Zero-One Laws of Graph Neural Networks

Benjamín Álvarez Stevenson

November 2024

Zero-One Laws of Graph Neural Networks

Sam Adam-Day*
Department of Mathematics
University of Oxford
Oxford, UK
sam.adam-day@cs.ox.ac.uk

Theodor-Mihai Iliant
Department of Computer Science
University of Oxford
Oxford, UK
theodor-mihai.iliant@lmh.ox.ac.uk

İsmail İlkan Ceylan
Department of Computer Science
University of Oxford
Oxford, UK
ismail.ceylan@cs.ox.ac.uk

¿Cómo se comportan las GNN cuando el número de nodos de los grafos que clasifican crece arbitrariamente?

Contexto

¿Qué es una Ley 0-1?

Contexto: Teoría de Modelos Finitos

- Leyes 0-1 para lógicas definidas sobre vocabularios

¿Qué es una Ley 0-1?

Contexto: Teoría de Modelos Finitos

- Leyes 0-1 para lógicas definidas sobre vocabularios
  - LPO tiene la ley 0-1 sobre $\mathcal{L} = \{R(\cdot, \cdot)\}$

¿Qué es una Ley 0-1?

Contexto: Teoría de Modelos Finitos

- Leyes 0-1 para lógicas definidas sobre vocabularios
  - LPO tiene la ley 0-1 sobre $\mathcal{L} = \{R(\cdot, \cdot)\}$

¿Por qué nos interesan las leyes 0-1?

¿Qué es una Ley 0-1?

Contexto: Teoría de Modelos Finitos

- Leyes 0-1 para lógicas definidas sobre vocabularios
  - LPO tiene la ley 0-1 sobre $\mathcal{L} = \{R(\cdot, \cdot)\}$

¿Por qué nos interesan las leyes 0-1?

- Estudiar el comportamiento de GNNs sobre grafos grandes

¿Qué es una Ley 0-1?

Contexto: Teoría de Modelos Finitos

- Leyes 0-1 para lógicas definidas sobre vocabularios
  - LPO tiene la ley 0-1 sobre $\mathcal{L} = \{R(\cdot, \cdot)\}$

¿Por qué nos interesan las leyes 0-1?

- Estudiar el comportamiento de GNNs sobre grafos grandes
- Establecer límites sobre la expresividad de GNNs

¿Qué es una Ley 0-1?

Contexto: Teoría de Modelos Finitos

- Leyes 0-1 para lógicas definidas sobre vocabularios
  - LPO tiene la ley 0-1 sobre $\mathcal{L} = \{R(\cdot, \cdot)\}$

¿Por qué nos interesan las leyes 0-1?

- Estudiar el comportamiento de GNNs sobre grafos grandes
- Establecer límites sobre la expresividad de GNNs
- Estudiar los límites de extrapolación de las GNNs

Trabajos relacionados

Trabajos relacionados

Algunos trabajos previos se han concentrado en:

Trabajos relacionados

Algunos trabajos previos se han concentrado en:

- Convergencia a **redes límite teóricas** (Keriven et. al)

Trabajos relacionados

Algunos trabajos previos se han concentrado en:

- Convergencia a **redes límite teóricas** (Keriven et. al)
- **Estabilidad** bajo perturbaciones en grafos (Gama et. al)

Trabajos relacionados

Algunos trabajos previos se han concentrado en:

- Convergencia a **redes límite teóricas** (Keriven et. al)
- **Estabilidad** bajo perturbaciones en grafos (Gama et. al)
- Arquitecturas que alcanzan límites de expresibilidad o distintas caracterizaciones para sus expresibilidades
  - **SumGNN$^+$** alcanza el límite superior para MPNNs dado por el test **1-WL** (Morris et. al)
  - **SumGNN$^+$** captura cualquier función que puede ser expresada en la lógica C^2 (Barceló et. al)

Trabajos relacionados

Algunos trabajos previos se han concentrado en:

- Convergencia a **redes límite teóricas** (Keriven et. al)
- **Estabilidad** bajo perturbaciones en grafos (Gama et. al)
- Arquitecturas que alcanzan límites de expresibilidad o distintas caracterizaciones para sus expresibilidades
  - **SumGNN$^+$** alcanza el límite superior para MPNNs dado por el test **1-WL** (Morris et. al)
  - **SumGNN$^+$** captura cualquier función que puede ser expresada en la lógica C^2 (Barceló et. al)
- Resultados **no uniformes** (Abboud et. al; Grohe)

Trabajos relacionados

Algunos trabajos previos se han concentrado en:

- Convergencia a **redes límite teóricas** (Keriven et. al)
- **Estabilidad** bajo perturbaciones en grafos (Gama et. al)
- Arquitecturas que alcanzan límites de expresibilidad o distintas caracterizaciones para sus expresibilidades
  - **SumGNN$^+$** alcanza el límite superior para MPNNs dado por el test **1-WL** (Morris et. al)
  - **SumGNN$^+$** captura cualquier función que puede ser expresada en la lógica C^2 (Barceló et. al)
- Resultados **no uniformes** (Abboud et. al; Grohe)
- Otras limitaciones relacionadas a la cantidad de *layers*
  - **Over-smoothing** y **over-squashing** (Li et. al; Alon et. al)

Conceptos Preliminares

Preliminares: Message Passing Neural Networks (MPNNs)

Trabajamos con MPNNs, las cuales encapsulan la mayoría de las GNNs. Para cada nodo $v \in V$, actualizamos el valor de su vector de *features* inicial $\mathbf{x}_v^{(0)} = \mathbf{x}_v$ por $T \in \mathbb{N}$ iteraciones (*layers*), basado en su propio estado y el de sus vecinos $\mathcal{N}(v)$:

$$\mathbf{x}_v^{(t+1)} = \phi^{(t)}\left(\mathbf{x}_v^{(t)}, \psi^{(t)}(\mathbf{x}_v^{(t)}, \{\!\{\mathbf{x}_u^{(t)} \mid u \in \mathcal{N}(v)\}\!\})\right)$$

- $\phi^{(t)}$ es una función de **combinación**
- $\psi^{(t)}$ es una función de **agregación**

Preliminares: Message Passing Neural Networks (MPNNs)

Trabajamos con MPNNs, las cuales encapsulan la mayoría de las GNNs. Para cada nodo $v \in V$, actualizamos el valor de su vector de *features* inicial $\mathbf{x}_v^{(0)} = \mathbf{x}_v$ por $T \in \mathbb{N}$ iteraciones (*layers*), basado en su propio estado y el de sus vecinos $\mathcal{N}(v)$:

$$\mathbf{x}_v^{(t+1)} = \phi^{(t)}\left(\mathbf{x}_v^{(t)}, \psi^{(t)}(\mathbf{x}_v^{(t)}, \{\!\{\mathbf{x}_u^{(t)} \mid u \in \mathcal{N}(v)\}\!\})\right)$$

- $\phi^{(t)}$ es una función de **combinación**
- $\psi^{(t)}$ es una función de **agregación**

Nota: las representaciones de los nodos en cada iteración t pueden tener distintas dimensiones, las cuales denotamos por $d(t)$. Usamos $d(0) = d$

Preliminares: Clasificación

Después de la última iteración ($t = T$):

- $\mathbf{x}_v$ se puede usar para realizar predicciones a nivel del nodo v
- Si queremos realizar predicciones a nivel de grafo, realizamos **pooling** con los *embeddings* finales de los nodos
  - Por lo general una suma, promedio o máximo por componentes

Preliminares: Clasificación

Después de la última iteración ($t = T$):

- $\mathbf{x}_v$ se puede usar para realizar predicciones a nivel del nodo v
- Si queremos realizar predicciones a nivel de grafo, realizamos **pooling** con los *embeddings* finales de los nodos
  - Por lo general una suma, promedio o máximo por componentes

Para **clasificación booleana**, utilizamos un clasificador booleano

$$\mathfrak{C} : \mathbb{R}^{d(T)} \to \mathbb{B}$$

Preliminares: Arquitecturas

Nos concentramos en las siguientes arquitecturas, las cuales difieren principalmente en la forma en la que realizan la agregación al actualizar cada capa:

Preliminares: Arquitecturas

Nos concentramos en las siguientes arquitecturas, las cuales difieren principalmente en la forma en la que realizan la agregación al actualizar cada capa:

- **GCN** (Graph Convolutional Networks)

Preliminares: Arquitecturas

Nos concentramos en las siguientes arquitecturas, las cuales difieren principalmente en la forma en la que realizan la agregación al actualizar cada capa:

- **GCN** (Graph Convolutional Networks)
- **MeanGNN$^+$**

Preliminares: Arquitecturas

Nos concentramos en las siguientes arquitecturas, las cuales difieren principalmente en la forma en la que realizan la agregación al actualizar cada capa:

- **GCN** (Graph Convolutional Networks)
- **MeanGNN$^+$**
- **SumGNN$^+$**

Preliminares: Arquitecturas

Nos concentramos en las siguientes arquitecturas, las cuales difieren principalmente en la forma en la que realizan la agregación al actualizar cada capa:

- **GCN** (Graph Convolutional Networks)
- **MeanGNN$^+$**
- **SumGNN$^+$**

Cada una de ellas realiza una actualización en los nodos $v \in V$ de la forma $\mathbf{x}_v^{(t)} = \sigma(\mathbf{y}_v^{(t)})$, donde $\mathbf{y}_v^{(t)}$ varía según la arquitectura.

