FACULTAD DE ESTUDIOS ESTADÍSTICOS

MÁSTER EN CIENCIA DE DATOS E INTELIGENCIA DE NEGOCIOS

Curso 2024 / 2025

Trabajo de Fin de Máster

TÍTULO: Polarización estructural en grafos y redes sociales: un estudio temporal sobre OpenAI y DeepSeek en X

***Alumna*: Senxiao Xu**

***Tutor:* Juan Antonio Guevara Gil**

Junio de 2025

UNIVERSIDAD COMPLUTENSE MADRID

Declaración Responsable sobre Autoría y Uso Ético de Herramientas de Inteligencia Artificial (IA)

Yo, Xu Senxiao

Con DNI/NIE/PASAPORTE: X7615410H

declaro de manera responsable que el/la presente:

- Trabajo de Fin de Máster (TFM)

Titulado/a

Polarización estructural en grafos y redes sociales: un estudio temporal sobre OpenAI y DeepSeek en X

es el resultado de mi trabajo intelectual personal y creativo, y ha sido elaborado de acuerdo con los principios éticos y las normas de integridad vigentes en la comunidad académica y, más específicamente, en la Universidad Complutense de Madrid.

Soy, pues, autor del material aquí incluido y, cuando no ha sido así y he tomado el material de otra fuente, lo he citado o bien he declarado su procedencia de forma clara -incluidas, en su caso, herramientas de inteligencia artificial-. Las ideas y aportaciones principales incluidas en este trabajo, y que acreditan la adquisición de competencias, son mías y no proceden de otras fuentes o han sido reescritas usando material de otras fuentes.

Asimismo, aseguro que los datos y recursos utilizados son legítimos, verificables y han sido obtenidos de fuentes confiables y autorizadas. Además, he tomado medidas para garantizar la confidencialidad y privacidad de los datos utilizados, evitando cualquier tipo de sesgo o discriminación injusta en el tratamiento de la información.

En Madrid a 27 de junio de 2025

FIRMA 

RESUMEN

Este Trabajo de Fin de Master se centra en el análisis de la polarización estructural en grafos, tomando como el caso de estudio de la interacción de usuarios en la plataforma X (anteriormente Twitter) en torno a los modelos de lenguaje OpenAI y DeepSeek. Para ello, se recopilaron datos manualmente a lo largo de once semanas mediante técnicas de web scraping, identificando usuarios que mencionan o responden a cuentas relacionadas con estos modelos. A partir de dicha información, se construyó una red no dirigida de usuarios y se aplicaron métodos de análisis de grafos, polarización basada en fuzzy-sets: JDJ y análisis temporal.

Una vez definida una partición forzada de la red en dos comunidades principales, se estimaron los grados de pertenencia ideológica de cada nodo usando la proporción de vecinos en cada comunidad. Esta información permitió calcular el índice JDJ, una medida difusa de polarización ideológica desarrollada en trabajos previos. El análisis presentó niveles de polarización estructural altos y persistentes, con momentos puntuales de convergencia entre comunidades, reflejados tanto en la visualización de grafos como en la evolución del índice JDJ.

Los resultados muestran que la red estudiada evoluciona hacia una estructura de tipo *small-world*, con comunidades cada vez más cohesivas y a veces separadas. Asimismo, se observa que los valores del índice JDJ reflejan con precisión los cambios estructurales y discursivos detectados en la red. El estudio aporta una contribución metodológica al combinar técnicas de análisis estructural, lógica difusa y visualización de datos en el estudio de procesos de polarización ideológica en contextos reales.

ABSTRACT

This thesis focuses on the analysis of ideological polarization in graphs, using user interactions on the platform X (Twitter) as a case study, specifically regarding the discourse surrounding OpenAI and DeepSeek language models. Data was manually collected over eleven weeks using web scraping techniques, identifying users who mentioned or replied to accounts related to these models. Based on this information, an undirected user network was constructed, and methods of graph analysis, fuzzy-set-based polarization (JDJ index), and temporal analysis were applied.

After defining a forced partition of the network into two main communities, each node's ideological affiliation degree was estimated based on the proportion of its neighbors belonging to each community. This information enabled the calculation of the JDJ index, a fuzzy measure of ideological polarization developed in previous works. The analysis revealed consistently high levels of structural polarization, with occasional moments of convergence between communities, observable both in the graph visualizations and in the evolution of the JDJ index.

The results indicate that the network evolves towards a small-world structure, with increasingly cohesive and distinct communities. Furthermore, the JDJ index accurately captures both structural and discursive changes within the network. This study offers a methodological contribution by combining structural network analysis, fuzzy logic, and data visualization techniques to examine ideological polarization processes in real-world contexts.

ÍNDICE

1. INTRODUCCIÓN

1.1 Contextualización del problema: polarización en redes sociales

En los últimos años, las redes sociales se han convertido en un entorno fundamental de discursos públicos, el debate político, la divulgación científica y la interacción cultural. Plataformas como Twitter (actualmente X) permiten a millones de usuarios conectarse, intercambiar opiniones y construir comunidades en torno a temas de interés. Sin embargo, esta conectividad masiva también ha facilitado la aparición de fenómenos de polarización, que consiste en que las personas tienden a rodearse de quienes comparten su visión, generando comunidades con pocos vínculos hacia el exterior.

El análisis estructural de estas dinámicas suele representarse mediante grafos, donde los nodos son individuos y las aristas representan interacciones como menciones o respuestas. A través de métricas como la **modularidad** (Newman, 2006), el **coeficiente de agrupamiento** o la **centralidad** (Freeman, L. C. (1979)), se puede cuantificar el grado de cohesión interna y desconexión entre grupos. No obstante, estas métricas tradicionales, aunque útiles, pueden resultar insuficientes para capturar los matices de pertenencia y ambigüedad.

Frente a este reto, se han propuesto enfoques que incorporan el uso de **conjuntos difusos (fuzzy sets),** fue propuesto por Zadeh (1965), que permite que un mismo nodo pertenezca a varias comunidades con diferentes grados de intensidad. Este enfoque es útil porque permite captar situaciones en las que un usuario se identifica principalmente con una comunidad, pero aún mantiene cierta interacción o afinidad con la otra, algo común en las redes sociales, donde los límites no siempre son tan estrictos.

En este marco, el **índice JDJ** (Guevara et al. (2020)) surge como una medida avanzada para medir la polarización en redes bipartitas o forzadas a dos comunidades. A diferencia de otras métricas centradas únicamente en la densidad interna o la frontera entre grupos, el JDJ considera tanto la radicalidad de los individuos como su grado de afinidad cruzada.

