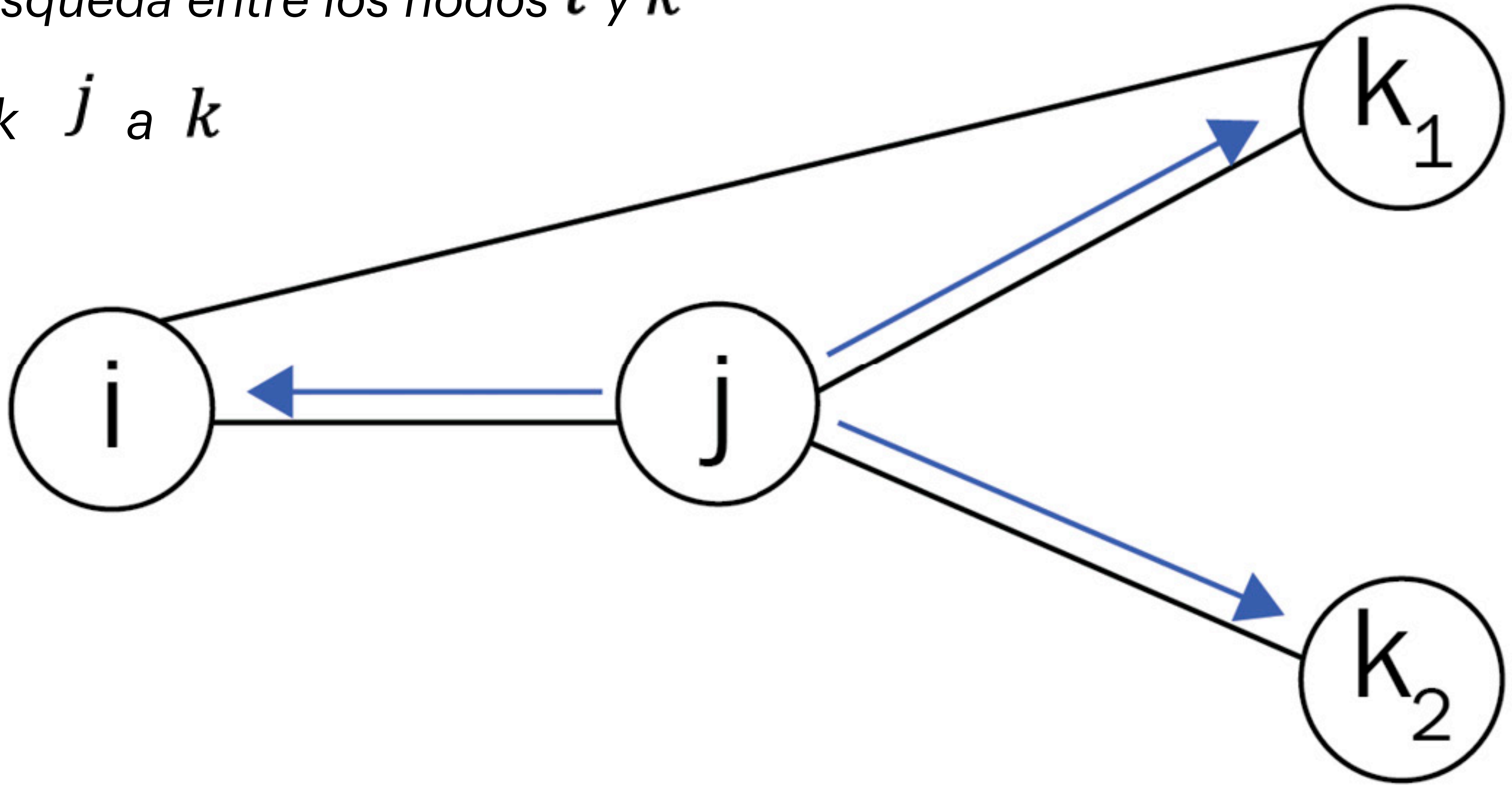
↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

$$\pi_{jk} = \alpha(i,k) \cdot \omega_{jk}$$

$\alpha(i,k)$ Sesgo de búsqueda entre los nodos i y k

ω_{jk} Peso del link j a k



Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

↓

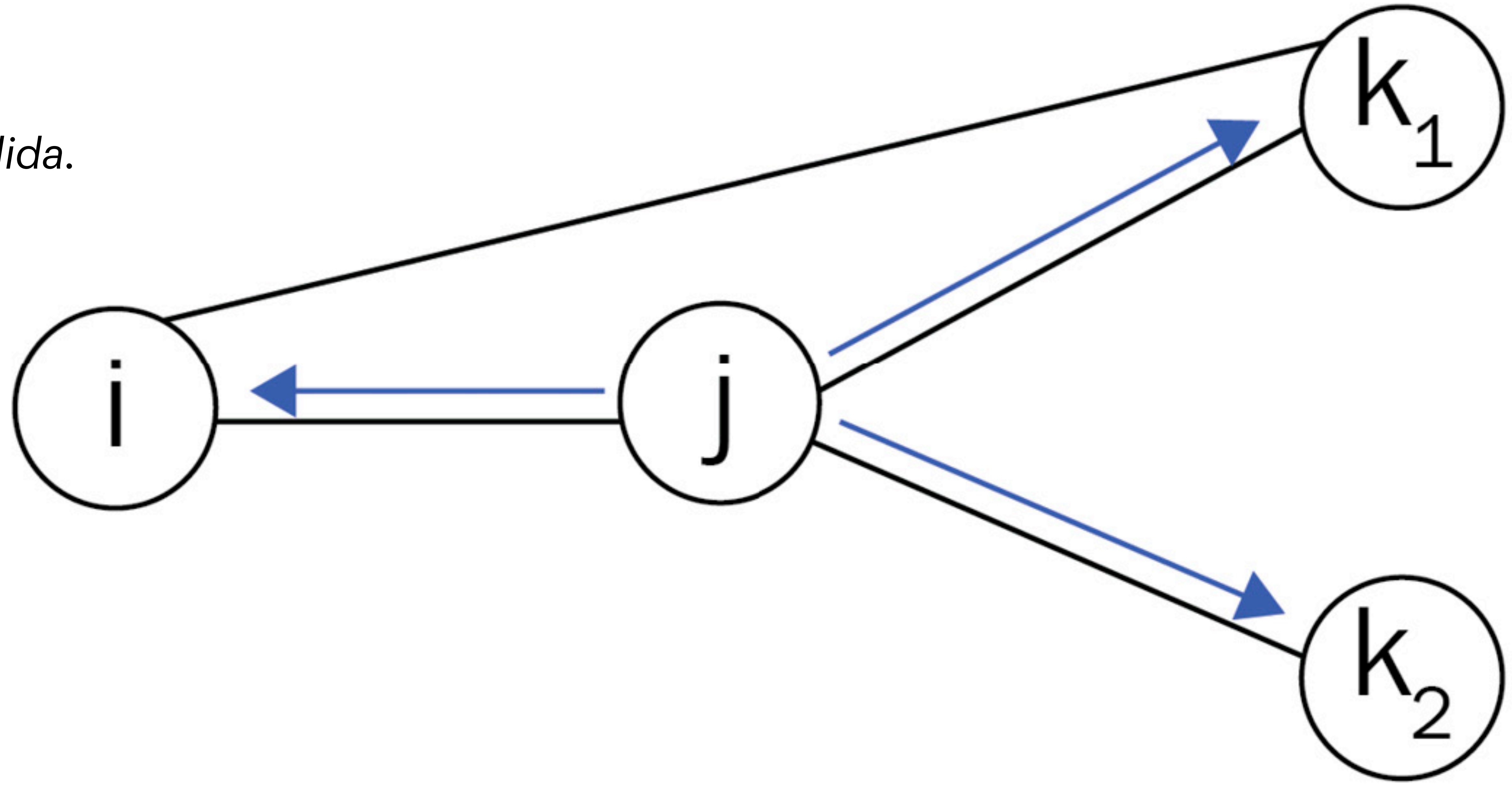
Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

Deep Walk: $\alpha(a, b) = 1$

Node2Vec: función de distancia entre los nodos y dos parámetros.