- $\sigma : \mathbb{R} \to \mathbb{R}$ es una función de combinación o *update*, también referida a veces como *non-linearity*

Arquitecturas: **GCN**

Las actualizaciones se realizan de la forma $\mathbf{x}_v^{(t)} = \sigma(\mathbf{y}_v^{(t)})$, con:

$$\mathbf{y}_v^{(t)} = \mathbf{W}_n^{(t)} \sum_{u \in \mathcal{N}^+(v)} \frac{1}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}} \mathbf{x}_u^{(t-1)} + \mathbf{b}^{(t)}$$

donde $\mathbf{W}_n^{(t)} \in \mathbb{R}^{d(t) \times d(t-1)}$ representa una transformación lineal, $\mathbf{b}^{(t)} \in \mathbb{R}^{d(t)}$ es un término de *bias* y $\mathcal{N}^+(v)$ corresponde al vecindario extendido del nodo v:

$$\mathcal{N}^+(v) := \mathcal{N}(v) \cup \{v\}$$

Arquitecturas: **MeanGNN$^+$**

Las actualizaciones se realizan de la forma $\mathbf{x}_v^{(t)} = \sigma(\mathbf{y}_v^{(t)})$, con:

$$\mathbf{y}_v^{(t)} = \frac{1}{|\mathcal{N}^+(v)|}\mathbf{W}_n^{(t)} \sum_{u \in \mathcal{N}^+(v)} \mathbf{x}_u^{(t-1)}$$

$$+ \frac{1}{n}\mathbf{W}_r^{(t)} \sum_{u \in V} \mathbf{x}_u^{(t-1)} \quad + \quad \mathbf{b}^{(t)}$$

En este caso, la transformación lineal $\mathbf{W}_r^{(t)}$ se aplica a la media de las representaciones de todos los nodos. A este componente se le llama **global readout**, y arquitecturas con este tipo de agregación son más expresivas. Sin este término, tenemos una **MeanGNN**

Arquitecturas: **SumGNN$^+$**

Las actualizaciones se realizan de la forma $\mathbf{x}_v^{(t)} = \sigma(\mathbf{y}_v^{(t)})$, con:

$$\mathbf{y}_v^{(t)} = \mathbf{W}_s^{(t)}\mathbf{x}_v^{(t-1)} + \mathbf{W}_n^{(t)} \sum_{u \in \mathcal{N}(v)} \mathbf{x}_u^{(t-1)}$$

$$+ \mathbf{W}_r^{(t)} \sum_{u \in V} \mathbf{x}_v^{(t-1)} + \mathbf{b}^{(t)}$$

Esta vez **separamos** la contribución del nodo mismo para su preactivación, por lo que la arquitectura se dice que no tiene *self-loops*. Sin el *global readout*, tenemos una **SumGNN**

Preliminares: Grafos y Matrices aleatorias

Definimos un grafo de Erdős–Rényi equipado con *features* aleatorias en sus nodos como un par $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ donde:

- $\mathbf{A} \sim \mathbb{G}(n, r)$ es la matriz de adyacencia del grafo $G = (V, E)$
  - $\mathbb{G}(n, r)$ es la distribución de Erdős–Rényi
- $\mathbf{X} \in \mathbb{R}^{d \times n}$ es una matriz de *features* tal que

$$\mathbf{X} = [\mathbf{x}_{v_1}\ \mathbf{x}_{v_2}\ \ldots\ \mathbf{x}_{v_n}] \text{ con } V = \{v_1, v_2, \ldots, v_n\}$$

  y para todo $v \in V$ se tiene que $\mathbf{x}_v \sim \mathbb{D}(d)$
  - $\mathbb{D}(d)$ es una distribución de vectores de *features* sobre $\mathbb{R}^d$

Definición del problema

Definición: invariante de grafos

Un **invariante de grafos** ξ es una función sobre grafos tal que, para todo par de grafos G_1 y G_2, y para todo isomorfismo f entre G_1 y G_2, se tiene que:

$$\xi(G_1) = \xi(f(G_2))$$

Los invariantes para grafos con *features* en sus nodos se definen de manera análoga.

Importante

Todo modelo GNN $\mathcal{M}$ que es usado para clasificación binaria es un invariante de grafos.

Importante

Todo modelo GNN $\mathcal{M}$ que es usado para clasificación binaria es un invariante de grafos.

¿Por qué?

Importante

Todo modelo GNN $\mathcal{M}$ que es usado para clasificación binaria es un invariante de grafos.

¿Por qué?

- $\mathcal{M}$ es una función de un espacio de grafos a $\mathbb{B} = \{0, 1\}$

Definición: leyes 0-1 para invariantes de grafos

Sea $\mathcal{G} = (\mathbf{A}, \mathbf{X})$ un grafo con *features* en sus nodos, donde:

- $\mathbf{A} \sim \mathbb{G}(n, r)$ es la matriz de adyacencia de un grafo
- $\mathbf{X}$ es una matriz de *node embeddings*
  - $\mathbf{x}_v \sim \mathbb{D}(d)$ para cada nodo $v \in V$

Un invariante para grafos ξ con *features* en sus nodos **satisface una ley 0-1** con respecto a $\mathbb{G}(n, r)$ y $\mathbb{D}(d)$ si, cuando n tiende a infinito, la probabilidad de que $\xi(\mathcal{G}) = 1$ **tiende a 0 o 1**

Primer resultado:

GCNs satisfacen leyes 0-1

Teorema 4.6

Sea $\mathcal{M}$ una GCN usada para clasificación binaria y sea $r \in [0, 1]$. Entonces, $\mathcal{M}$ **satisface una ley 0-1** con respecto a la distribución de grafos $\mathbb{G}(n, r)$ y la distribución de *features* $\mathbb{D}(d)$

Teorema 4.6

Sea $\mathcal{M}$ una GCN usada para clasificación binaria y sea $r \in [0, 1]$. Entonces, $\mathcal{M}$ **satisface una ley 0-1** con respecto a la distribución de grafos $\mathbb{G}(n, r)$ y la distribución de *features* $\mathbb{D}(d)$, si las siguientes **condiciones** se cumplen:

(i) la distribución $\mathbb{D}(d)$ es sub-Gaussiana

(ii) la función no-lineal σ es Lipschitz-continua

(iii) la representación a nivel de grafo usa *average pooling*

(iv) el clasificador $\mathfrak{C}$ usado por $\mathcal{M}$ es *non-splitting*

Teorema 4.6

Sea $\mathcal{M}$ una GCN usada para clasificación binaria y sea $r \in [0, 1]$. Entonces, $\mathcal{M}$ **satisface una ley 0-1** con respecto a la distribución de grafos $\mathbb{G}(n, r)$ y la distribución de *features* $\mathbb{D}(d)$, si las siguientes **condiciones** se cumplen:

(i) la distribución $\mathbb{D}(d)$ es sub-Gaussiana

(ii) la función no-lineal σ es Lipschitz-continua

(iii) la representación a nivel de grafo usa *average pooling*

(iv) el clasificador $\mathfrak{C}$ usado por $\mathcal{M}$ es *non-splitting*

¿Demasiadas condiciones?

Definición: vector aleatorio sub-Gaussiano

Un vector aleatorio $\mathbf{x} \in \mathbb{R}^d$ se dice **sub-Gaussiano** si existe $C > 0$ tal que para cada vector unitario $\mathbf{y} \in \mathbb{R}^d$, la variable aleatoria $\mathbf{x} \cdot \mathbf{y}$ satisface la *propiedad sub-Gaussiana*, es decir, para todo $t > 0$:

$$\mathbb{P}(|\mathbf{x} \cdot \mathbf{y}| \geq t) \leq 2\exp\left(- \frac{t^2}{C^2} \right)$$

Definición: vector aleatorio sub-Gaussiano

Un vector aleatorio $\mathbf{x} \in \mathbb{R}^d$ se dice **sub-Gaussiano** si existe $C > 0$ tal que para cada vector unitario $\mathbf{y} \in \mathbb{R}^d$, la variable aleatoria $\mathbf{x} \cdot \mathbf{y}$ satisface la *propiedad sub-Gaussiana*, es decir, para todo $t > 0$:

$$\mathbb{P}(|\mathbf{x} \cdot \mathbf{y}| \geq t) \leq 2\exp\left(- \frac{t^2}{C^2} \right)$$

¿Entonces?