Este trabajo se centra este enfoque estructural, aplicando técnicas de análisis de redes y lógica difusa para examinar la evolución de la polarización en un escenario específico. A través del grafo de menciones en Twitter (X), se reconstruyen las comunidades emergentes y se mide, semana a semana, la intensidad y la estructura de dicha polarización.

1.2 Objetivos del trabajo

El trabajo tiene como finalidad analizar la polarización estructural en una red de menciones extraída de Twitter (X), centrada en el debate generado en torno a los modelos de inteligencia artificial **OpenAI** y **DeepSeek**. Esta elección responde al interés de observar en qué medida los usuarios se agrupan o se fragmentan en función de su posicionamiento respecto a estos dos modelos, los cuales representan enfoques contrastantes en cuanto a filosofía, accesibilidad y control: OpenAI como un referente

de modelos cerrados y corporativos, y DeepSeek como una alternativa open-source impulsada por la comunidad. Este enfoque permite estudiar la polarización digital no solo como una propiedad estructural de la red, sino también como un reflejo del debate técnico y social en torno al desarrollo de la inteligencia artificial.

Para ello, se utiliza un conjunto de datos recopilado entre los meses de enero y abril de 2025, compuesto por tweets, menciones y usuarios, y se construye un grafo no dirigido que representa la interacción entre usuarios a lo largo del tiempo. Sobre esta estructura, se aplican diversas técnicas de análisis de redes con el propósito de estudiar tanto su organización general como el grado de segmentación en comunidades diferenciadas.

El análisis parte de la elaboración de un grafo completo de menciones, construido a partir de datos extraídos mediante la herramienta Web Data Research Assistant (WDRA). Esta plataforma permite capturar de forma estructurada la actividad de usuarios en Twitter, generando archivos HTML que contienen información sobre autores, textos, menciones y marcas temporales. A través del procesamiento de estos archivos se construye una red no dirigida de interacciones, sobre la cual se aplican algoritmos de detección de comunidades —como el método de Louvain (Blondel et al. (2008)) — y se calculan diversas métricas de centralidad (Freeman, L. C. (1979), incluyendo grado, intermediación, cercanía y autovector, con el fin de caracterizar la relevancia estructural de los nodos.

Posteriormente, se fuerza una partición binaria de la red en dos comunidades principales, correspondientes a los polos discursivos de OpenAI y DeepSeek, mediante el algoritmo Fast-Greedy (Clauset, Newman y Moore (2004)). Esta división permite comparar la estructura impuesta con la segmentación natural de la red y analizar la dinámica de polarización desde una doble perspectiva.

A continuación, se estudian las diferencias entre ambas comunidades en términos de vocabulario empleado (hashtags), idioma predominante y composición de perfiles, así como su comportamiento estructural (tamaño, densidad o presencia de hubs). Este análisis se complementa con la aplicación del índice JDJ (Guevara et al. (2020)), una medida basada en lógica difusa que cuantifica la polarización según el grado de interacción de cada nodo con miembros de su misma comunidad y de la opuesta. Dicha métrica permite representar con mayor precisión los matices de afinidad parcial o ambigüedad que se dan en contextos reales de interacción social.

Finalmente, se incorpora una dimensión temporal al análisis mediante la reconstrucción acumulativa del grafo semana a semana. Esto permite observar la evolución de distintas métricas estructurales, como el número de nodos, la densidad, el coeficiente de agrupamiento o la propiedad de pequeño mundo (Humphries y Gurney (2008)), así como el comportamiento del índice JDJ (Guevara et al. (2020)) a lo largo del tiempo. El conjunto de estos elementos ofrece una visión detallada del fenómeno de la polarización desde una perspectiva estructural, cuantitativa y dinámica.

1.3 Motivación del análisis centrado en OpenAI y DeepSeek

La elección de OpenAI y DeepSeek como ejes del presente análisis se fundamenta en su protagonismo actual dentro del ecosistema de inteligencia artificial generativa, así como en el contraste discursivo que representan. OpenAI, conocida por desarrollar modelos ampliamente adoptados como ChatGPT o DALL·E, ha mantenido una estrategia basada en el control centralizado, con políticas restrictivas de acceso y una orientación comercial claramente definida. Frente a esta postura, DeepSeek ha emergido recientemente como una alternativa que promueve el desarrollo open source, lo que ha despertado el interés de comunidades técnicas y de investigación que valoran la transparencia, la reproducibilidad y el acceso abierto al conocimiento.

Este contraste entre dos modelos —uno cerrado y corporativo, y otro abierto y comunitario— no solo refleja diferencias en estrategias de desarrollo tecnológico, sino que también activa posicionamientos simbólicos, ideológicos y culturales entre los usuarios. En Twitter (X), estas posturas se manifiestan a través de interacciones, menciones, retuits y respuestas que, aunque dispersas en apariencia, configuran estructuras reconocibles a través del análisis de redes.

El seguimiento de estas interacciones permite observar cómo se forman comunidades, se refuerzan afinidades y se consolidan bloques discursivos, incluso en debates que no son explícitamente políticos. En este contexto, el caso OpenAI–DeepSeek representa un ejemplo especialmente adecuado para analizar cómo la polarización puede emerger en torno a temas tecnológicos, y cómo esta segmentación se refleja en la forma de la conversación digital.

Otro punto a favor de este caso es que se trata de un debate reciente y en plena evolución, lo que permite observar cómo la polarización estructural cambia con el tiempo. Así, no solo se puede analizar cómo están formadas las comunidades en un momento dado, sino también cómo se transforman, se intensifican o se reorganizan a medida que surgen nuevos eventos o cambia el contexto. Por eso, este escenario resulta ideal para aplicar herramientas como el análisis de grafos, la detección de comunidades y métodos de medición de polarización basados en conjuntos difusos, como los propuestos por Zadeh (1965).

2. MARCO TEÓRICO

2.1 Teoría de grafos y análisis de redes sociales

Un grafo es una estructura matemática compuesta por un conjunto de nodos (también llamados vértices) y un conjunto de enlaces (o aristas) que conectan pares de nodos.

En este contexto, los nodos suelen representar autores —como individuos, cuentas o instituciones— mientras que las aristas reflejan interacciones entre ellos,

como menciones, respuestas o vínculos de amistad. Formalmente, un grafo se denota como $G = (V, E)$, donde V es el conjunto de nodos y E el conjunto de aristas.