p *Parámetro de retorno.*

q *Parámetro de entrada y salida.*



Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

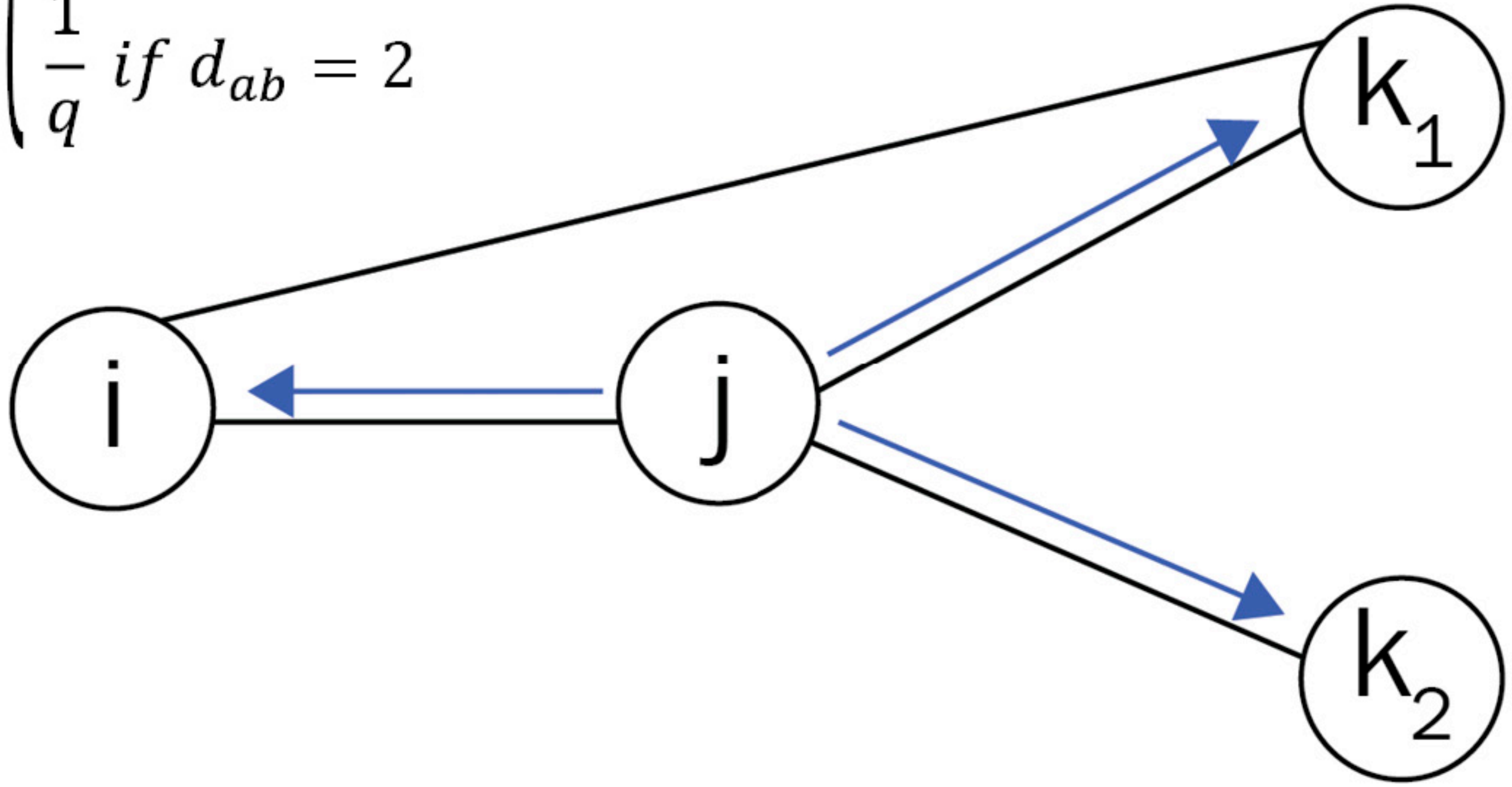
↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

Node2Vec:

$$\alpha(a,b) = \begin{cases} \frac{1}{p} \; if \; d_{ab} = 0 \\ 1 \; if \; d_{ab} = 1 \\ \frac{1}{q} \; if \; d_{ab} = 2 \end{cases}$$

d_{ab} *Distancia de ruta más corta entre a y b.*



Graph Representation

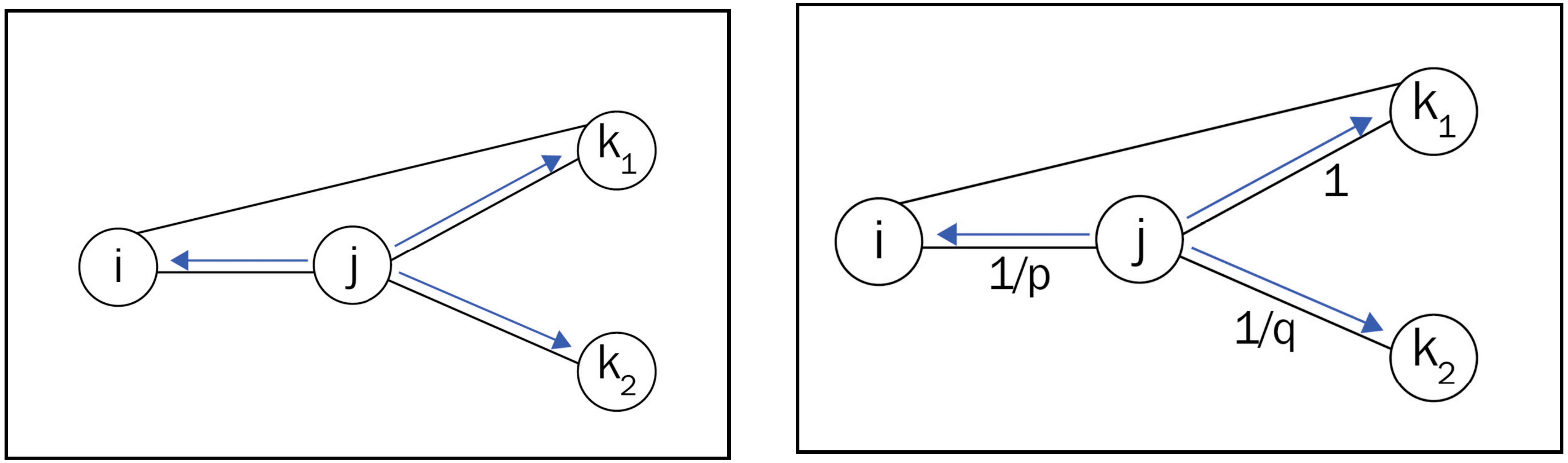
Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.



Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

$1/p$

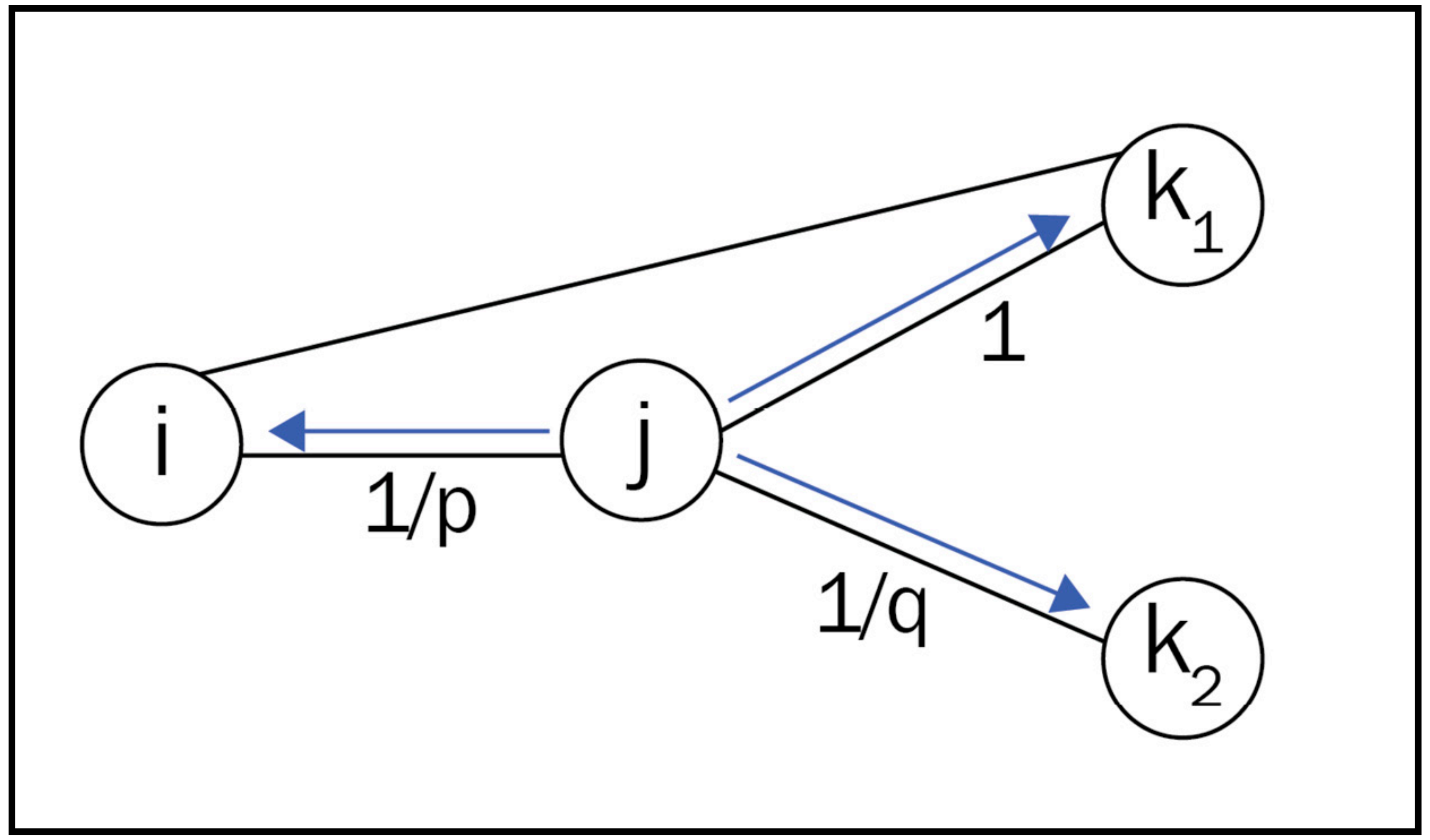
Sí este vecino es el nodo anterior.