Definición: vector aleatorio sub-Gaussiano

Un vector aleatorio $\mathbf{x} \in \mathbb{R}^d$ se dice **sub-Gaussiano** si existe $C > 0$ tal que para cada vector unitario $\mathbf{y} \in \mathbb{R}^d$, la variable aleatoria $\mathbf{x} \cdot \mathbf{y}$ satisface la *propiedad sub-Gaussiana*, es decir, para todo $t > 0$:

$$\mathbb{P}(|\mathbf{x} \cdot \mathbf{y}| \geq t) \leq 2\exp\left(- \frac{t^2}{C^2} \right)$$

¿Entonces?

Definición: vector aleatorio sub-Gaussiano

Un vector aleatorio $\mathbf{x} \in \mathbb{R}^d$ se dice **sub-Gaussiano** si existe $C > 0$ tal que para cada vector unitario $\mathbf{y} \in \mathbb{R}^d$, la variable aleatoria $\mathbf{x} \cdot \mathbf{y}$ satisface la *propiedad sub-Gaussiana*, es decir, para todo $t > 0$:

$$\mathbb{P}(|\mathbf{x} \cdot \mathbf{y}| \geq t) \leq 2\exp\left(-\frac{t^2}{C^2}\right)$$

¿Entonces?

- Todos los vectores aleatorios acotados son sub-Gaussianos

Definición: vector aleatorio sub-Gaussiano

Un vector aleatorio $\mathbf{x} \in \mathbb{R}^d$ se dice **sub-Gaussiano** si existe $C > 0$ tal que para cada vector unitario $\mathbf{y} \in \mathbb{R}^d$, la variable aleatoria $\mathbf{x} \cdot \mathbf{y}$ satisface la *propiedad sub-Gaussiana*, es decir, para todo $t > 0$:

$$\mathbb{P}(|\mathbf{x} \cdot \mathbf{y}| \geq t) \leq 2\exp\left(- \frac{t^2}{C^2} \right)$$

¿Entonces?

- Todos los vectores aleatorios acotados son sub-Gaussianos
- Todos los vectores multivariables aleatorios normales son sub-Gaussianos

Definición: vector aleatorio sub-Gaussiano

Un vector aleatorio $\mathbf{x} \in \mathbb{R}^d$ se dice **sub-Gaussiano** si existe $C > 0$ tal que para cada vector unitario $\mathbf{y} \in \mathbb{R}^d$, la variable aleatoria $\mathbf{x} \cdot \mathbf{y}$ satisface la *propiedad sub-Gaussiana*, es decir, para todo $t > 0$:

$$\mathbb{P}(|\mathbf{x} \cdot \mathbf{y}| \geq t) \leq 2\exp\left(-\frac{t^2}{C^2}\right)$$

¿Entonces?

- Todos los vectores aleatorios acotados son sub-Gaussianos
- Todos los vectores multivariables aleatorios normales son sub-Gaussianos
- Cota dada por la cantidad de bits del sistema de almacenamiento en escenarios prácticos

Definición: función Lipschitz-continua

Una función $f : \mathbb{R} \to \mathbb{R}$ se dice **Lipschitz-continua** si existe un $C > 0$ tal que para todo $x, y \in \mathbb{R}$ se cumple que:

$$|f(x) - f(y)| \leq C|x - y|$$

Definición: función Lipschitz-continua

Una función $f : \mathbb{R} \to \mathbb{R}$ se dice **Lipschitz-continua** si existe un $C > 0$ tal que para todo $x, y \in \mathbb{R}$ se cumple que:

$$|f(x) - f(y)| \leq C|x - y|$$

¿Entonces?

Definición: función Lipschitz-continua

Una función $f : \mathbb{R} \to \mathbb{R}$ se dice **Lipschitz-continua** si existe un $C > 0$ tal que para todo $x, y \in \mathbb{R}$ se cumple que:

$$|f(x) - f(y)| \leq C|x - y|$$

¿Entonces?

- Lipschitz-continuidad implica continuidad

Definición: función Lipschitz-continua

Una función $f : \mathbb{R} \to \mathbb{R}$ se dice **Lipschitz-continua** si existe un $C > 0$ tal que para todo $x, y \in \mathbb{R}$ se cumple que:

$$|f(x) - f(y)| \leq C|x - y|$$

¿Entonces?

- Lipschitz-continuidad implica continuidad
- Todas las funciones de activación usadas en la práctica son Lipschitz-continuas: ReLU, clipped ReLU, sigmoide, sigmoide linearizada y *tanh*

Definición: clasificador *non-splitting*

Sea $\mathbb{D}(d)$ una distribución con media $\boldsymbol{\mu}$ y sea $\mathcal{M}$ una GCN usada para clasificación binaria de grafos. Definimos la secuencia de vectores $\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_T$ inductivamente como:

- $\boldsymbol{\mu}_0 := \boldsymbol{\mu}$
- $\boldsymbol{\mu}_t := \sigma(\mathbf{W}_n^{(t)} \boldsymbol{\mu}_{t-1} + \mathbf{b}^{(t)})$

El clasificador $\mathfrak{C} : \mathbb{R}^{d(T)} \to \mathbb{B}$ se dice **non-splitting** para $\mathcal{M}$ si el vector $\boldsymbol{\mu}_T$ **no** yace en una frontera de decisión de $\mathfrak{C}$

Definición: clasificador *non-splitting*

Sea $\mathbb{D}(d)$ una distribución con media $\boldsymbol{\mu}$ y sea $\mathcal{M}$ una GCN usada para clasificación binaria de grafos. Definimos la secuencia de vectores $\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_T$ inductivamente como:

- $\boldsymbol{\mu}_0 := \boldsymbol{\mu}$
- $\boldsymbol{\mu}_t := \sigma(\mathbf{W}_n^{(t)} \boldsymbol{\mu}_{t-1} + \mathbf{b}^{(t)})$

El clasificador $\mathfrak{C} : \mathbb{R}^{d(T)} \to \mathbb{B}$ se dice **non-splitting** para $\mathcal{M}$ si el vector $\boldsymbol{\mu}_T$ **no** yace en una frontera de decisión de $\mathfrak{C}$

¿Entonces?

Definición: clasificador *non-splitting*

Sea $\mathbb{D}(d)$ una distribución con media $\boldsymbol{\mu}$ y sea $\mathcal{M}$ una GCN usada para clasificación binaria de grafos. Definimos la secuencia de vectores $\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_T$ inductivamente como:

- $\boldsymbol{\mu}_0 := \boldsymbol{\mu}$
- $\boldsymbol{\mu}_t := \sigma(\mathbf{W}_n^{(t)} \boldsymbol{\mu}_{t-1} + \mathbf{b}^{(t)})$

El clasificador $\mathfrak{C} : \mathbb{R}^{d(T)} \to \mathbb{B}$ se dice **non-splitting** para $\mathcal{M}$ si el vector $\boldsymbol{\mu}_T$ **no** yace en una frontera de decisión de $\mathfrak{C}$

¿Entonces?

- Para todas las decisiones "razonables" de $\mathfrak{C}$, la frontera de decisión tiene dimensión menor a $d(T)$, por lo que es un conjunto de medida 0 $\to$ "insignificante"

Teorema 4.6

Sea $\mathcal{M}$ una GCN usada para clasificación binaria y sea $r \in [0, 1]$. Entonces, $\mathcal{M}$ **satisface una ley 0-1** con respecto a la distribución de grafos $\mathbb{G}(n, r)$ y la distribución de *features* $\mathbb{D}(d)$, si las siguientes **condiciones** se cumplen:

(i) la distribución $\mathbb{D}(d)$ es sub-Gaussiana

(ii) la función no-lineal σ es Lipschitz-continua

(iii) la representación a nivel de grafo usa *average pooling*

(iv) el clasificador $\mathfrak{C}$ usado por $\mathcal{M}$ es *non-splitting*

¿Cómo se demuestra el Teorema?