Dependiendo del tipo de relación, las aristas pueden tener una dirección. Por ejemplo, si un usuario menciona a otro en Twitter, se genera una relación dirigida. En cambio, si se trata de una relación mutua, como una amistad bidireccional, hablamos de un grafo no dirigido. En el análisis ha optado por modelar las menciones como relaciones no dirigidas, con el objetivo de centrar el estudio en la estructura general de interacción, más que en la direccionalidad de cada mención.

Partiendo de esta representación, es posible calcular diversas métricas que describen el comportamiento de los nodos y la estructura global de la red. Entre estas se incluyen las medidas de centralidad, los algoritmos de detección de comunidades y los índices de polarización, que serán abordados en las secciones siguientes.

2.1.1 Centralidades: grado, cercanía, intermediación, autovector

El análisis de centralidad en redes sociales, propuesto por Freeman, L. C. (1979), permite identificar los nodos más relevantes dentro de una estructura relacional. Existen diversas medidas que capturan distintas nociones de "importancia" o "influencia", y su elección depende del tipo de red y del fenómeno que se desea estudiar. A continuación, se exponen las medidas más empleadas: grado, cercanía, intermediación y autovector.

Centralidad de grado

La centralidad de grado es una de las métricas más simples y directas. Indica cuántos vínculos directos tiene un nodo. En redes no dirigidas, se corresponde con el número total de conexiones; en redes dirigidas, puede distinguirse entre grado de entrada (*in-degree*) y de salida (*out-degree*).

Formalmente, para un grafo no dirigido $G = (V, E)$, la centralidad de grado $C_D(i)$ de un nodo i se define como:

$$C_D(i) = \deg(i) = \sum_{j \in V} a_{ij}$$

donde a_{ij} es el elemento (i, j) de la matriz de adyacencia A, que toma valor 1 si existe una arista entre i y j, y 0 en caso contrario.

En el análisis aplicado, esta medida permite detectar los usuarios que reciben o emiten más menciones, lo cual puede interpretarse como una forma básica de visibilidad.

Centralidad de cercanía

La centralidad de cercanía evalúa la facilidad con la que un nodo puede alcanzar a los demás en la red. Se basa en la suma de las distancias geodésicas (más cortas) desde un nodo hacia todos los demás. Cuanto menor sea esta suma, mayor será su cercanía.

La fórmula habitual para la centralidad de cercanía $C_C(i)$ es:

$$C_C(i) = \frac{1}{\sum_{j \in V} \pi_{ij}}$$

donde π_{ij} representa la longitud del camino más corto entre los nodos i y j. Esta medida requiere que la red esté conectada (o considerar solo la componente gigante), ya que las distancias infinitas imposibilitan su cálculo.

En el caso de estudio, se utiliza para identificar usuarios "estratégicamente" bien ubicados, que pueden difundir información de forma eficiente.

Centralidad de intermediación

La centralidad de intermediación cuantifica en qué medida un nodo actúa como puente en los caminos más cortos que conectan pares de nodos en la red.

Para un grafo no dirigido $G = (V, E)$, la centralidad de intermediación $C_B(k)$ de un nodo k se define como:

$$C_B(k) = \sum_{i<j,\ i,j \neq k} \frac{\sigma_{ij}(k)}{\sigma_{ij}}$$

donde σ_{ij} es el número total de caminos más cortos entre los nodos i y j, y $\sigma_{ij}(k)$ representa cuántos de esos caminos pasan por el nodo k. Esta medida captura la capacidad de un nodo para actuar como intermediario clave en la red, un valor alto indica que el nodo conecta muchas otras parejas de nodos a través de las rutas más eficientes.

En el contexto del estudio, esta métrica permite identificar usuarios que facilitan la circulación de información entre distintas partes de la red.

Centralidad de autovector

La centralidad de autovector evalúa la relevancia de un nodo no solo en función del número de conexiones que posee, sino también considerando la importancia de los nodos a los que está conectado. Esta métrica parte de la idea de que estar vinculado a nodos influyentes incrementa la propia influencia de un nodo dentro de la red.

Matemáticamente, esta centralidad se define como el valor propio principal (autovalor dominante) de la matriz de adyacencia. Sea A la matriz de adyacencia del grafo y $\vec{x}$ el vector de centralidades, entonces:

$$A\vec{x} = \lambda\vec{x}$$

donde λ es el mayor autovalor asociado y $\vec{x}$ el vector propio correspondiente. Cada componente x_i del vector representa la centralidad de autovector del nodo i.

Esta medida resulta especialmente útil para identificar actores que, aunque puedan no tener muchas conexiones directas, están conectados con otros actores

altamente centrales, lo que les otorga un papel destacado en la estructura relacional de la red.

2.1.2 Detección de comunidades: Louvain y Fast-Greedy

La detección de comunidades es una de las tareas fundamentales en el análisis estructural de redes sociales. Su objetivo consiste en identificar subconjuntos de nodos densamente conectados entre sí, pero con pocas conexiones hacia el resto de la red. Entre los métodos más empleados se encuentran los algoritmos de **Louvain** y **Fast-Greedy**, ambos basados en la maximización de la modularidad.

Algoritmo de Louvain

El algoritmo de **Louvain** es uno de los métodos más utilizados para la detección de comunidades en redes complejas, especialmente por su eficiencia y escalabilidad. Fue propuesto por Blondel et al. (2008) y se fundamenta en la **optimización de la modularidad Q**, una métrica que compara la densidad de enlaces intra-comunidad con la esperada en una red aleatoria con igual distribución de grados.

Este algoritmo adopta un enfoque jerárquico y consta de dos fases iterativas. En la primera fase, cada nodo se asigna inicialmente a su propia comunidad. Luego, de manera secuencial, se evalúa para cada nodo el efecto de trasladarlo a la comunidad de uno de sus vecinos, y se lleva a cabo el cambio únicamente si este movimiento incrementa la modularidad total de la red. Este proceso se repite hasta alcanzar un óptimo local en el que no puede mejorarse la modularidad a través de más reasignaciones.

Una vez alcanzado este estado, comienza la segunda fase. En ella, se construye una nueva red en la que cada nodo representa una comunidad detectada en la fase anterior, y los enlaces entre estas nuevas unidades se ponderan según el número de conexiones existentes entre los nodos originales. A partir de esta red agregada, se repite el proceso de optimización hasta que la modularidad global deja de mejorar significativamente.

A diferencia de otros métodos jerárquicos, el algoritmo de Louvain no requiere que se especifique de antemano el número de comunidades. Además, es eficaz para redes grandes, por su estructura recursiva y su bajo coste computacional.