1

Sí el vecino esta conectado al nodo anterior.

$1/q$

En caso contrario.



Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

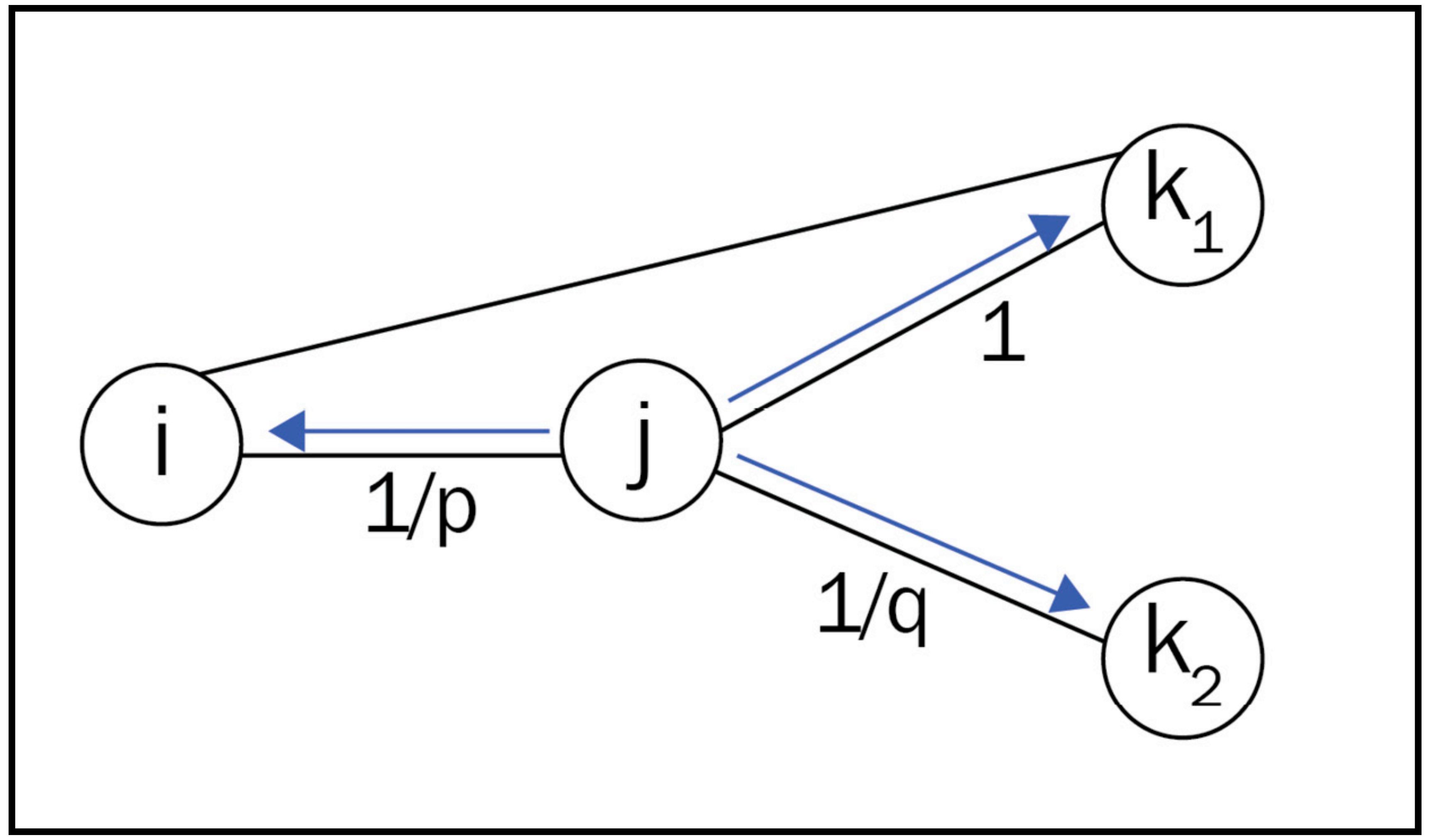
↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

Intuición

Ejemplo detrás de una caminata aleatoria

$$q = 1$$
$$p = 1$$



$$[0,4,7,6,4,5,4,5,6]$$

Caminata aleatoria (deep walk)

Esto debería ser aleatorio ya que cada nodo vecino tiene la misma probabilidad de transición

Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

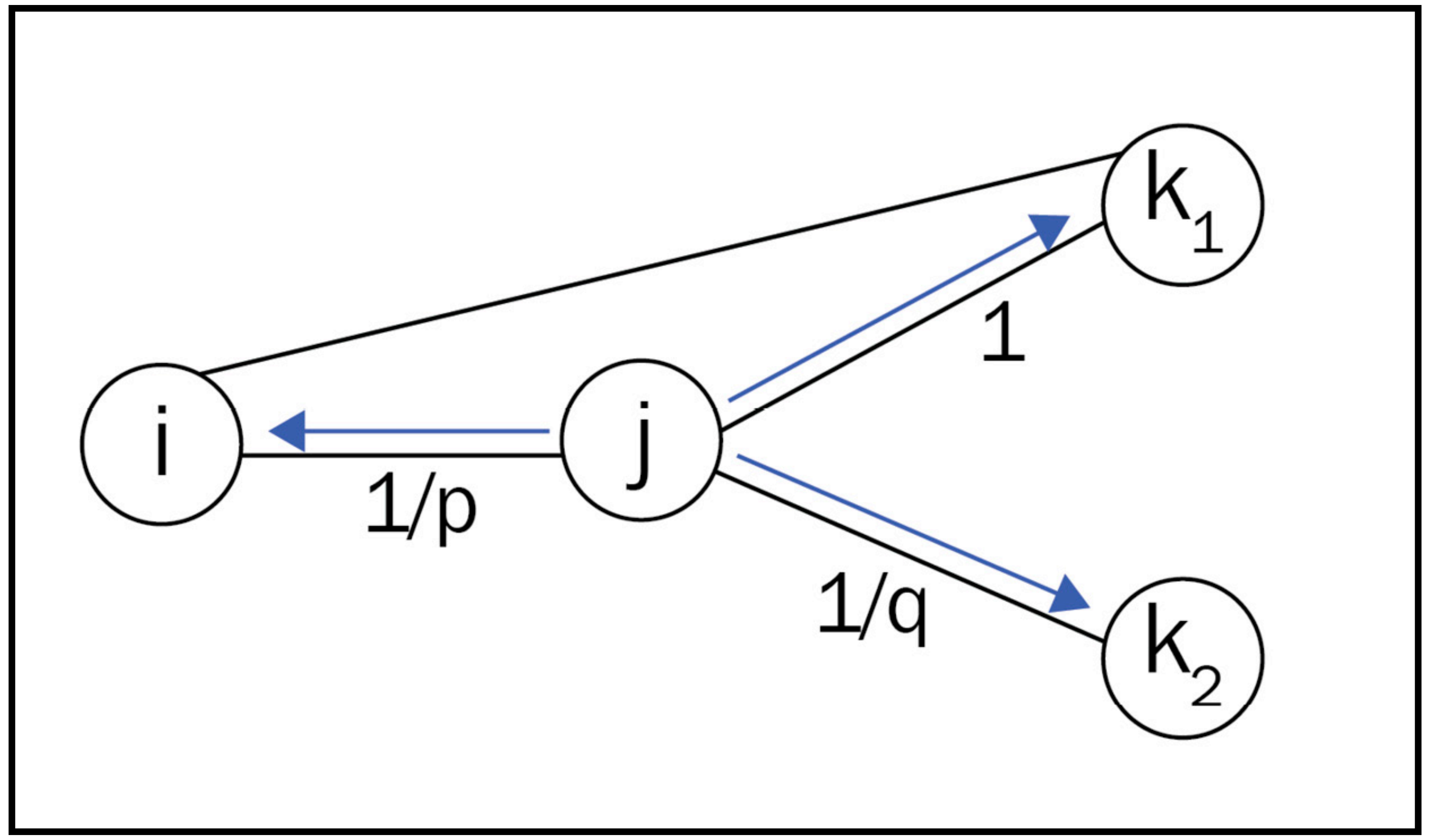
Intuición

Ejemplo detrás de una caminata aleatoria

$$q = 10$$
$$p = 1$$

$$[0,9,1,9,1,9,1,0,1]$$

Siempre regresa al nodo anterior.



Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

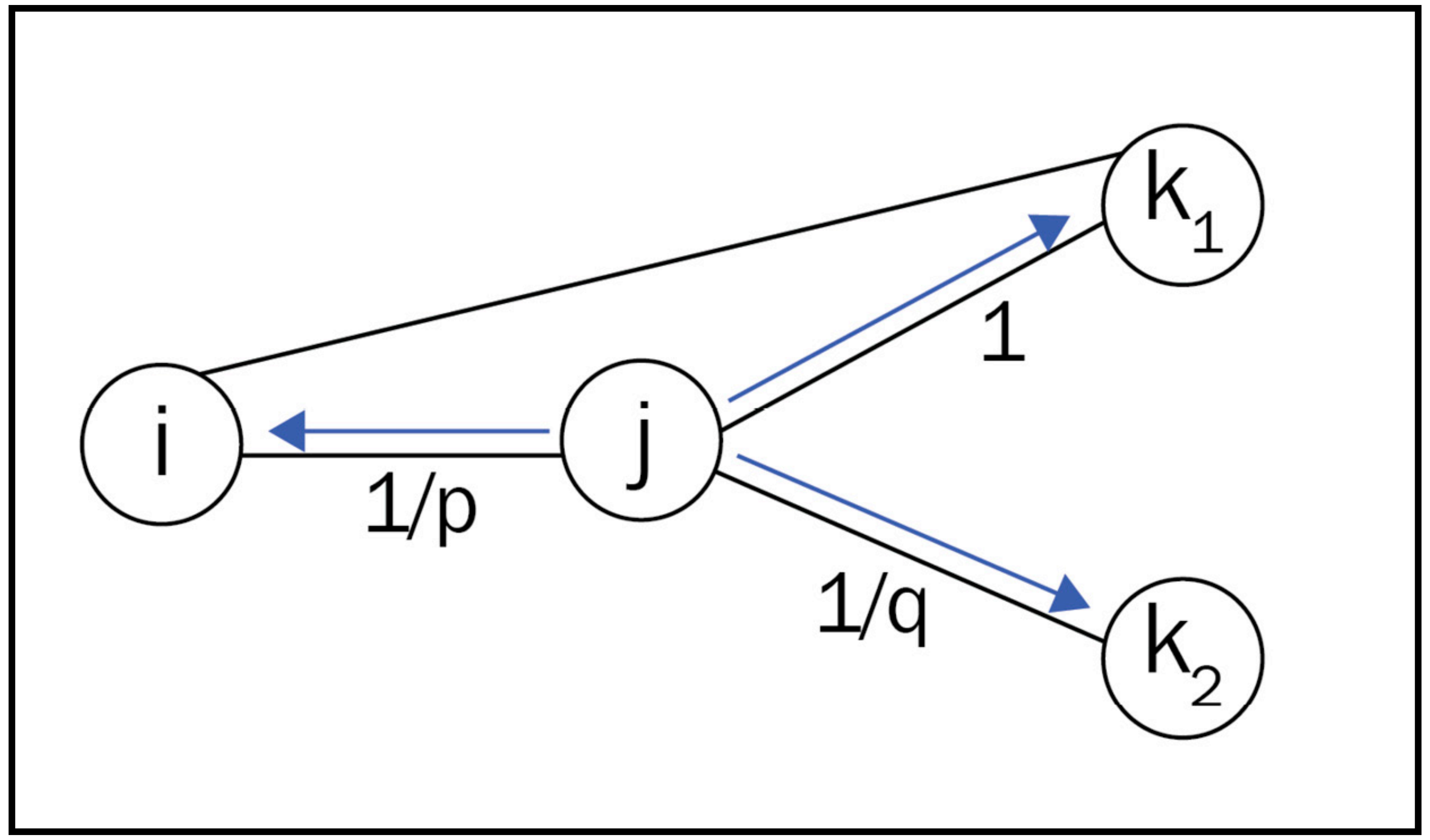
↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

Intuición

Ejemplo detrás de una caminata aleatoria

$$q = 1$$
$$p = 10$$



$$[0,1,9,4,7,8,7,4,6]$$

Nunca vuelve al nodo anterior

El paseo aleatorio explora más nodos en el gráfico

Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

Intuición

Ejemplo detrás de una caminata aleatoria

q *probabilidad de que un paseo aleatorio pueda pasar a través de una parte del gráfico nunca antes vista*

p *probabilidad de que un paseo aleatorio regrese al nodo anterior*

Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

Contras

hay muchos procesos estocásticos involucrados

↓

Se puede obtener un mejor resultado con el peor modelo.

Ajuste de parametros

Hyperparameter tuning

Medir Variabilidad

repetir este proceso 100 veces y tomar el valor medio y la desviación...

Graph Representation

Node2Vec

Biased Random Walks

En Node2Vec, nuestro objetivo es sesgar la aleatoriedad de estos paseos

↓

Caminatas aleatorias con sesgo basado en probabilidades no normalizadas.

Medir Variabilidad

repetir este proceso 100 veces y tomar el valor medio y la desviación...

	p=1	p=2	p=3	p=4	p=5	p=6	p=7
q=1	92.95% (± 4.61%)	94.45% (± 4.19%)	96.36% (± 4.69%)	95.41% (± 4.14%)	95.59% (± 4.30%)	95.82% (± 4.67%)	95.41% (± 3.94%)
q=2	93.64% (± 4.36%)	93.95% (± 3.97%)	95.09% (± 4.34%)	95.55% (± 3.80%)	96.27% (± 3.82%)	96.18% (± 3.90%)	97.45% (± 3.60%)
q=3	93.45% (± 3.82%)	94.41% (± 4.11%)	95.77% (± 3.59%)	95.27% (± 3.63%)	96.68% (± 3.90%)	95.64% (± 3.69%)	96.00% (± 3.82%)
q=4	94.14% (± 3.93%)	94.14% (± 3.93%)	95.45% (± 3.40%)	95.05% (± 3.58%)	95.95% (± 3.46%)	96.41% (± 3.71%)	95.59% (± 3.31%)
q=5	94.41% (± 3.68%)	94.18% (± 3.64%)	94.68% (± 3.58%)	95.36% (± 3.75%)	95.64% (± 3.34%)	95.55% (± 3.58%)	95.27% (± 4.01%)
q=6	94.91% (± 3.71%)	94.55% (± 3.08%)	94.59% (± 3.13%)	95.05% (± 3.86%)	95.77% (± 3.23%)	94.55% (± 4.17%)	95.05% (± 3.75%)
q=7	94.64% (± 4.03%)	95.00% (± 3.78%)	93.59% (± 3.97%)	94.86% (± 3.67%)	94.14% (± 3.87%)	95.27% (± 3.74%)	95.82% (± 3.38%)

Deep Walk:

$$q = 1 \quad p = 1$$

Es la peor combinación de p y q. Los paseos aleatorios no sesgado pueden funcionar mejor en otro conjunto de datos.

Graph Representation

Node2Vec

Step 1:

Definir un vecindario.

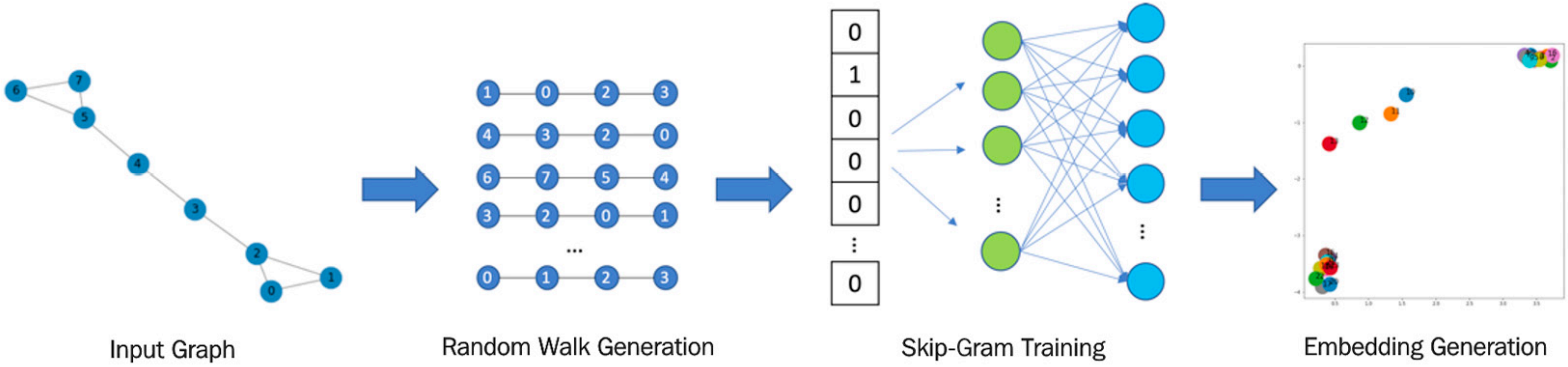
Step 2:

Introducción de sesgos en paseos aleatorios.