Lema 4.7

Sean $\mathcal{M}$ y $\mathbb{D}(d)$ tal que satisfacen las condiciones del Teorema 4.6. Luego, para cada *layer* t existe un $\mathbf{z}_t \in \mathbb{R}^{d(t)}$ tal que al muestrear aleatoriamente un grafo con *features* en sus nodos desde $\mathbb{G}(n, r)$ y $\mathbb{D}(d)$, para cada $i \in \{1, \ldots, d(t)\}$ y para cada $\epsilon > 0$ se tiene que:

$$\mathbb{P}\left(\forall v : \left|\left[\mathbf{x}_v^{(t)} - \mathbf{z}_t\right]_i\right| < \epsilon\right) \xrightarrow[n\to\infty]{} 1$$

Demostración (Lema): idea

Analizamos probabilísticamente las preactivaciones en cada *layer*:

Demostración (Lema): idea

Analizamos probabilísticamente las preactivaciones en cada *layer*:

- Usamos la **propiedad sub-Gaussiana** para mostrar que la desviación de cada una de las preactivaciones de la primera iteración $\mathbf{y}_v^{(1)}$ de su valor esperado disminuye a medida que el número de nodos n tiende a infinito

Demostración (Lema): idea

Analizamos probabilísticamente las preactivaciones en cada *layer*:

- Usamos la **propiedad sub-Gaussiana** para mostrar que la desviación de cada una de las preactivaciones de la primera iteración $\mathbf{y}_v^{(1)}$ de su valor esperado disminuye a medida que el número de nodos n tiende a infinito
- Usando esto y el hecho de que σ es **Lipschitz-continua**, mostramos que cada activación $\mathbf{x}_v^{(1)}$ tiende a un valor fijo

Demostración (Lema): idea

Analizamos probabilísticamente las preactivaciones en cada *layer*:

- Usamos la **propiedad sub-Gaussiana** para mostrar que la desviación de cada una de las preactivaciones de la primera iteración $\mathbf{y}_v^{(1)}$ de su valor esperado disminuye a medida que el número de nodos n tiende a infinito
- Usando esto y el hecho de que σ es **Lipschitz-continua**, mostramos que cada activación $\mathbf{x}_v^{(1)}$ tiende a un valor fijo
- Usamos un argumento de **inducción** para extender este argumento a todos los *layers* de la GCN

■

Demostración (Teorema)

Demostración (Teorema)

Por el Lema 4.7, los *node embeddings* $\mathbf{x}_v^T$ disminuyen su desviación de $\mathbf{z}_T$ a medida que el número de nodos n incrementa.

Demostración (Teorema)

Por el Lema 4.7, los *node embeddings* $\mathbf{x}_v^T$ disminuyen su desviación de $\mathbf{z}_T$ a medida que el número de nodos n incrementa.

- Luego, la representación por **average pooling** a nivel de grafo también se desvía cada vez menos de $\mathbf{z}_T$

Demostración (Teorema)

Por el Lema 4.7, los *node embeddings* $\mathbf{x}_v^T$ disminuyen su desviación de $\mathbf{z}_T$ a medida que el número de nodos n incrementa.

- Luego, la representación por **average pooling** a nivel de grafo también se desvía cada vez menos de $\mathbf{z}_T$
- De la demostración del Lema, este vector $\mathbf{z}_T$ es exactamente el vector $\boldsymbol{\mu}_T$ de la definición de un clasificador **non-splitting**

Demostración (Teorema)

Por el Lema 4.7, los *node embeddings* $\mathbf{x}_v^T$ disminuyen su desviación de $\mathbf{z}_T$ a medida que el número de nodos n incrementa.

- Luego, la representación por **average pooling** a nivel de grafo también se desvía cada vez menos de $\mathbf{z}_T$
- De la demostración del Lema, este vector $\mathbf{z}_T$ es exactamente el vector $\boldsymbol{\mu}_T$ de la definición de un clasificador **non-splitting**
- Por lo tanto, $\mathbf{z}_T$ no yace sobre una frontera de decisión del clasificador $\mathfrak{C}$, y existe un $\epsilon > 0$ tal que $\mathfrak{C}$ es constante en:

$$\{\boldsymbol{x} \in \mathbb{R}^{d(T)} \mid \forall i \in \{1, \ldots, d(T)\} : |[\boldsymbol{z}_T - \boldsymbol{x}]_i| < \epsilon\}$$

Demostración (Teorema)

Por el Lema 4.7, los *node embeddings* $\mathbf{x}_v^T$ disminuyen su desviación de $\mathbf{z}_T$ a medida que el número de nodos n incrementa.

- Luego, la representación por **average pooling** a nivel de grafo también se desvía cada vez menos de $\mathbf{z}_T$
- De la demostración del Lema, este vector $\mathbf{z}_T$ es exactamente el vector $\boldsymbol{\mu}_T$ de la definición de un clasificador **non-splitting**
- Por lo tanto, $\mathbf{z}_T$ no yace sobre una frontera de decisión del clasificador $\mathfrak{C}$, y existe un $\epsilon > 0$ tal que $\mathfrak{C}$ es constante en:

$$\{\boldsymbol{x} \in \mathbb{R}^{d(T)} \mid \forall i \in \{1, \ldots, d(T)\} : |[\boldsymbol{z}_T - \boldsymbol{x}]_i| < \epsilon\}$$

Finalmente, la probabilidad de que la salida de $\mathcal{M}$ sea $\mathfrak{C}(\boldsymbol{z}_T)$ tiende a 1 cuando n tiende a infinito.

■

Comentarios

Algunos comentarios:

Comentarios

Algunos comentarios:

- Se espera que la rapidez de esta convergencia dependa del **número de layers**, **dimensionalidad de los embeddings** y la elección de **función de activación**

Comentarios

Algunos comentarios:

- Se espera que la rapidez de esta convergencia dependa del **número de layers**, **dimensionalidad de los embeddings** y la elección de **función de activación**
- Dada la manera en la que se demuestra el Lema 4.7, se esperaría que la rapidez de convergencia disminuya a medida la dimensionalidad de los *embeddings* crece.

Comentarios

Algunos comentarios:

- Se espera que la rapidez de esta convergencia dependa del **número de layers**, **dimensionalidad de los embeddings** y la elección de **función de activación**
- Dada la manera en la que se demuestra el Lema 4.7, se esperaría que la rapidez de convergencia disminuya a medida la dimensionalidad de los *embeddings* crece.
  - **¿Por qué?**

Comentarios

Algunos comentarios:

- Se espera que la rapidez de esta convergencia dependa del **número de layers**, **dimensionalidad de los embeddings** y la elección de **función de activación**
- Dada la manera en la que se demuestra el Lema 4.7, se esperaría que la rapidez de convergencia disminuya a medida la dimensionalidad de los *embeddings* crece.
  - **¿Por qué?**
- Se puede demostrar de manera muy similar una ley 0-1 para el caso de **MeanGNN$^+$**, pero tomando en cuenta el término adicional de *global readout*

Tercer resultado:

SumGNN$^+$ satisfacen leyes 0-1

Teorema 4.10

Sea $\mathcal{M}$ una SumGNN$^+$ usada para clasificación binaria y sea $r \in [0, 1]$. Entonces, $\mathcal{M}$ **satisface una ley 0-1** con respecto a la distribución de grafos $\mathbb{G}(n, r)$ y la distribución de *features* $\mathbb{D}(d)$

Teorema 4.10

Sea $\mathcal{M}$ una SumGNN$^+$ usada para clasificación binaria y sea $r \in [0, 1]$. Entonces, $\mathcal{M}$ **satisface una ley 0-1** con respecto a la distribución de grafos $\mathbb{G}(n, r)$ y la distribución de *features* $\mathbb{D}(d)$, si las siguientes **condiciones** se cumplen:

(i) la distribución $\mathbb{D}(d)$ es sub-Gaussiana

(ii) la función no-lineal es eventualmente constante en ambas direcciones

(iii) la representación a nivel de grafo usa *average pooling* o *component-wise maximum pooling*

(iv) $\mathcal{M}$ es sincrónicamente saturante para $\mathbb{G}(n, r)$ y $\mathbb{D}(d)$

Teorema 4.10

Sea $\mathcal{M}$ una SumGNN$^+$ usada para clasificación binaria y sea $r \in [0, 1]$. Entonces, $\mathcal{M}$ **satisface una ley 0-1** con respecto a la distribución de grafos $\mathbb{G}(n, r)$ y la distribución de *features* $\mathbb{D}(d)$, si las siguientes **condiciones** se cumplen:

(i) la distribución $\mathbb{D}(d)$ es sub-Gaussiana

(ii) la función no-lineal es eventualmente constante en ambas direcciones

(iii) la representación a nivel de grafo usa *average pooling* o *component-wise maximum pooling*

(iv) $\mathcal{M}$ es sincrónicamente saturante para $\mathbb{G}(n, r)$ y $\mathbb{D}(d)$

¿Demasiadas condiciones?

Definición: función eventualmente constante

Una función $\sigma : \mathbb{R} \to \mathbb{R}$ es **eventualmente constante en ambas direcciones** si existen $x_{-\infty}, x_{\infty} \in \mathbb{R}$ tales que $\sigma(z)$ es constante para $z < x_{-\infty}$ y $\sigma(z)$ es constante para $z > x_{\infty}$. Denotamos por $\sigma_{-\infty}$ y σ_{∞} el mínimo y máximo respectivamente de una función eventualmente constante.

Definición: función eventualmente constante

Una función $\sigma : \mathbb{R} \to \mathbb{R}$ es **eventualmente constante en ambas direcciones** si existen $x_{-\infty}, x_{\infty} \in \mathbb{R}$ tales que $\sigma(z)$ es constante para $z < x_{-\infty}$ y $\sigma(z)$ es constante para $z > x_{\infty}$. Denotamos por $\sigma_{-\infty}$ y σ_{∞} el mínimo y máximo respectivamente de una función eventualmente constante.