En cuanto al fundamento matemático, el algoritmo se basa en calcular la **variación de modularidad** ΔQ que implica mover un nodo de una comunidad a otra. Esta variación puede expresarse mediante la siguiente fórmula:

$$\Delta Q = \left[\frac{\sum_{in} 1 + 2k_{i,in}}{2m} - \left(\frac{\sum_{tot} 1 + k_i}{2m}\right)^2\right] - \left[\frac{\sum_{in} 1}{2m} - \left(\frac{\sum_{tot} 1}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2\right]$$

Donde:

- k_i: grado del nodo i,

- $k_{i,in}$: suma de los pesos de los enlaces entre el nodo i y los nodos de la comunidad objetivo,
- $\sum_{in} 1$: suma de los pesos de los enlaces internos dentro de la comunidad,
- $\sum_{tot} 1$: suma de los grados de todos los nodos en la comunidad,
- m : número total de enlaces en la red (o suma total de pesos si es ponderada).

Algoritmo Fast-Greedy

El algoritmo de detección de comunidades **Fast Greedy**, propuesto por Clauset, Newman y Moore (2004), está basado en una estrategia jerárquica aglomerativa. Su objetivo es **maximizar la modularidad** a través de un proceso iterativo que comienza con cada nodo formando su propia comunidad individual, y va fusionando aquellas comunidades cuya unión incrementa más el valor de la modularidad total del grafo.

A diferencia del algoritmo Louvain, que puede dividir y reagrupar comunidades dinámicamente en múltiples fases, Fast Greedy sigue una estructura jerárquica estricta de fusión ascendente (bottom-up). En cada paso, se calcula el cambio en la modularidad que resultaría de fusionar cualquier par de comunidades, y se selecciona la fusión que más incrementa el valor de Q. Este proceso continúa hasta que no es posible mejorar más la modularidad, o hasta que se alcanza una estructura jerárquica deseada (por ejemplo, un número predeterminado de comunidades).

La función objetivo que optimiza este algoritmo es la **modularidad global** Q, definida como:

$$Q = \frac{1}{2m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(c_i, c_j)$$

Donde:

- A_{ij} es la matriz de adyacencia (1 si hay un enlace entre i y j, 0 en caso contrario)
- k_i y k_j son los grados de los nodos i y j,
- m es el número total de enlaces en la red.
- $\delta(c_i, c_j)$ es la función de Kronecker que vale 1 si los nodos i y j están en la misma comunidad y 0 en caso contrario.

Ambos métodos se han aplicado en este estudio, Louvain para detectar comunidades naturales emergentes, y Fast-Greedy para forzar una partición en dos grupos principales y facilitar así el análisis de polarización.

2.1.3 Análisis topológico de una red

El análisis topológico de redes sociales consiste en estudiar la estructura global y local de la red para identificar patrones de conexión, propiedades estructurales y fenómenos sociales subyacentes. Esta etapa es fundamental para comprender cómo se relacionan los elementos de la red (en este caso, usuarios de Twitter), y para facilitar análisis posteriores como la detección de comunidades, polarización o centralidad.

A continuación, se describen las principales métricas utilizadas:

a) Tamaño de la red: Hace referencia al número de nodos $|V|$ y aristas $|E|$. Un nodo representa una entidad (usuario) y una arista indica la existencia de una interacción (por ejemplo, una mención entre usuarios).

- El **número de nodos** es el cardinal del conjunto V.
- El **número de aristas** es el cardinal del conjunto E, y representa el número de relaciones observadas.

Este dato permite tener una primera idea de la escala del grafo, siendo más complejas las redes con mayor tamaño.

b) Densidad: La densidad cuantifica el grado de conexión global de la red. Se calcula como la proporción de enlaces existentes respecto al número total posible de enlaces si la red estuviera completamente conectada.

$$d(G) = \frac{|E|}{\frac{|V|(|V|-1)}{2}}$$

Un valor de $d(G)$ = 1 indica que todos los nodos están conectados entre sí (grafo completo), mientras que valores cercanos a 0 indican una red muy dispersa.

c) Coeficiente de agrupación (Clustering Coefficient): Esta métrica evalúa hasta qué punto los vecinos de un nodo también están conectados entre sí, reflejando la tendencia a formar triángulos o cliques.

Para un nodo i, se calcula como:

$$C_i = \frac{L_i}{k_i(k_i-1)/2}$$

Donde:

- k_i es el número de vecinos del nodo i (su grado),
- L_i es el número de enlaces reales entre los vecinos de i.

El **clustering global** de la red se obtiene promediando los valores de todos los nodos:

$$C(G) = \frac{1}{|V|}\sum_{i \in V} C_i$$

Esta medida proporciona información sobre la **cohesión local** de la red. En redes sociales, un alto valor de $C(G)$ suele indicar que los usuarios tienden a formar grupos cerrados.

d) Diámetro: El diámetro de un grafo es la mayor distancia mínima entre dos nodos cualquiera de la red. Es decir, la longitud del camino más corto más largo:

$$\text{Dia}(G) = \max_{\{i,j\} \in V} \pi_{ij}$$

Donde π_{ij} es la longitud del camino mínimo entre los nodos i y j.

Este indicador mide la extensión estructural máxima de la red.

e) Distancia media: Indica la media de las distancias mínimas entre todos los pares de nodos conectados. Mide la eficiencia global de la red en términos de transmisión de información:

$$\text{AverSP}(G) = \frac{1}{M} \sum_{\{i,j\} \in V, \pi_{ij} < \infty} \pi_{ij}$$

donde:

- M es el número de pares de nodos que están conectados,
- π_{ij} es la distancia mínima entre i y j.

En redes sociales, valores bajos indican una fuerte conectividad y facilidad de propagación de información.

f) Homofilia: La homofilia refleja la tendencia de los nodos a conectarse con otros que comparten características similares. En redes sociales, esto puede expresarse en términos ideológicos, lingüísticos o culturales. Su análisis se basa en correlaciones entre atributos de los nodos conectados.

Por ejemplo, si X_i y Y_i representan el grado de los nodos origen y destino de cada arista i, se puede calcular un coeficiente de correlación para evaluar si los nodos de alto grado tienden a conectarse con otros también de alto grado.

Una correlación positiva indica homofilia (similares se conectan), mientras que una correlación negativa sugiere heterofilia (opuestos se conectan).

g) Componentes conexas: Una componente conexa es un subconjunto de nodos entre los que existe al menos un camino. La componente gigante es la más grande, y suele ser el objeto principal de análisis, ya que representa la parte de la red donde se concentra la mayor actividad.