Graph Representation

Node2Vec

Steps

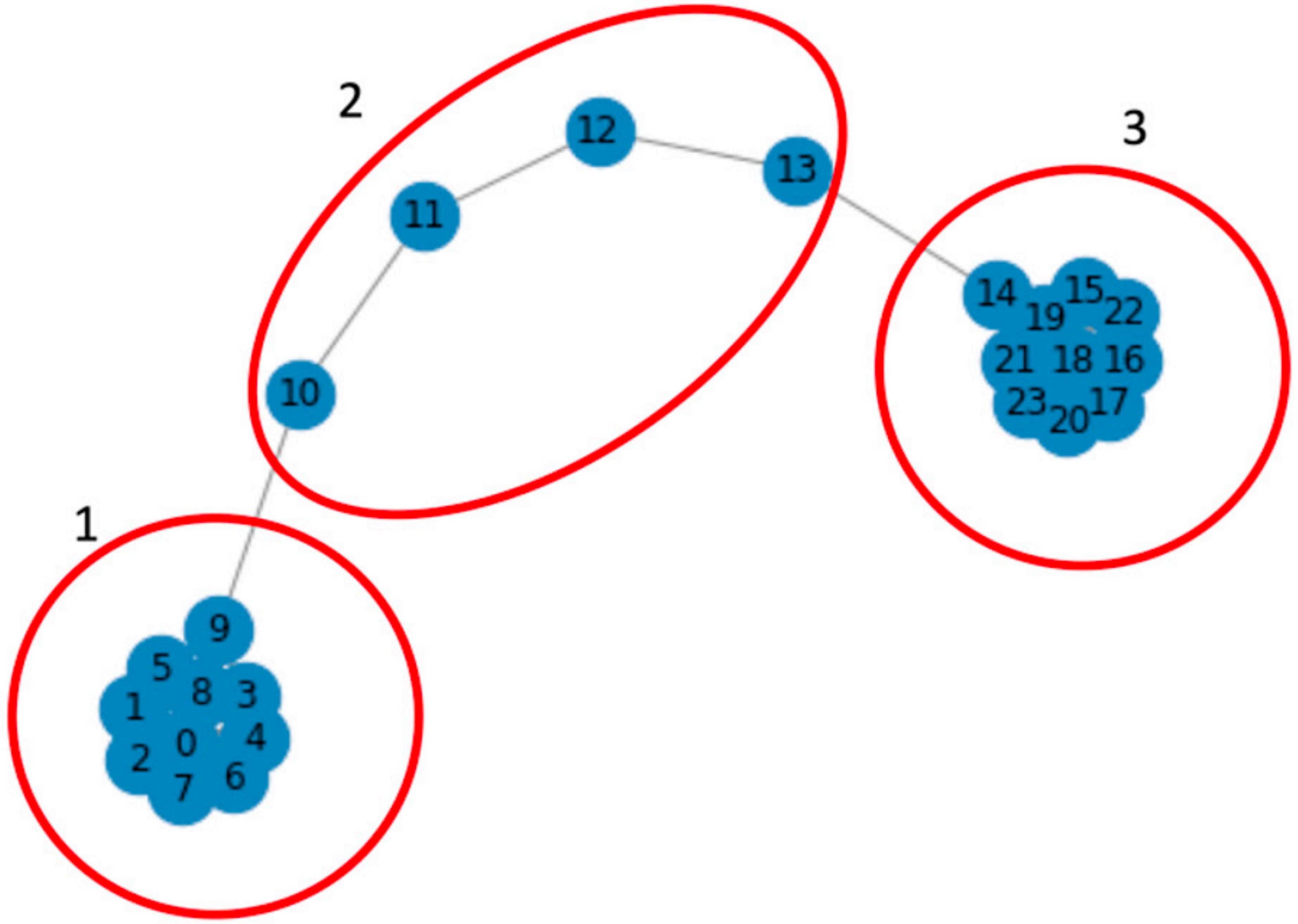


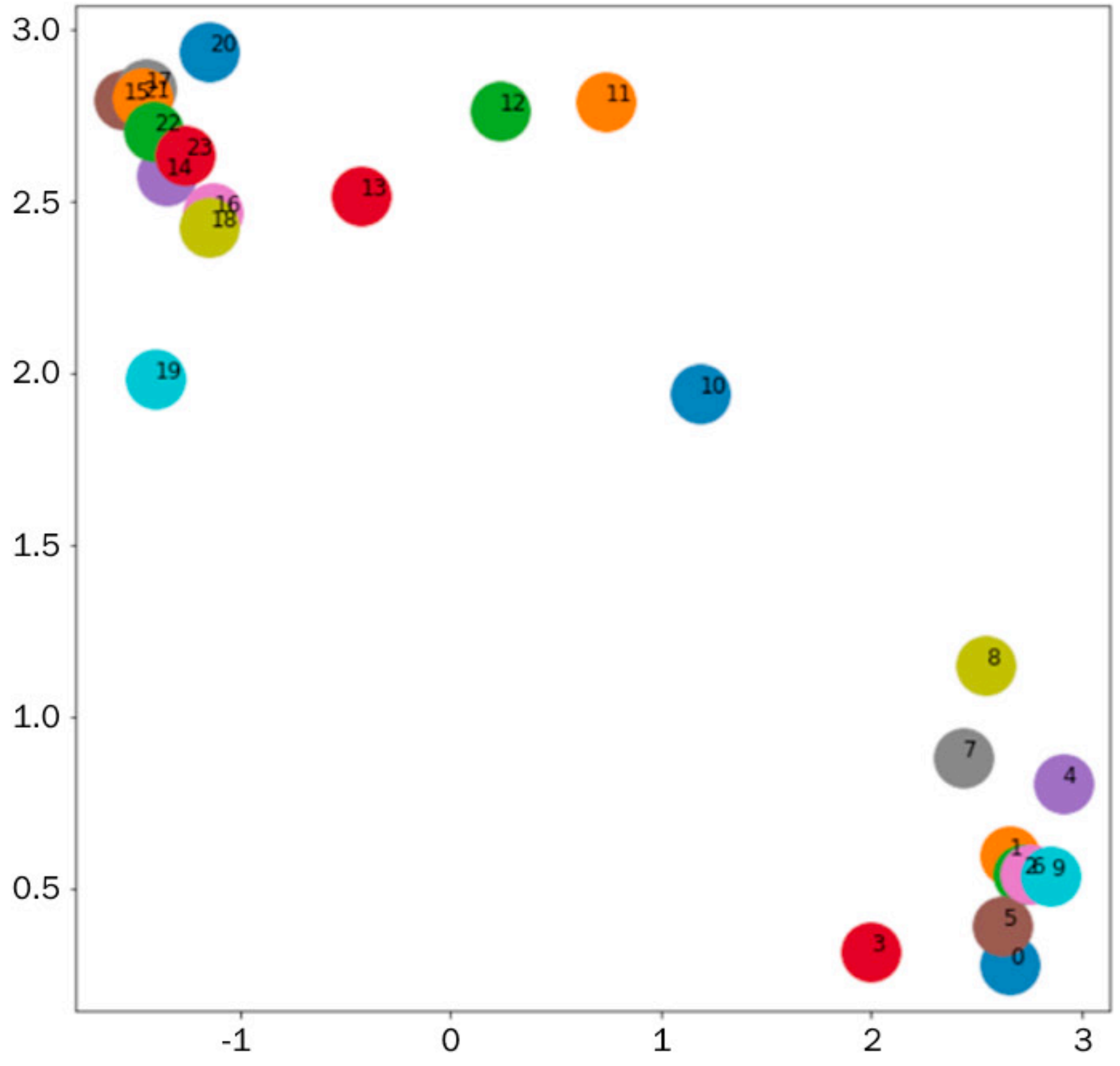
Graph Representation

Node2Vec

Deep Walk Embedding

Aplicación del algoritmo embeddings a un gráfico para generar el vector de incrustación de sus nodos