¿Entonces?

Definición: función eventualmente constante

Una función $\sigma : \mathbb{R} \to \mathbb{R}$ es **eventualmente constante en ambas direcciones** si existen $x_{-\infty}, x_{\infty} \in \mathbb{R}$ tales que $\sigma(z)$ es constante para $z < x_{-\infty}$ y $\sigma(z)$ es constante para $z > x_{\infty}$. Denotamos por $\sigma_{-\infty}$ y σ_{∞} el mínimo y máximo respectivamente de una función eventualmente constante.

¿Entonces?

Definición: función eventualmente constante

Una función $\sigma : \mathbb{R} \to \mathbb{R}$ es **eventualmente constante en ambas direcciones** si existen $x_{-\infty}, x_{\infty} \in \mathbb{R}$ tales que $\sigma(z)$ es constante para $z < x_{-\infty}$ y $\sigma(z)$ es constante para $z > x_{\infty}$. Denotamos por $\sigma_{-\infty}$ y σ_{∞} el mínimo y máximo respectivamente de una función eventualmente constante.

¿Entonces?

- Existe un threshold para ambos lados más allá de los cuales σ es constante

Definición: función eventualmente constante

Una función $\sigma : \mathbb{R} \to \mathbb{R}$ es **eventualmente constante en ambas direcciones** si existen $x_{-\infty}, x_{\infty} \in \mathbb{R}$ tales que $\sigma(z)$ es constante para $z < x_{-\infty}$ y $\sigma(z)$ es constante para $z > x_{\infty}$. Denotamos por $\sigma_{-\infty}$ y σ_{∞} el mínimo y máximo respectivamente de una función eventualmente constante.

¿Entonces?

- Existe un threshold para ambos lados más allá de los cuales σ es constante
- La función sigmoide linearizada y clipped ReLU son eventualmente constantes en ambas direcciones

Definición: función eventualmente constante

Una función $\sigma : \mathbb{R} \to \mathbb{R}$ es **eventualmente constante en ambas direcciones** si existen $x_{-\infty}, x_{\infty} \in \mathbb{R}$ tales que $\sigma(z)$ es constante para $z < x_{-\infty}$ y $\sigma(z)$ es constante para $z > x_{\infty}$. Denotamos por $\sigma_{-\infty}$ y σ_{∞} el mínimo y máximo respectivamente de una función eventualmente constante.

¿Entonces?

- Existe un threshold para ambos lados más allá de los cuales σ es constante
- La función sigmoide linearizada y clipped ReLU son eventualmente constantes en ambas direcciones
- Más aún, cuando se trabaja con precisión finita, cualquier función con gradiente *vanishing* (como la sigmoide) puede tratarse como eventualmente constante en ambas direcciones

Definición: sincrónicamente saturante

Sea $\mathcal{M}$ una SumGNN$^+$ para clasificación binaria de grafos con una función no lineal σ que es eventualmente constante en ambas direcciones. Sean $\mathbb{D}(d)$ una distribución con media $\boldsymbol{\mu}$ y $r \in [0,1]$. Luego, el modelo $\mathcal{M}$ es **sincrónicamente saturante** para $\mathbb{G}(n,r)$ y $\mathbb{D}(d)$ si las siguientes condiciones se cumplen:

(i) Para cada $1 \leq i \leq d(1)$:

$$\left[(r\mathbf{W}_n^{(1)} + \mathbf{W}_g^{(1)})\boldsymbol{\mu}\right]_i \neq 0$$

(ii) Para cada *layer* $1 < t \leq T$, cada $1 \leq i \leq d(t)$ y cada $\boldsymbol{z} \in \{\sigma_{-\infty}, \sigma_{\infty}\}^{d(t-1)}$

$$\left[(r\mathbf{W}_n^{(t)} + \mathbf{W}_g^{(t)})\boldsymbol{z}\right]_i \neq 0$$

Definición: sincrónicamente saturante

El comportamiento asintótico de modelos SumGNN$^+$ está determinado por matrices $\mathbf{Q}_t := r\mathbf{W}_n^{(t)} + \mathbf{W}_g^{(t)}$

Definición: sincrónicamente saturante

El comportamiento asintótico de modelos SumGNN$^+$ está determinado por matrices $\mathbf{Q}_t := r\mathbf{W}_n^{(t)} + \mathbf{W}_g^{(t)}$

- Los *embeddings* asintóticos en el último *layer* están dados por:

$$\sigma(\mathbf{Q}_T(\sigma(\mathbf{Q}_{T-1}\cdots\sigma(\mathbf{Q_0}\boldsymbol{\mu})\cdot)))$$

Definición: sincrónicamente saturante

El comportamiento asintótico de modelos SumGNN$^+$ está determinado por matrices $\mathbf{Q}_t := r\mathbf{W}_n^{(t)} + \mathbf{W}_g^{(t)}$

- Los *embeddings* asintóticos en el último *layer* están dados por:

$$\sigma(\mathbf{Q}_T(\sigma(\mathbf{Q}_{T-1}\cdots\sigma(\mathbf{Q_0}\boldsymbol{\mu})\cdot)))$$

Ser sincrónicamente saturante significa evitar el caso borde donde alguno de estos pasos tiene una componente que es cero.

Definición: sincrónicamente saturante

El comportamiento asintótico de modelos SumGNN$^+$ está determinado por matrices $\mathbf{Q}_t := r\mathbf{W}_n^{(t)} + \mathbf{W}_g^{(t)}$

- Los *embeddings* asintóticos en el último *layer* están dados por:

$$\sigma(\mathbf{Q}_T(\sigma(\mathbf{Q}_{T-1}\cdots\sigma(\mathbf{Q_0}\boldsymbol{\mu})\cdot)))$$

Ser sincrónicamente saturante significa evitar el caso borde donde alguno de estos pasos tiene una componente que es cero.

- El espacio de modelos que no son sincrónicamente saturantes tiene menor dimensión que el espacio de todos los modelos, por lo que tiene medida 0.

Teorema 4.10

Sea $\mathcal{M}$ una SumGNN$^+$ usada para clasificación binaria y sea $r \in [0, 1]$. Entonces, $\mathcal{M}$ **satisface una ley 0-1** con respecto a la distribución de grafos $\mathbb{G}(n, r)$ y la distribución de *features* $\mathbb{D}(d)$, si las siguientes **condiciones** se cumplen:

(i) la distribución $\mathbb{D}(d)$ es sub-Gaussiana

(ii) la función no-lineal es eventualmente constante en ambas direcciones

(iii) la representación a nivel de grafo usa *average pooling* o *component-wise maximum pooling*

(iv) $\mathcal{M}$ es sincrónicamente saturante para $\mathbb{G}(n, r)$ y $\mathbb{D}(d)$

¿Cómo se demuestra el Teorema?

Lema 4.11

Sean $\mathcal{M}$, $\mathbb{D}(d)$ y r como en el Teorema 4.10 y sean $\sigma_{-\infty}$ y σ_{∞} los valores extremos de la función de activación. Luego, para cada *layer* t existe un $\mathbf{z}_t \in \{\sigma_{-\infty}, \sigma_{\infty}\}^{d(t)}$ tal que al muestrear grafos con *features* en sus nodos aleatoriamente desde $\mathbb{G}(n, r)$ y $\mathbb{D}(d)$, la probabilidad de que $\mathbf{x}_v^{(t)} = \mathbf{z}_t$ para cada nodo v tiende a 1 cuando n tiende a infinito.

Lema 4.11

Sean $\mathcal{M}$, $\mathbb{D}(d)$ y r como en el Teorema 4.10 y sean $\sigma_{-\infty}$ y σ_{∞} los valores extremos de la función de activación. Luego, para cada *layer* t existe un $\mathbf{z}_t \in \{\sigma_{-\infty}, \sigma_{\infty}\}^{d(t)}$ tal que al muestrear grafos con *features* en sus nodos aleatoriamente desde $\mathbb{G}(n, r)$ y $\mathbb{D}(d)$, la probabilidad de que $\mathbf{x}_v^{(t)} = \mathbf{z}_t$ para cada nodo v tiende a 1 cuando n tiende a infinito.

- La demostración es similar al lema anterior, mostrando que **el valor de cada preactivación tiende a infinito** a medida que incrementa el número de nodos

Lema 4.11

Sean $\mathcal{M}$, $\mathbb{D}(d)$ y r como en el Teorema 4.10 y sean $\sigma_{-\infty}$ y σ_{∞} los valores extremos de la función de activación. Luego, para cada *layer* t existe un $\mathbf{z}_t \in \{\sigma_{-\infty}, \sigma_{\infty}\}^{d(t)}$ tal que al muestrear grafos con *features* en sus nodos aleatoriamente desde $\mathbb{G}(n, r)$ y $\mathbb{D}(d)$, la probabilidad de que $\mathbf{x}_v^{(t)} = \mathbf{z}_t$ para cada nodo v tiende a 1 cuando n tiende a infinito.