2.1.4 Tipos de Redes en Grafos

En el análisis estructural de redes sociales, es importante comprender las distintas estructuras topológicas que pueden tener los grafos, ya que cada una de ellas posee propiedades particulares que afectan la forma en que fluye la información. A continuación, se presentan tres tipos fundamentales de redes: redes regulares, redes aleatorias y redes de mundo pequeño.

a) **Redes Regulares:** Una red regular es aquella en la que cada nodo está conectado con un número fijo de vecinos, típicamente de forma simétrica. Estas

redes tienen una alta agrupación local y una alta longitud promedio de los caminos más cortos entre nodos. Si bien son estructuralmente ordenadas, presentan baja eficiencia global en la propagación de información.

b) Redes Aleatorias: Introducidas por Erdős y Rényi (1959), las redes aleatorias se construyen conectando pares de nodos con una probabilidad uniforme. Estas redes presentan una longitud media de camino baja y un bajo coeficiente de agrupamiento, con una distribución de grados próxima a una Poisson. Aunque permiten una propagación eficiente, carecen de estructura modular o comunidades definidas.

c) Redes de Mundo Pequeño: Watts y Strogatz (1998) propusieron un modelo intermedio entre las redes regulares y las aleatorias, conocido como red de mundo pequeño. Estas redes conservan un alto coeficiente de agrupamiento como las regulares, pero tienen una longitud de camino promedio baja, como las aleatorias. Se caracterizan por una mezcla de orden local y accesibilidad global, lo que las hace eficientes para la difusión de información.

La transición desde una red regular hacia una red aleatoria, introduciendo un grado creciente de aleatoriedad en las conexiones, da lugar al fenómeno de mundo pequeño. Este proceso se ilustra en la Figura 2.1, adaptada del trabajo original de Watts y Strogatz:

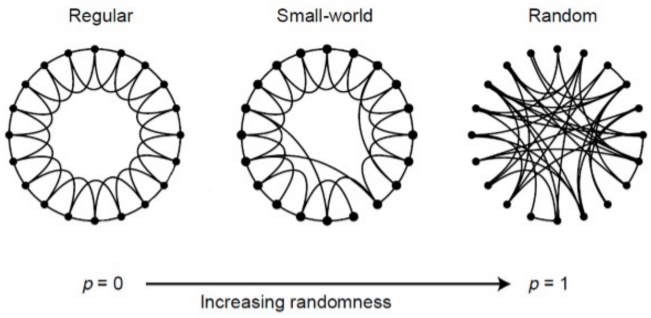


***Imagen 2.1**. Construcción de redes de mundo pequeño. Adaptado de (Watts & Strogatz, 1998)*

Una forma cuantitativa de determinar si una red posee propiedades de mundo pequeño es mediante el índice σ, definido por Humphries y Gurney (2008). Este índice compara la red observada con una red aleatoria de igual número de nodos y enlaces:

$$\sigma = \frac{C_{\text{obs}}/C_{\text{rand}}}{L_{\text{obs}}/L_{\text{rand}}}$$

Donde:

- C_{obs} es el coeficiente de agrupamiento global de la red observada.
- C_{rand} es el coeficiente de agrupamiento de una red aleatoria equivalente.
- L_{obs} es la longitud media del camino más corto en la red observada.

- L_{rand} es la longitud media del camino más corto en la red aleatoria.

Un valor de $\sigma > 1$ indica la presencia de estructura de mundo pequeño, mientras que valores cercanos a 1 corresponden a redes aleatorias.

d) **Redes de Mundo Pequeño Escala-Libre:** Una categoría particular son las redes de mundo pequeño escala-libre, que además de presentar alto agrupamiento y caminos cortos, exhiben una distribución de grados con cola pesada, típicamente siguiendo una ley de potencia. Estas redes tienen unos pocos nodos (hubs) con un alto número de conexiones y muchos nodos con pocas conexiones, lo que refleja una organización jerárquica eficiente. Barabási y Albert (1999) formalizaron este tipo de redes mediante un modelo de crecimiento con preferencia de conexión, dando lugar a estructuras robustas pero vulnerables a ataques dirigidos.

2.2 Polarización en redes sociales: definiciones y consecuencias

La polarización se expresa cuando los individuos dentro de una sociedad se agrupan en comunidades con opiniones cada vez más alejadas entre sí, reduciendo los puntos intermedios y dificultando el consenso. Este fenómeno se ha visto amplificado por el entorno digital, donde las redes sociales facilitan la creación de espacios ideológicamente homogéneos.

La polarización en contextos digitales no debe entenderse únicamente como una expresión de opiniones extremas, sino también como un patrón estructural observable en la forma en que se relacionan los usuarios. A través del análisis de redes, es posible detectar comunidades cohesionadas interna y escasamente conectadas entre sí, lo cual refleja una segmentación ideológica significativa. Cuando los grupos no se conectan entre sí, se crean cámaras de eco, que son espacios donde la gente solo encuentra opiniones similares a las suyas. Esto refuerza sus ideas y hace más difícil escuchar otras opiniones.

Las consecuencias de este fenómeno son diversas. Desde una perspectiva social, la polarización dificulta la deliberación pública, fomenta la radicalización y puede intensificar conflictos políticos. En términos comunicativos, limita la exposición a opiniones divergentes y reduce la pluralidad del debate.

Con el objetivo de estudiar y cuantificar este fenómeno, se han desarrollado diversas métricas. Algunas se centran en la estructura de la red, mientras que otras analizan las posiciones ideológicas de los individuos. En los siguientes apartados se presentan dos enfoques complementarios para evaluar la polarización: la modularidad, como métrica estructural, y el índice JDJ, basado en lógica difusa.

2.2.1 Modularidad

La modularidad, propuesta por Newman (2006), es una de las métricas más utilizadas en el análisis de redes para evaluar la calidad de una partición en

comunidades. Es decir, mide cómo de bien está dividida una red en grupos o subredes que presentan una alta densidad de conexiones internas y una baja densidad de conexiones externas. Esta característica es especialmente relevante cuando se estudian redes sociales donde se sospecha la existencia de polarización, ya que esta suele manifestarse estructuralmente como una separación marcada entre grupos.

El método de la modularidad parte de comparar la estructura observada de la red con una estructura esperada si los enlaces fueran distribuidos al azar, manteniendo los grados de los nodos constantes. La idea es que, si una red presenta una estructura interna muy ordenada —donde los nodos de un grupo se conectan mucho entre sí y muy poco con nodos de otros grupos—, el valor de la modularidad será alto. Por el contrario, si los enlaces están repartidos sin un patrón claro o están distribuidos de manera homogénea entre todos los nodos, el valor de modularidad será bajo.