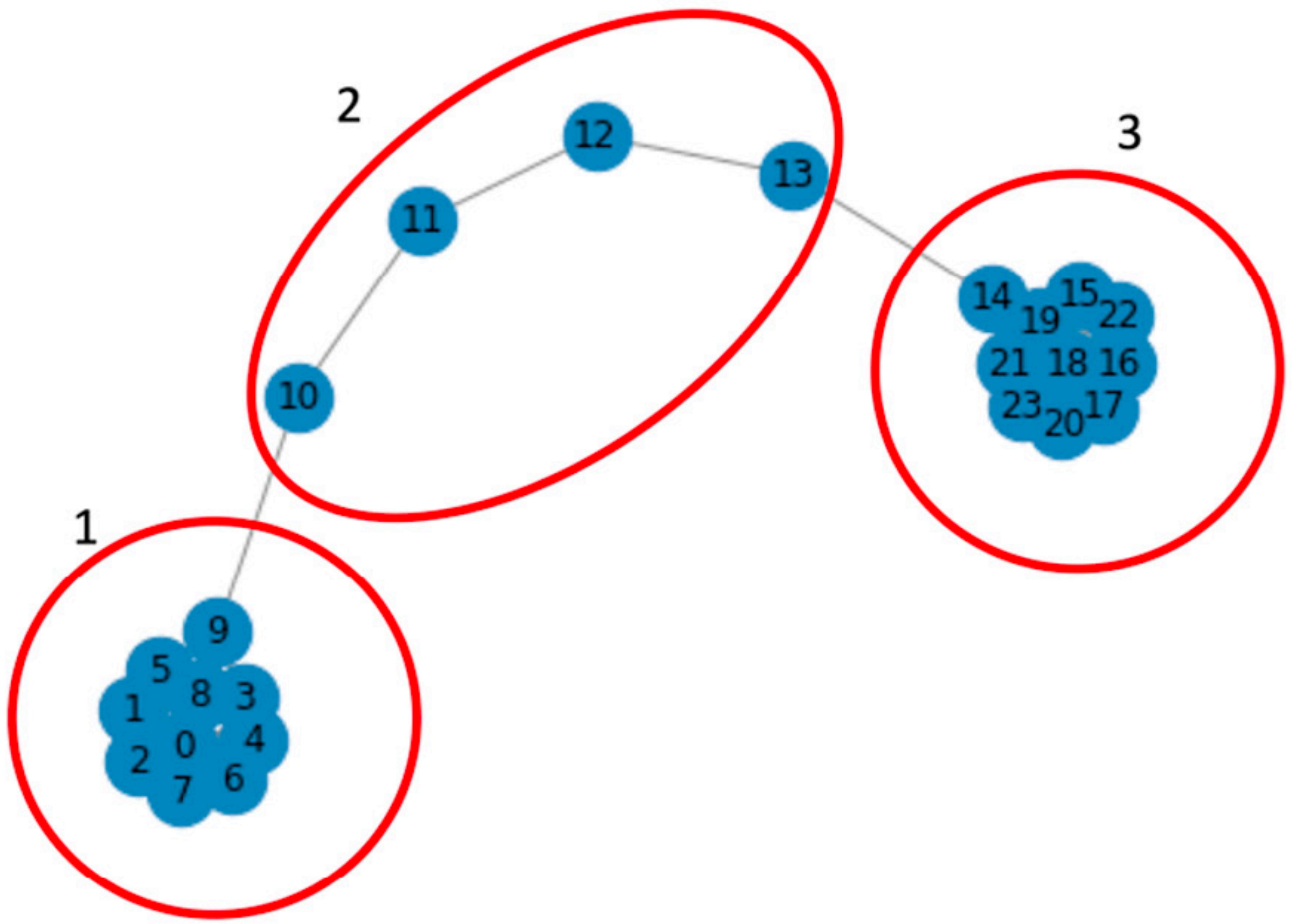


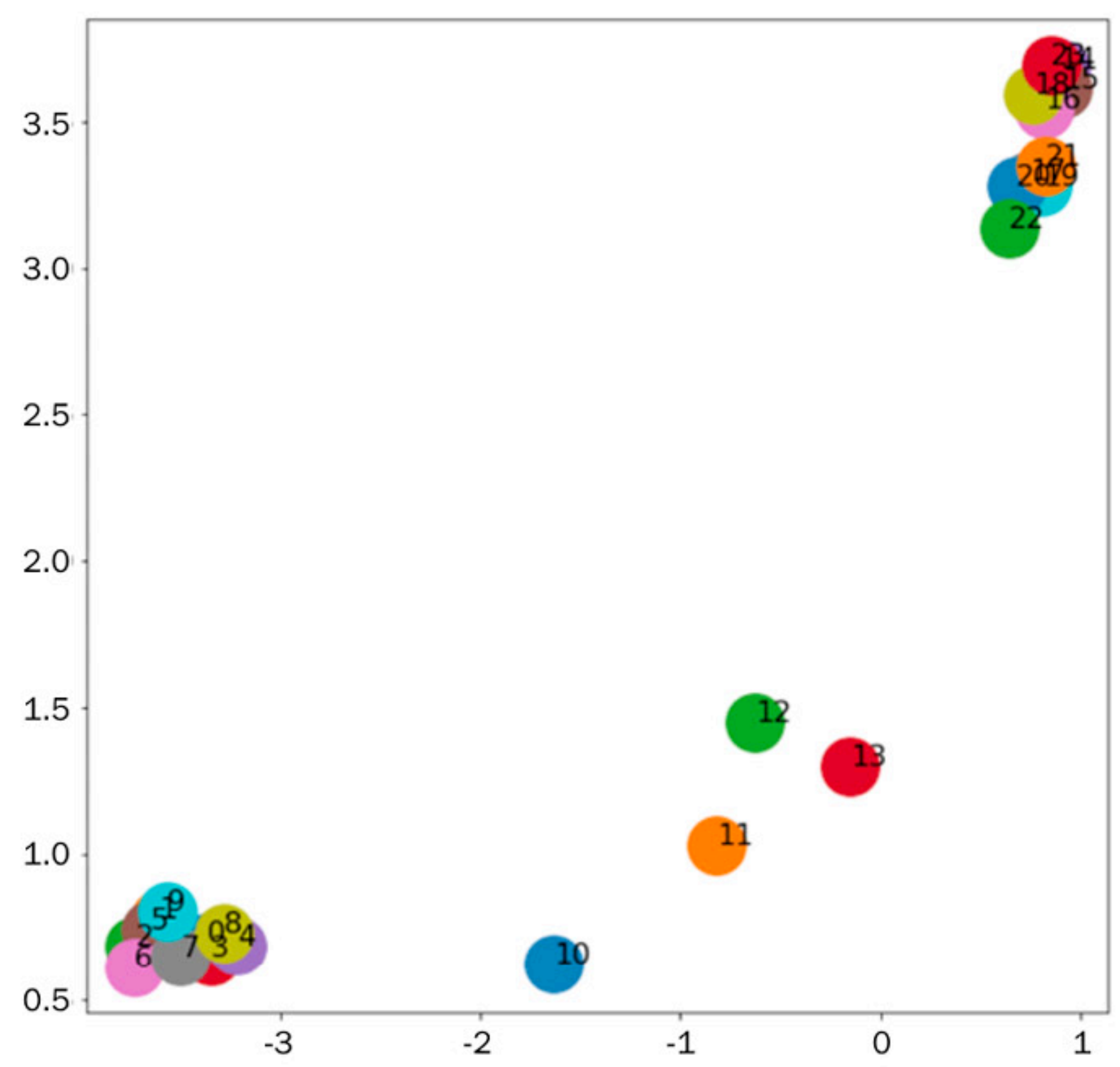
Graph Representation

Node2Vec

Node2Vec Embedding

Aplicación del algoritmo embeddings a un gráfico para generar el vector de incrustación de sus nodos





Graph Representation

Edge2Vec

Edge2Vec Embedding

Genera el espacio de embeddings en los links, en lugar de los nodos

Definición formal

$$v_i \qquad v_j$$

Nodos adyacentes

$$\downarrow \qquad \downarrow$$

$$f(v_i) \qquad f(v_j)$$

Embeddings con Node2Vec

Graph Representation

Edge2Vec

Edge2Vec Embedding

Genera el espacio de embeddings en los links, en lugar de los nodos

Definición formal

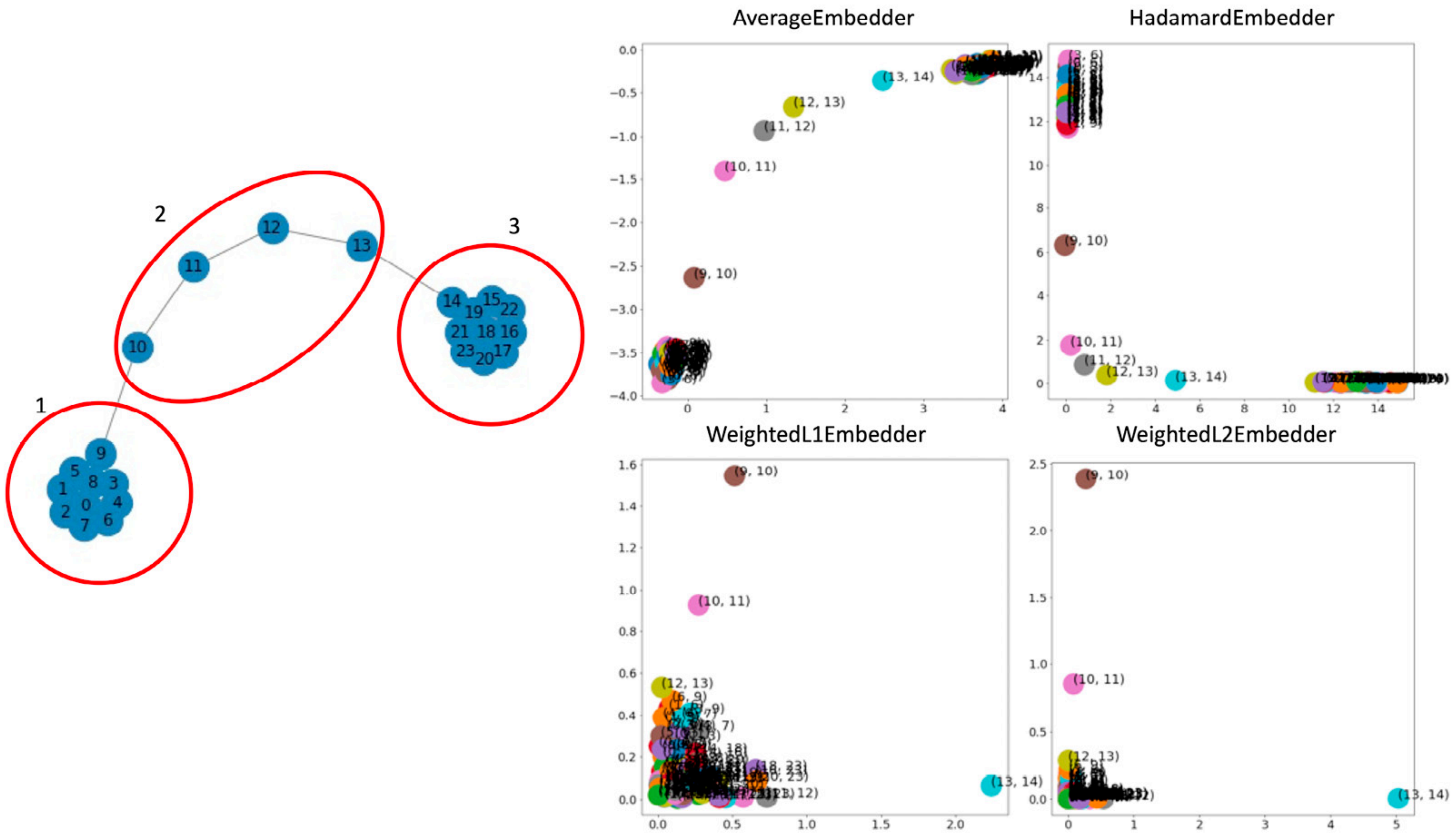
Operator	Equation	Class Name
Average	$\frac{f(v_i) + f(v_j)}{2}$	AverageEmbedder
Hadamard	$f(v_i) * f(v_j)$	HadamardEmbedder
Weighted-L1	$\lvert f(v_i) - f(v_j) \rvert$	WeightedL1Embedder
Weighted-L2	$\lvert f(v_i) - f(v_j) \rvert^2$	WeightedL2Embedder