- La demostración es similar al lema anterior, mostrando que **el valor de cada preactivación tiende a infinito** a medida que incrementa el número de nodos
  - De hecho, la probabilidad de que este valor caiga bajo cualquier valor fijo tiende exponencialmente a 0

Lema 4.11

Sean $\mathcal{M}$, $\mathbb{D}(d)$ y r como en el Teorema 4.10 y sean $\sigma_{-\infty}$ y σ_{∞} los valores extremos de la función de activación. Luego, para cada *layer* t existe un $\mathbf{z}_t \in \{\sigma_{-\infty}, \sigma_{\infty}\}^{d(t)}$ tal que al muestrear grafos con *features* en sus nodos aleatoriamente desde $\mathbb{G}(n, r)$ y $\mathbb{D}(d)$, la probabilidad de que $\mathbf{x}_v^{(t)} = \mathbf{z}_t$ para cada nodo v tiende a 1 cuando n tiende a infinito.

- La demostración es similar al lema anterior, mostrando que **el valor de cada preactivación tiende a infinito** a medida que incrementa el número de nodos
  - De hecho, la probabilidad de que este valor caiga bajo cualquier valor fijo tiende exponencialmente a 0
- El **paso inductivo** utiliza el hecho de que $\mathcal{M}$ es sincrónicamente saturante

GNNs con features aleatorios en sus nodos

SumGNN$^+$ con features aleatorios en los nodos

Hasta ahora hemos considerado un grafo junto con los *features* de sus nodos como el input (aleatorio) para las GNNs

SumGNN$^+$ con features aleatorios en los nodos

Hasta ahora hemos considerado un grafo junto con los *features* de sus nodos como el input (aleatorio) para las GNNs

- Cambiamos la perspectiva y ahora **consideramos los features iniciales de los nodos como parte del modelo**

SumGNN$^+$ con features aleatorios en los nodos

Hasta ahora hemos considerado un grafo junto con los *features* de sus nodos como el input (aleatorio) para las GNNs

- Cambiamos la perspectiva y ahora **consideramos los features iniciales de los nodos como parte del modelo**
- Nos concentramos en modelos **SumGNN$^+$**

SumGNN$^+$ con features aleatorios en los nodos

Hasta ahora hemos considerado un grafo junto con los *features* de sus nodos como el input (aleatorio) para las GNNs

- Cambiamos la perspectiva y ahora **consideramos los features iniciales de los nodos como parte del modelo**
- Nos concentramos en modelos **SumGNN$^+$**
- Buscamos la **clase de funciones que se pueden aproximar por estos modelos**

Definición: aproximación de funciones

Sean f una función Booleana sobre grafos, ζ una función aleatoria sobre grafos y $\delta > 0$. Luego, decimos que **ζ δ-aproxima uniformemente a f** si:

$$\forall n \in \mathbb{N} : \mathbb{P}(\zeta(G) = f(G) \mid |G| = n) \geq 1 - \delta$$

cuando muestreamos $G \sim \mathbb{G}(n, 1/2)$

Definición: aproximación de funciones

Sean f una función Booleana sobre grafos, ζ una función aleatoria sobre grafos y $\delta > 0$. Luego, decimos que **ζ δ-aproxima uniformemente a f** si:

$$\forall n \in \mathbb{N} : \mathbb{P}(\zeta(G) = f(G) \mid |G| = n) \geq 1 - \delta$$

cuando muestreamos $G \sim \mathbb{G}(n, 1/2)$

- $\mathbb{G}(n, 1/2)$ es la distribución bajo la cual todos los grafos con n nodos son igual de probables

Teorema 5.2

Sea ξ un invariante de grafos que satisface una ley 0-1 respecto a $\mathbb{G}(n, 1/2)$. Entonces, para cada $\delta > 0$ existe una SumGNN$^+$ con features aleatorios en los nodos que δ-aproxima uniformemente a ξ

Teorema 5.2

Sea ξ un invariante de grafos que satisface una ley 0-1 respecto a $\mathbb{G}(n, 1/2)$. Entonces, para cada $\delta > 0$ existe una SumGNN$^+$ con features aleatorios en los nodos que δ-aproxima uniformemente a ξ

- Este es un **converso parcial** para el Teorema 4.10

Resultados experimentales

Setup experimental

Se realizaron experimentos para GCN, MeanGNN y SumGNN considerando:

- **10 modelos de cada arquitectura** (por cada número de *layers*) con pesos inicializados aleatoriamente, no-linearidad eventualmente constante en ambas direcciones y *pooling* por promedio
- **Clasificador binario estándar** para todos los modelos
- **Distribuciones** $\mathbb{G}(n, 1/2)$ para grafos y $U(0, 1)$ para *features*
- Cada arquitectura con 1, 2 y 3 *layers*

Resultados principales

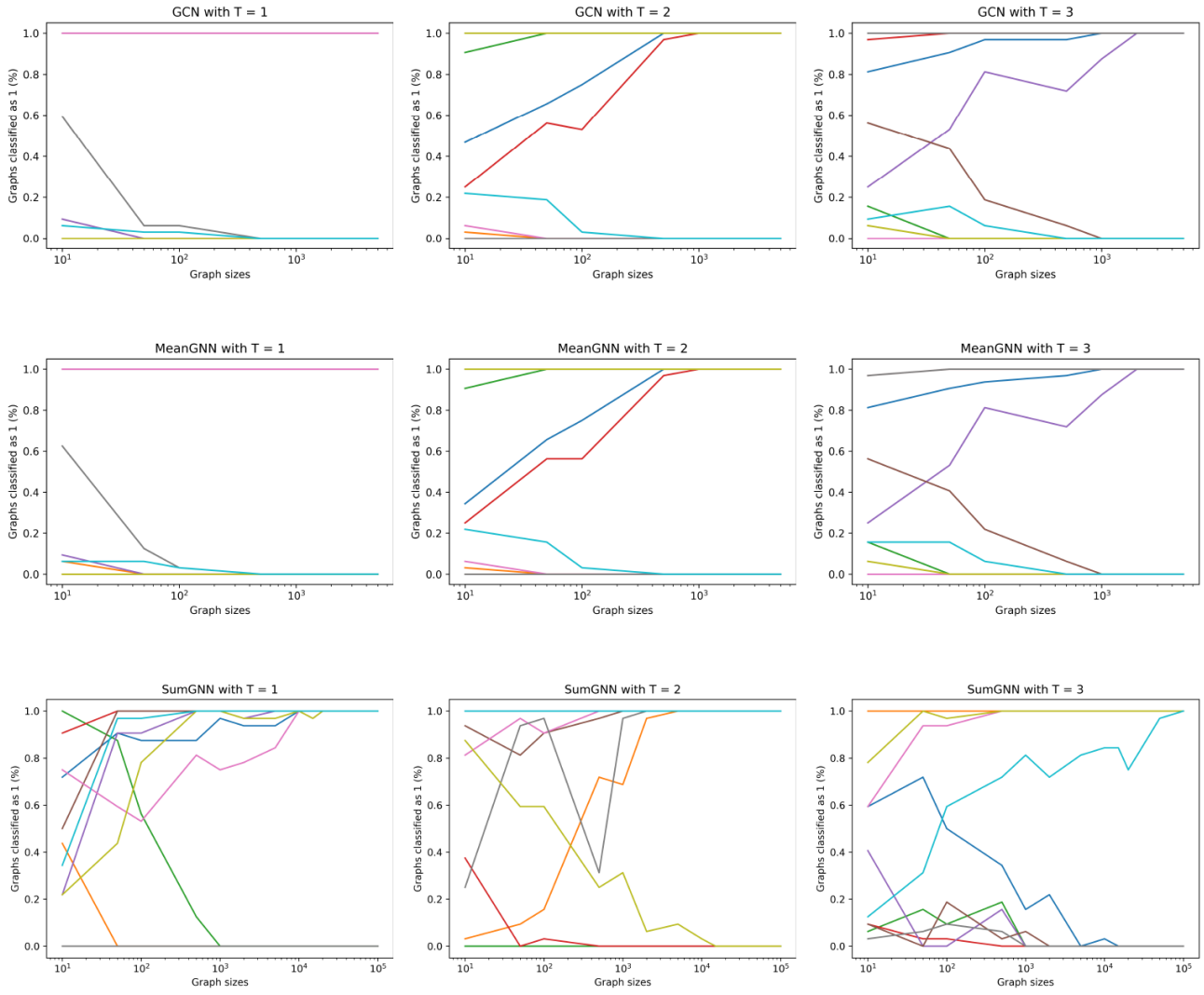

Figure: Resultados principales del paper

Conclusiones

Limitaciones y trabajo futuro

Las principales limitaciones del trabajo son:

Limitaciones y trabajo futuro

Las principales limitaciones del trabajo son:

- Condiciones puestas sobre los modelos en los teoremas principales

Limitaciones y trabajo futuro

Las principales limitaciones del trabajo son:

- Condiciones puestas sobre los modelos en los teoremas principales
- Uso de la distribución de Erdős–Rényi
  - **Buena** desde la perspectiva de la **expresividad**
  - **Mala** desde la perspectiva de la **extrapolación**

Limitaciones y trabajo futuro

Las principales limitaciones del trabajo son:

- Condiciones puestas sobre los modelos en los teoremas principales
- Uso de la distribución de Erdős–Rényi
  - **Buena** desde la perspectiva de la **expresividad**
  - **Mala** desde la perspectiva de la **extrapolación**
- Falta estudio sobre las tasas de convergencia

Limitaciones y trabajo futuro

Las principales limitaciones del trabajo son:

- Condiciones puestas sobre los modelos en los teoremas principales
- Uso de la distribución de Erdős–Rényi
  - **Buena** desde la perspectiva de la **expresividad**
  - **Mala** desde la perspectiva de la **extrapolación**
- Falta estudio sobre las tasas de convergencia
- El Teorema 5.2 requiere que la distribución sea $\mathbb{G}(n, 1/2)$
  - De poder relajarse, se tendría una completa caracterización del poder expresivo de estos modelos

Conclusiones

Conclusiones

- Estudiamos cómo se comportan las GNNs cuando el número de nodos de los grafos de input crece arbitrariamente

Conclusiones

- Estudiamos cómo se comportan las GNNs cuando el número de nodos de los grafos de input crece arbitrariamente
- Cota superior para el poder expresivo de las GNNs
  - Cualquier propiedad que puede ser *uniformemente* expresada por una GNN debe obedecer una ley 0-1

Conclusiones

- Estudiamos cómo se comportan las GNNs cuando el número de nodos de los grafos de input crece arbitrariamente
- Cota superior para el poder expresivo de las GNNs
  - Cualquier propiedad que puede ser *uniformemente* expresada por una GNN debe obedecer una ley 0-1
- Cota inferior para su poder expresivo
  - Converso del Teorema 5.2

Conclusiones

- Estudiamos cómo se comportan las GNNs cuando el número de nodos de los grafos de input crece arbitrariamente
- Cota superior para el poder expresivo de las GNNs
  - Cualquier propiedad que puede ser *uniformemente* expresada por una GNN debe obedecer una ley 0-1
- Cota inferior para su poder expresivo
  - Converso del Teorema 5.2
- Los resultados no dependen de la inicialización aleatoria de los modelos, de su entrenamiento ni de su número de *layers*

Conclusiones

- Estudiamos cómo se comportan las GNNs cuando el número de nodos de los grafos de input crece arbitrariamente
- Cota superior para el poder expresivo de las GNNs
  - Cualquier propiedad que puede ser *uniformemente* expresada por una GNN debe obedecer una ley 0-1
- Cota inferior para su poder expresivo
  - Converso del Teorema 5.2
- Los resultados no dependen de la inicialización aleatoria de los modelos, de su entrenamiento ni de su número de *layers*
- Los resultados se comprobaron empíricamente

Key takeaway

Key takeaway

"Our result, combined with the manifest success of GNNs in practice, suggests that zero-one laws must be abundant in nature"