Una manera intuitiva de entender esta métrica es que una modularidad alta indica que la partición elegida (es decir, la forma en la que se agrupan los nodos) coincide bien con la forma real en la que el grafo está conectado. En otras palabras, hay muchas aristas dentro de cada grupo y pocas entre grupos. Esta propiedad permite identificar comunidades naturales dentro de una red, sin necesidad de conocerlas previamente, lo cual es fundamental en el estudio de fenómenos como la polarización.

En los casos prácticos, se ha observado que los usuarios que comparten una ideología tienden a interactuar mayoritariamente entre sí, evitando interacciones con usuarios de ideologías opuestas. Esto da lugar a una estructura de red altamente modular, en la que las comunidades ideológicas están claramente separadas. Cuando la modularidad supera ciertos valores (por ejemplo, 0.3 o 0.4), suele interpretarse como un indicio de segmentación relevante, y por tanto de polarización estructural.

Además, la modularidad permite realizar comparaciones entre distintas redes o momentos temporales, ayudando a evaluar si la polarización está aumentando, disminuyendo o manteniéndose estable. También puede utilizarse en combinación con algoritmos de detección de comunidades, como el método de Louvain, para obtener divisiones óptimas del grafo.

Por todo ello, la modularidad no solo cumple una función técnica dentro del análisis de redes, sino que también constituye una medida clave para entender cómo se organiza el debate público en entornos digitales y qué tan fragmentada está la sociedad en términos de interacción y exposición a ideas diferentes.

Matemáticamente, la modularidad se expresa como:

$$Q = \frac{1}{2m}\sum_{i,j}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(c_i, c_j)$$

En esta fórmula, A_{ij} representa la matriz de adyacencia de la red, es decir, si existe una conexión entre los nodos i y j. Los términos k_i y k_j corresponden al grado de

cada nodo, m es el número total de enlaces, y $\delta(c_i, c_j)$ es una función que vale 1 si los nodos i y j pertenecen a la misma comunidad, y 0 en caso contrario.

2.2.2 Polarización basada en fuzzy-sets: JDJ

Los enfoques clásicos para estudiar la polarización suelen asumir que los individuos pertenecen claramente a uno u otro grupo, como si las posiciones ideológicas fueran completamente dicotómicas. Sin embargo, en contextos sociales reales, muchas personas presentan ideas intermedias, coinciden parcialmente con más de una postura, o incluso mantienen posiciones ambiguas. Para abordar esta complejidad, Zadeh (1965) propuso la teoría de los conjuntos difusos (*fuzzy sets*), permitiendo representar grados parciales de pertenencia a distintas categorías, lo que ha dado lugar a modelos más flexibles para representar identidades ideológicas parciales en el análisis social.

Desde esta perspectiva, el índice JDJ constituye una propuesta metodológica innovadora introducida por Guevara et al. (2020) y posteriormente extendida al análisis de redes sociales. Esta medida se basa en dos conceptos clave, los grados de pertenencia a polos ideológicos y los operadores de agregación, en particular una función de *overlapping* (ϕ) y una función de *grouping* (φ).

A diferencia de otras métricas centradas en la estructura de red (como el número de enlaces entre grupos), JDJ se basa en la distribución ideológica de los nodos. Cada individuo tiene asociados dos valores en el intervalo $[0,1]$ que representan su grado de pertenencia a los polos extremos de una variable ideológica, por ejemplo, izquierda X_A y derecha X_B. Dichos valores se representan como $\eta_{XA}(i)$ y $\eta_{XB}(j)$, donde i es el nodo o individuo. Estos grados de pertenencia no están normalizados; es decir, $\eta_{XA}(i) + \eta_{XB}(i)$ no tiene por qué sumar 1, permitiendo así representar ambigüedad o neutralidad.

A partir de esta formulación, el índice JDJ se define formalmente (Guevara et al., 2020) como:

$$\text{JDJ}_{\text{pol}}(V, \eta_{XA}, \eta_{XB}, \varphi, \phi) = \sum_{i,j \in V} \varphi\left(\phi\left(\eta_{XA}(i), \eta_{XB}(j)\right), \phi\left(\eta_{XB}(i), \eta_{XA}(j)\right)\right)$$

Donde:

- V es la población de individuos.
- $\eta_{XA}(i)$, $\eta_{XB}(j)$ son los grados de pertenencia del individuo i a los polos X_A y X_B.
- ϕ: $[0,1]^2 \rightarrow [0,1]$ es un operador de agregación (*overlapping*), que mide la coincidencia cruzada entre polos opuestos.
- φ: $[0,1]^2 \rightarrow [0,1]$ es una función de agrupación, que combina ambos resultados para cada par (i, j).

Casos ilustrativos:

- **Caso 1**: i está cerca de X_A y j de X_B: $\phi(1,1) = 1$, $\phi(0,0) = 0 \rightarrow \varphi(1,0) = 1 \Rightarrow$ **alta polarización**.

- **Caso 2**: ambos tienen posiciones neutras: $\phi(0.5, 0.5) = 0.5$, $\varphi(0.5, 0.5) = 0.5 \Rightarrow$ **polarización media**.
- **Caso 3**: ambos están cerca del mismo polo: $\phi(1,0) = 0$, $\phi(0,1) = 0$, $\varphi(0,0) = 0 \Rightarrow$ **polarización nula**.

Este enfoque tiene la ventaja de capturar los matices entre los extremos ideológicos y las zonas grises. Si la red se divide en dos bloques ideológicamente definidos, JDJ tomará valores altos. Si hay una mayoría de individuos con posiciones moderadas, JDJ será bajo, incluso si existen comunidades estructurales en la red.

Versión grupal

En escenarios donde los individuos se agrupan (e.g., por comunidades detectadas en una red), se puede usar una versión grupal computacionalmente más eficiente:

$$JDJ_g(X) = \sum_{i=1}^{K}\sum_{j=1}^{K} \pi_i \pi_j \phi\left(\mu_{X_A}(i), \mu_{X_B}(j)\right)$$

Donde:

- π_i y π_j representan las proporciones de individuos en los grupos i y j.
- $\mu_{X_A}(i)$ y $\mu_{X_B}(j)$ son los grados medios de pertenencia a los polos ideológicos en cada grupo.

Normalización e implementación en grafos

Cuando se dispone de grados de pertenencia individuales, es posible definir versiones **normalizadas** del índice JDJ para facilitar la interpretación y comparación en distintos contextos. Estas versiones toman valores en el intervalo $[0,1]$.