Graph Representation

Edge2Vec

Edge2Vec Embedding

Genera el espacio de embeddings en los links, en lugar de los nodos



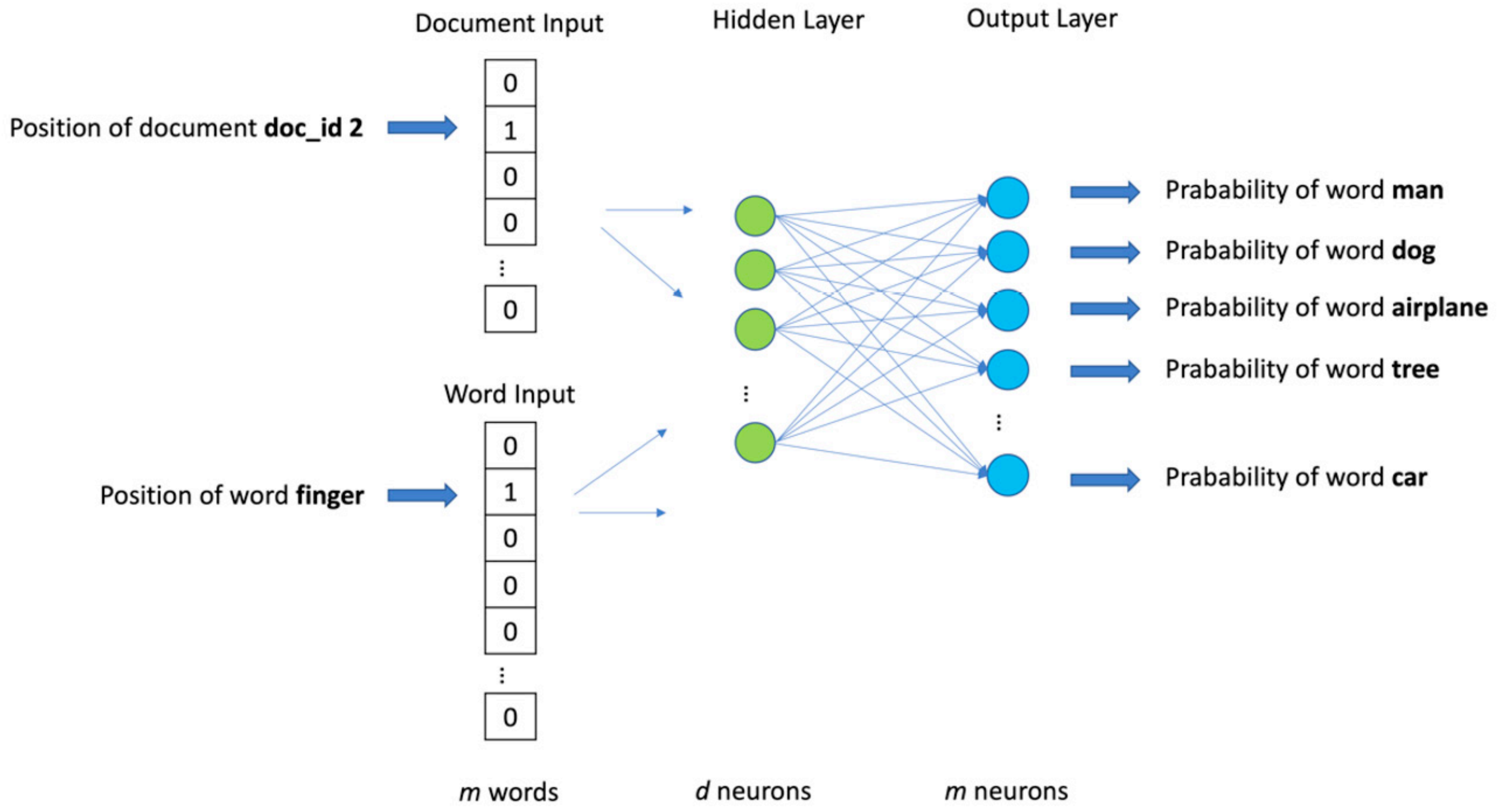
Graph Representation

Graph2Vec

Graph2Vec
Embedding

Doc2Vec

Graph2Vec



Graph Representation

Graph2Vec

Graph2Vec Embedding

Doc2Vec

Graph2Vec

Grafo completo → *Documento*

Subgrafos → *Palabra*

Ego-graph por nodo

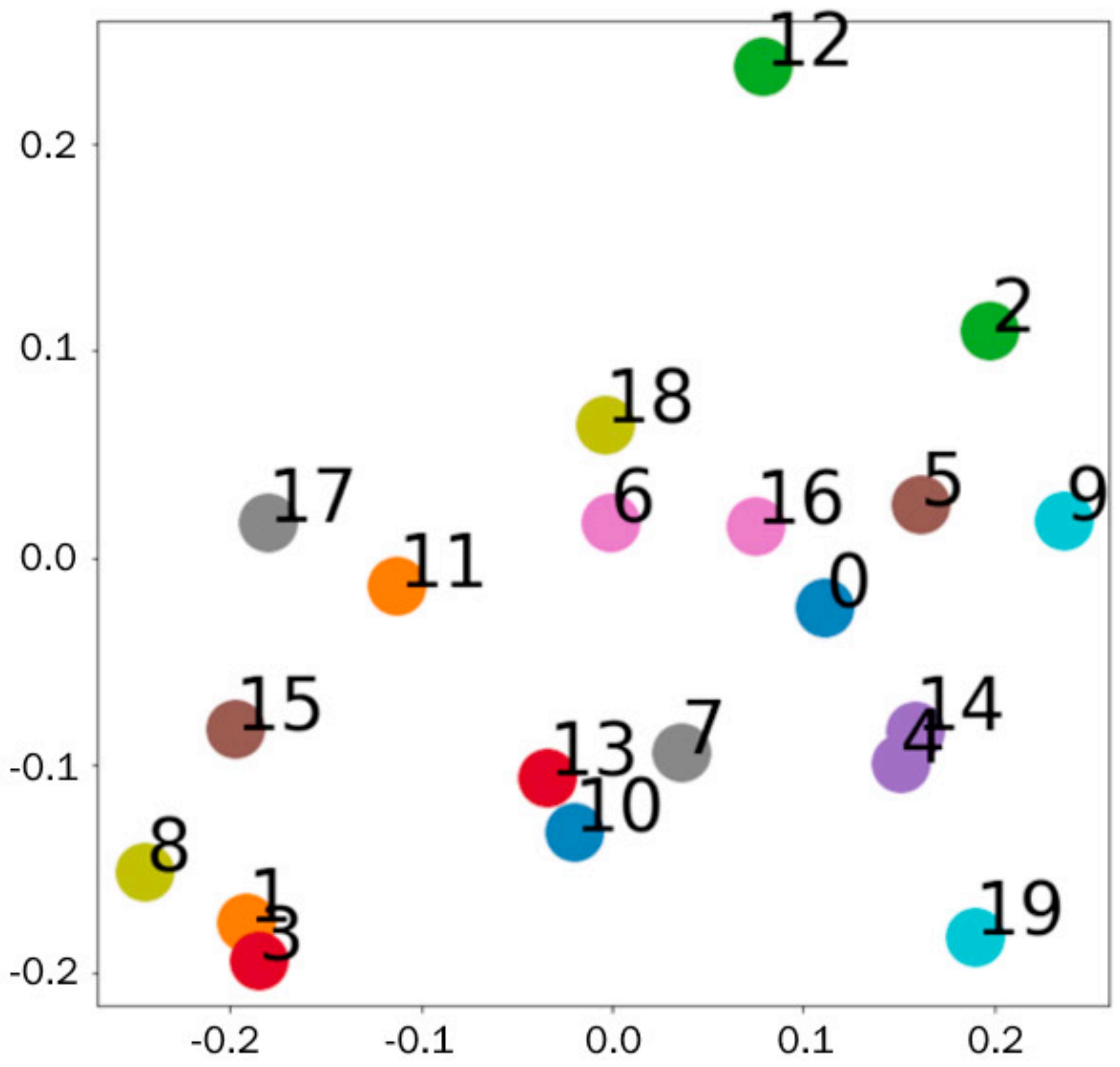
Graph Representation

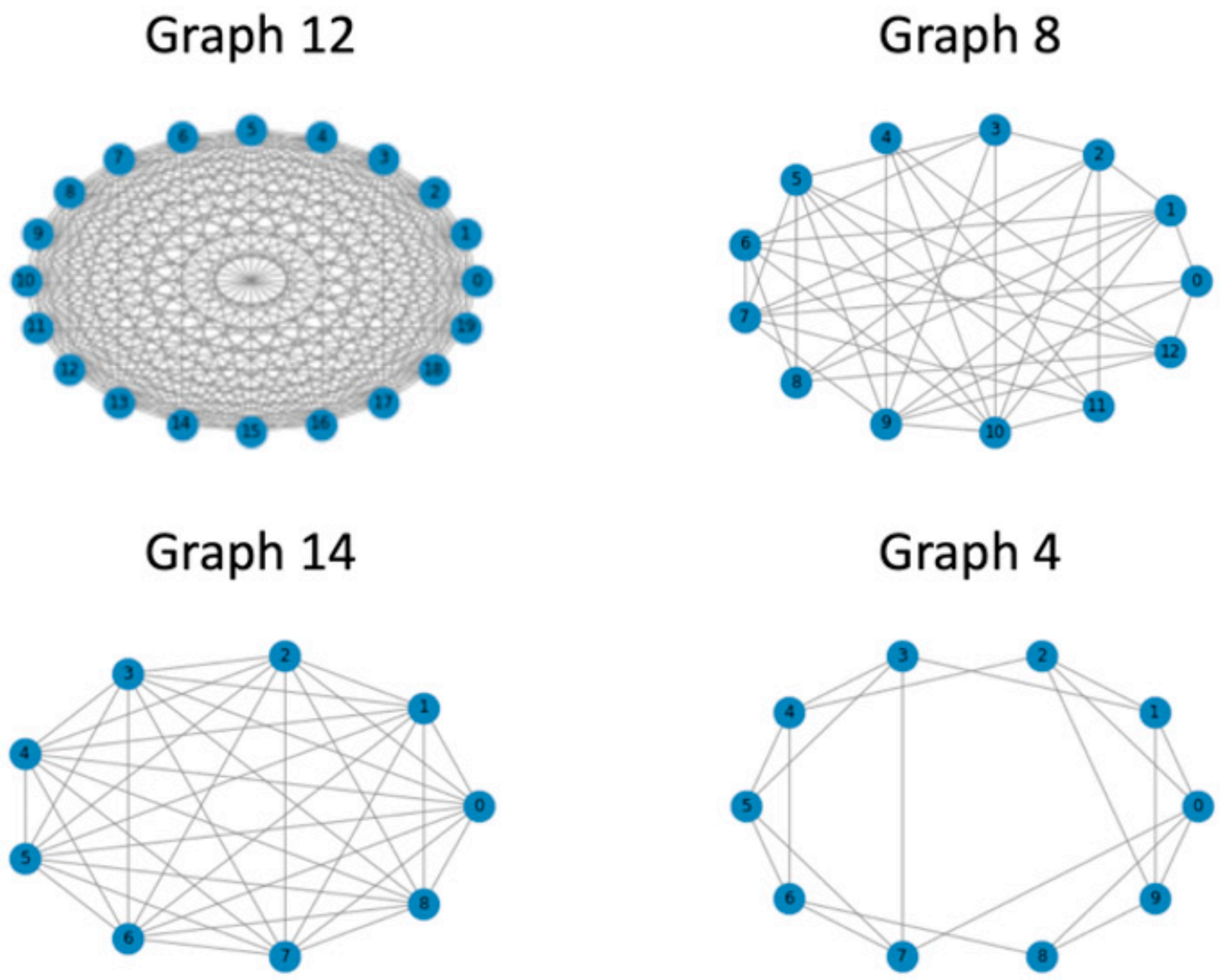
Graph2Vec

Graph2Vec
Embedding

Doc2Vec

Graph2Vec





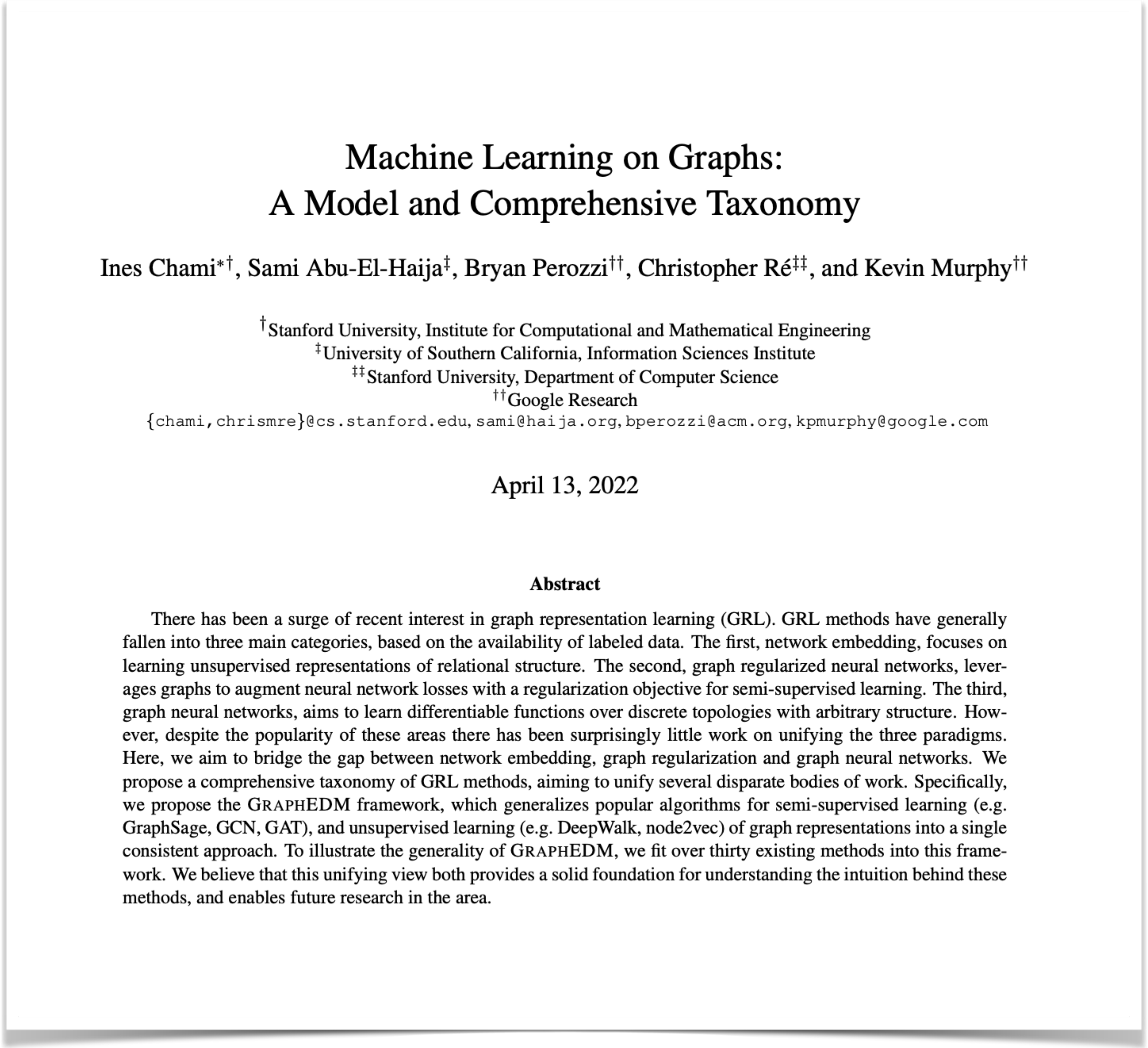
Graph Representation

Graph2Vec

Graph2Vec
Embedding

Doc2Vec

Graph2Vec



Machine Learning on Graphs:
A Model and Comprehensive Taxonomy

Ines Chami*†, Sami Abu-El-Haija‡, Bryan Perozzi††, Christopher Ré‡‡, and Kevin Murphy††

†Stanford University, Institute for Computational and Mathematical Engineering
‡University of Southern California, Information Sciences Institute
‡‡Stanford University, Department of Computer Science
††Google Research
{chami,chrismre}@cs.stanford.edu, sami@haija.org, bperozzi@acm.org, kpmurphy@google.com

April 13, 2022

Abstract

There has been a surge of recent interest in graph representation learning (GRL). GRL methods have generally fallen into three main categories, based on the availability of labeled data. The first, network embedding, focuses on learning unsupervised representations of relational structure. The second, graph regularized neural networks, leverages graphs to augment neural network losses with a regularization objective for semi-supervised learning. The third, graph neural networks, aims to learn differentiable functions over discrete topologies with arbitrary structure. However, despite the popularity of these areas there has been surprisingly little work on unifying the three paradigms. Here, we aim to bridge the gap between network embedding, graph regularization and graph neural networks. We propose a comprehensive taxonomy of GRL methods, aiming to unify several disparate bodies of work. Specifically, we propose the GRAPHEDM framework, which generalizes popular algorithms for semi-supervised learning (e.g. GraphSage, GCN, GAT), and unsupervised learning (e.g. DeepWalk, node2vec) of graph representations into a single consistent approach. To illustrate the generality of GRAPHEDM, we fit over thirty existing methods into this framework. We believe that this unifying view both provides a solid foundation for understanding the intuition behind these methods, and enables future research in the area.