Zero-One Laws of Graph Neural Networks

Benjamín Álvarez Stevenson

November 2024

Anexos

Resultados adicionales: features normales

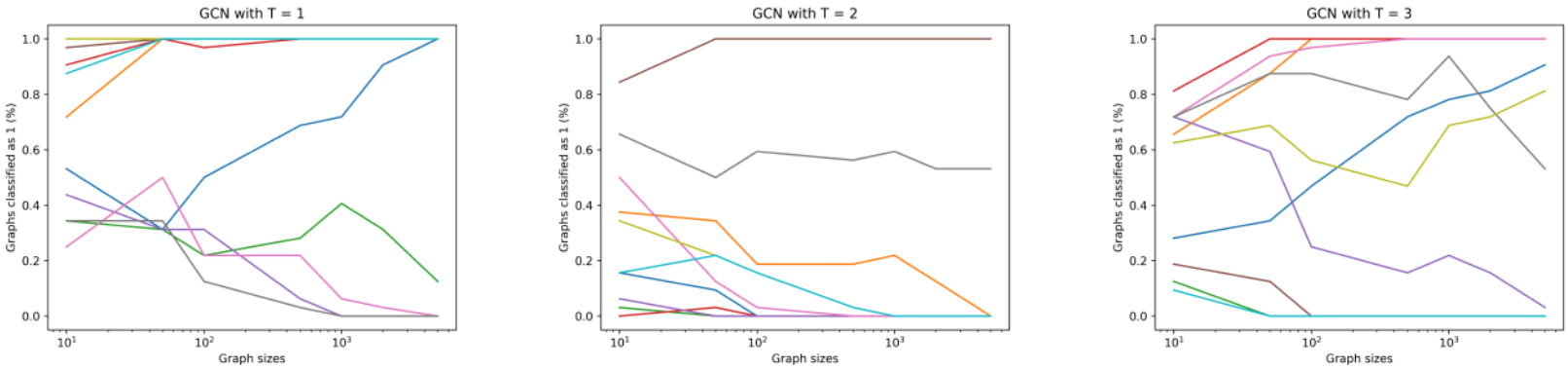

Figure: Features muestreados de una distribución normal

Resultados adicionales: función ReLU

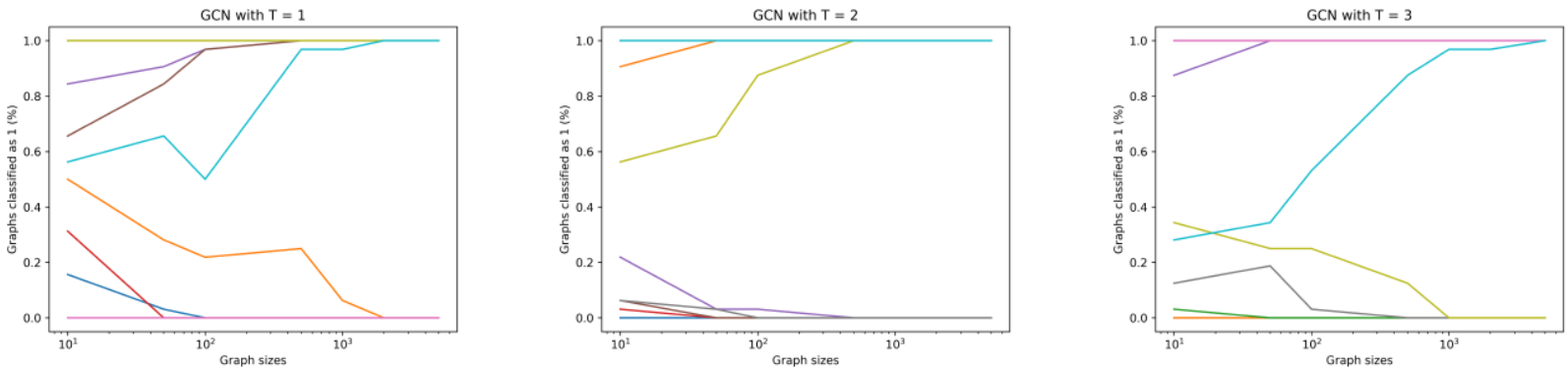

Figure: Función de activación ReLU

Resultados adicionales: función *tanh*

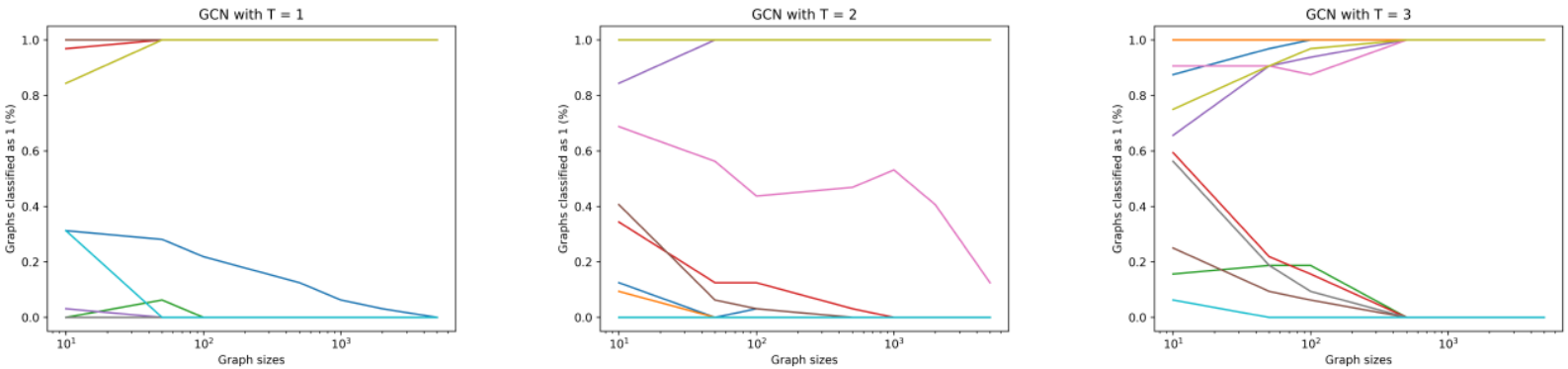

Figure: Función de activación *tanh*

Resultados adicionales: función sigmoide

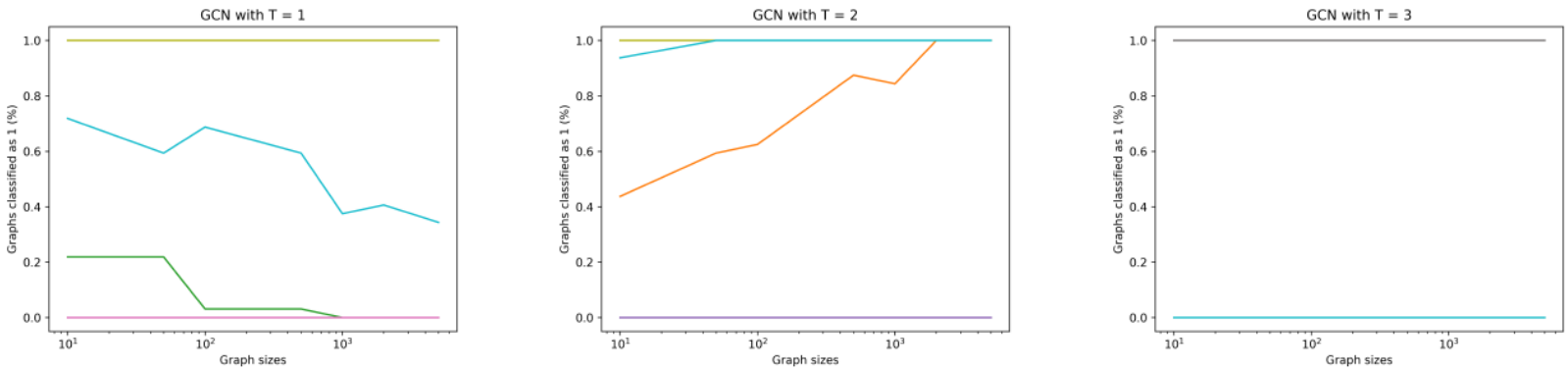

Figure: Función de activación sigmoide

Resultados adicionales: arquitectura GAT

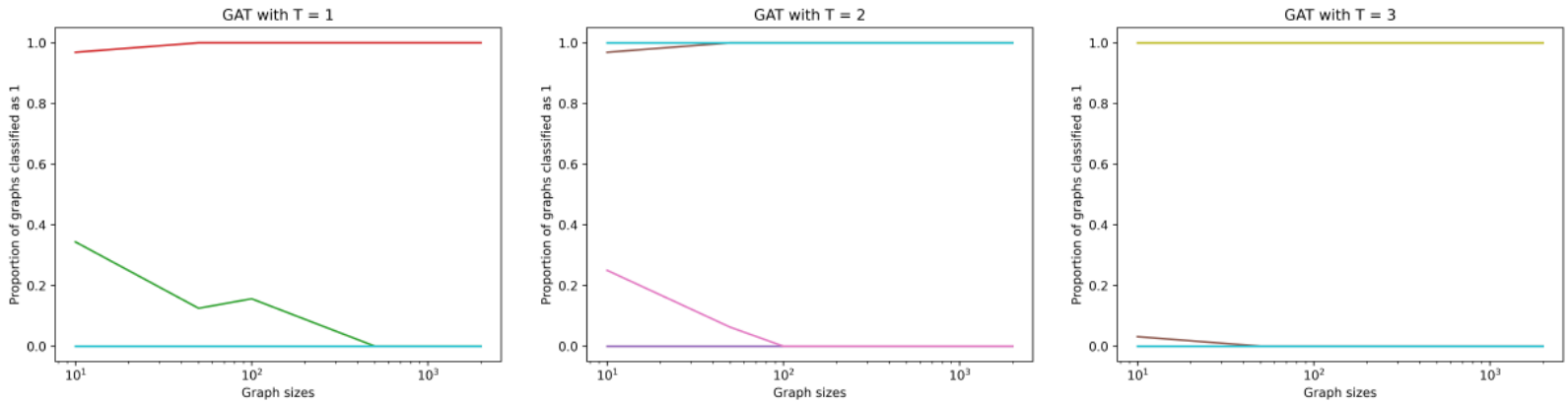

Figure: Arquitectura GAT

Resultados adicionales: grafos *sparse*

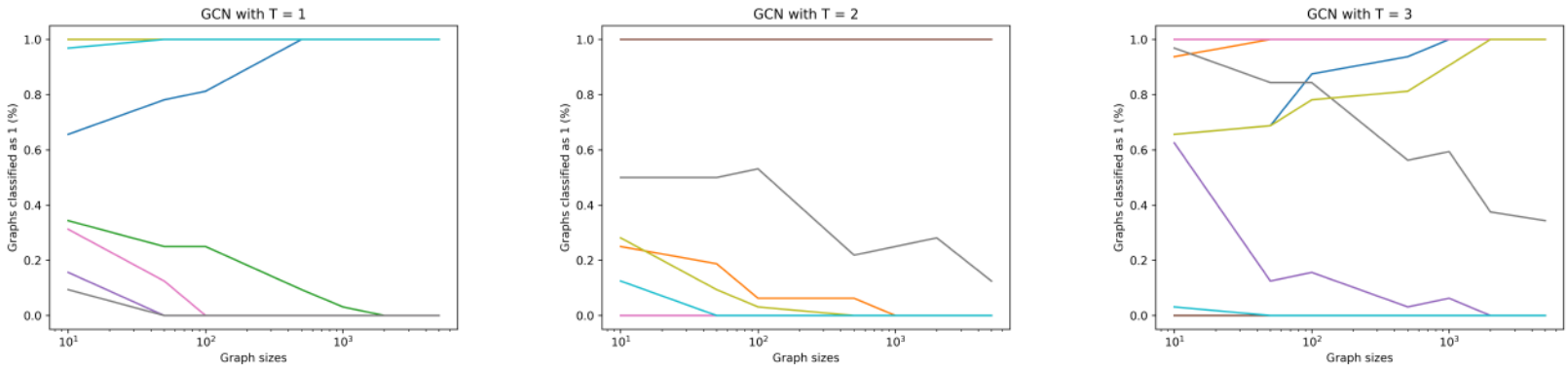

Figure: Grafos *sparse*

Resultados adicionales: grafos Barabási-Albert

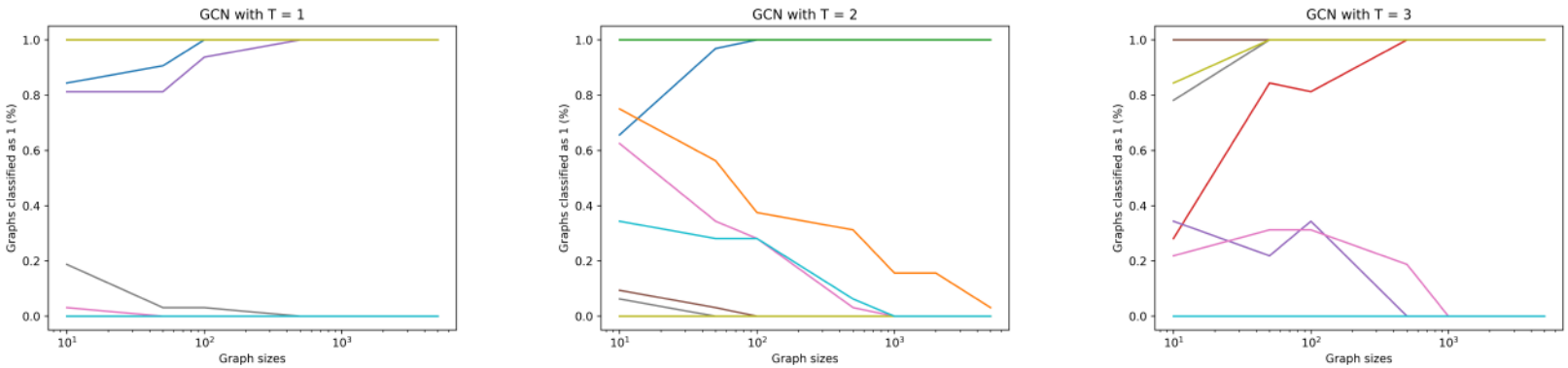

Figure: Grafos Barabási-Albert