- Para individuos:

$$IJDJ = \frac{JDJ}{N^2} \cdot 2$$

- Para grupos, en escenarios con grados de pertenencia conocidos:

$$IJDJ_g = JDJ_g \cdot 4$$

No obstante, **si no se dispone de los grados de pertenencia**, y únicamente se conoce el tamaño de los grupos resultantes de una partición binaria de la red, se define la polarización como:

$$\text{JDJ}_{\text{pol}}(V, E, P, G_O, G_G) = 4 \cdot \frac{|C_1|}{|V|} \cdot \frac{|C_2|}{|V|} = 4 \cdot \text{Var}\big(\text{Ber}(p)\big)$$

Donde:

- C_1 y C_2 son las dos comunidades,
- $p = \frac{|C_1|}{|V|}$ es la proporción de individuos en una de ellas.

En el contexto de redes con lógica difusa, cuando se parte de una partición estructural, pero se quiere capturar ambigüedad en la afiliación ideológica, se emplea la versión difusa del índice JDJ en grafos. En este caso, se transforma la partición binaria en una partición difusa mediante una función de *make fuzzy*, definida como:

$$\mu_{C_i}(v) = \frac{|N_k(v) \cap C_i|}{|N_k(v)|}, \quad \text{donde} \quad N_k(v) = \{u \in V \mid \text{dist}(u, v) \leq k\}$$

La medida final se expresa como:

$$\text{JDJ}_{\text{pol,gre,k}}(V, E, P, G_O, G_G) = \text{JDJ}_{\text{pol}}\left(V, E, \widetilde{P_{\text{gre}}}, G_O, G_G\right)$$

Donde $\widetilde{P_{gre}} = (\mu_1, \mu_2)$ es la partición difusa resultante.

El índice JDJ ha demostrado ser especialmente útil en estudios donde no basta con observar la fragmentación estructural, sino que es necesario capturar matices ideológicos en los discursos.

2.2.3 Ejemplo del cálculo de JDJ en grafos

En este apartado se expone ejemplos detallado del cálculo del índice de polarización JDJ basado en lógica difusa, aplicado a un grafo simple con tres y cuatro nodos. A partir de la estructura del grafo, se obtienen los grados de pertenencia de cada nodo a dos comunidades (A y B), considerando la proporción de vecinos en cada una. Posteriormente, se evalúa el índice JDJ.

Cálculo del grado de pertenencia de los nodos

Dado un grafo no dirigido con una partición binaria (por ejemplo, comunidad A y B), se define el grado de pertenencia de un nodo i a la comunidad A como la proporción de sus vecinos —incluyéndose a sí mismo— que pertenecen a dicha comunidad. Formalmente:

$$\mu_A(i) = \frac{\text{Nº } vecinos(i) \text{ } en \text{ } ComunidadA}{\text{Nº vecinos total}}$$

$$\mu_B(i) = 1 - \mu_A(i)$$

Esta estrategia de asignación difusa se conoce como algoritmo *make fuzzy*, propuesto por Gregory (2011), y permite generar comunidades difusas a partir de particiones binarias.

Ejemplo 1: Nodos parcialmente polarizados

Se presenta un ejemplo simple y visualmente claro para ilustrar el cálculo del índice de polarización JDJ. El objetivo es mostrar cómo a partir de los grados de

pertenencia de cada nodo a dos comunidades (A y B), se construye una matriz de polarización y se obtiene el valor final del índice.

Grados de pertenencia

Consideramos tres nodos u_1, u_2, u_3, con los siguientes grados de pertenencia a las comunidades A y B:

Nodo	μ_A	μ_B
u_1	1	0
u_2	0,5	0,5
u_3	0	1

***Tabla 2.1**. Grados de pertenencia de los nodos*

Construcción de la matriz de polarización

Utilizando los operadores de agregación:

$$\phi(x, y) = x \cdot y(producto)$$

$$\varphi(x, y) = max(x, y)$$

Se calcula el valor de polarización para cada par de nodos (i, j) como:

$$P_{ij} = \varphi\left(\phi\big(\mu_A(i), \mu_B(j)\big), \phi\big(\mu_B(i), \mu_A(j)\big)\right)$$

La matriz resultante de valores de polarización es:

$$P_{ij} = \begin{bmatrix} 0 & 0.5 & 1 \\ 0.5 & 0.25 & 0.5 \\ 1 & 0.5 & 0 \end{bmatrix}$$

Cálculo del índice JDJ

La suma total de los valores en la matriz de polarización es:

$$\sum_{i,j} P_{ij} = 4.25$$

El número total de pares posibles es $N^2 = 3^2 = 9$, por tanto, el índice JDJ normalizado se calcula como:

$$IJDJ = \frac{JDJ}{N^2} \cdot 2 = \frac{4.25}{9} \cdot 2 = 0.94$$

Este ejemplo muestra de manera clara cómo el índice JDJ captura los niveles de polarización entre nodos con diferentes grados de pertenencia, permitiendo observar tanto los extremos ideológicos como los matices intermedios.

Ejemplo 2: Nodos completamente polarizados

Supongamos un escenario extremo de polarización, donde los nodos 1 y 3 están completamente en la comunidad A, y los nodos 2 y 4 en la comunidad B. Los grados de pertenencia quedan definidos así:

Nodo	μ_A	μ_B
u_1	1	0
u_2	0	1
u_3	1	0
u_4	0	1

***Tabla 2.2**. Grados de pertenencia de los nodos*

Utilizamos los operadores de agregación y cálculo de la matriz de polarización P_{ij}:

La matriz resultante de valores de polarización es:

$$P_{ij} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Suma total de JDJ:

$$\sum_{i,j} P_{ij} = 8$$

Normalización:

$$IJDJ = \frac{JDJ}{N^2} \cdot 2 = \frac{8}{16} \cdot 2 = 1$$

Estos ejemplos ilustran cómo el índice JDJ permite cuantificar la polarización ideológica entre nodos de una red, considerando grados de pertenencia fuzzy obtenidos a partir de la estructura de vecindad. Cuando los nodos están rodeados de vecinos ideológicamente opuestos, la polarización es máxima. En cambio, cuando los nodos tienen vecindarios mixtos, el índice refleja una polarización media o baja.

2.3 Estado del arte: Polarización, redes sociales y lógica difusa

En el ámbito de la teoría de grafos y el análisis de redes, se han desarrollado diversas métricas y modelos para cuantificar la polarización, así como para entender su dinámica.

Una línea de trabajo ampliamente explorada se basa en el análisis estructural, incluyendo medidas como la **modularidad** (Newman, 2006) y el **coeficiente de homofilia** (McPherson et al., 2001). Estas métricas permiten evaluar cómo se agrupan los individuos en comunidades afines, pero no siempre capturan la intensidad o ambigüedad de las opiniones individuales.

En esta línea, Baumann et al. (2020) proponen modelos de agentes en redes que generan cámaras de eco y dinámicas de polarización a partir de reglas de homofilia y refuerzo. Aunque útiles, estos modelos se centran en la evolución dinámica, sin ofrecer métricas concretas para medir la polarización estructural a partir de datos observados.

Más recientemente, el enfoque de **lógica difusa** ha emergido como una alternativa teórica robusta para representar grados de pertenencia ideológica o de afinidad grupal. Wang y Mendel (2016) introducen la noción de *fuzzy opinion networks*, donde las opiniones no son binarias, sino representadas mediante conjuntos difusos con incertidumbre. A nivel de grafos, Guevara et al. (2024) aplican estas ideas para la detección de comunidades difusas, permitiendo que un nodo pertenezca parcialmente a más de un grupo.

En este contexto, destaca el **índice JDJ**, propuesto como herramienta flexible y formalmente sólida para medir la polarización ideológica basada en conjuntos difusos. El índice JDJ considera tanto el grado de identificación de los individuos con polos ideológicos como su alienación respecto al grupo opuesto, lo que permite capturar no solo la estructura de la red, sino también las ambigüedades y matices ideológicos individuales.

En el trabajo propuesto por Guevara et al. (2022), se muestra cómo JDJ puede identificar polarización *latente* (cuando hay posturas extremas sin agrupación estructural) y *visible* (cuando los nodos se agrupan en bloques ideológicos). Además, en dicho estudio, el índice se adapta a grafos mediante su combinación con modularidad difusa, y se aplica a redes reales y simuladas. Esta metodología ha sentado las bases para investigaciones posteriores que integran estructura de red con grados ideológicos.

El presente trabajo amplía esta línea de investigación al aplicar el índice JDJ sobre una red real de usuarios de Twitter vinculados a los modelos Openai y DeepSeek, estimando los grados de pertenencia *fuzzy* a partir del vecindario estructural de cada nodo. A diferencia de trabajos previos, este enfoque permite:

- Inferir las funciones $\mu_A(i)$ y $\mu_B(i)$ directamente desde la estructura local del grafo (sin encuestas externas).
- Aplicar JDJ de forma semanal y dinámica, observando la evolución temporal de la polarización.
- Comparar explícitamente la polarización “estructural” (modularidad) con la “ideológica difusa” (JDJ).

De este modo, el TFM ofrece una contribución metodológica original, combinando conjuntos difusos, teoría de grafos y análisis de datos reales, alineado con los trabajos de Guevara y extendiendo su aplicación a nuevos dominios.

3. DATOS Y METODOLOGÍA

3.1 Origen y naturaleza de los datos

El análisis de redes desarrollado en este trabajo se basa en datos extraídos de la red social **Twitter (X)**. Esta plataforma resulta especialmente adecuada para el estudio de dinámicas de interacción social en contextos digitales, debido a su estructura abierta y a la posibilidad de observar menciones directas entre usuarios.

El objetivo de la recolección fue identificar y analizar la red de usuarios que discuten o mencionan públicamente modelos de inteligencia artificial asociados a las compañías **OpenAI** y **DeepSeek**, durante el primer trimestre de 2025. Para delimitar este corpus temático, se utilizó una búsqueda avanzada con una consulta (query) que incluye los términos más representativos de ambas tecnologías:

"ChatGPT" OR "GPT-4" OR "GPT-4 Turbo" OR "GPT-4o" OR "GPT-o1" OR "GPT-5" OR "GPT-4 Plus" OR "OpenAI GPT" OR "OpenAI Assistant" OR "OpenAI Bot" OR "OpenAI Model" OR "DeepSeek" OR "DeepSeek AI" OR "DeepSeek LLM" OR "DeepSeek Math" OR "DeepSeek Coder" OR "DeepSeek V2" OR "DeepSeek Chat" OR "DeepSeek Pro" until:2025-04-21 since:2025-01-19

Esta consulta se aplicó directamente en el buscador de Twitter, delimitando el período comprendido entre el **19 de enero y el 20 de abril de 2025**, con el fin de acotar un intervalo temporal manejable pero representativo. Además, se optó por seleccionar únicamente los resultados categorizados como **"Top" tweets**, es decir, aquellos que han generado mayor interacción o visibilidad, descartando los tweets más recientes o aleatorios. Esta decisión es porque los tweets destacados suelen provenir de usuarios con mayor influencia, recibir más menciones, y estar mejor conectados en la red, lo que permite capturar estructuras de interacción más significativas.

Los datos recogidos se organizan en torno a tres conceptos fundamentales:

- **Tweets**: fragmentos de texto generados por los usuarios, que contienen tanto el contenido expresado como metadatos (autor, fecha de publicación, usuarios mencionados).
- **Autores**: usuarios que publican los mensajes, identificados por su nombre de cuenta. Cada autor se representa posteriormente como un nodo dentro del grafo.
- **Menciones**: interacciones directas entre usuarios, cuando uno menciona a otro mediante el símbolo @. Estas menciones constituyen las aristas de la red, al establecer una relación explícita entre dos nodos.

Cada sesión de extracción generó un archivo en formato HTML estructurado, que contiene diversas tablas con información organizada sobre los tweets recopilados y las menciones entre usuarios. Estas tablas fueron posteriormente procesadas para construir la base de datos que alimenta el análisis de redes.

3.2 Descarga y Procesamiento de archivos HTML

La recopilación de datos se llevó a cabo utilizando la herramienta **Web Data Research Assistant (WDRA)**, que permite extraer contenido público de Twitter de forma semiestructurada. Cada sesión de búsqueda en WDRA se guardó como un archivo en formato .html, que incluye varias tablas organizadas según distintos aspectos de la actividad en la red. Para este estudio, se realizaron 15 sesiones de descarga, correspondientes a distintas fechas entre el 19 de enero y el 21 de abril de 2025.

Cada archivo HTML incluye una estructura común con varias tablas. Las más relevantes para este análisis fueron:

- **Tabla 3**: contiene los **tweets** recolectados, con columnas que indican el identificador del tweet, el nombre del autor, el contenido textual, la fecha y las menciones explícitas a otros usuarios.
- **Tabla 5**: contiene los **menciones** como pares de usuario fuente y destino (quién menciona a quién), acompañadas de un campo que indica la frecuencia de aparición de ese vínculo si se repite en varios tweets.

Para procesar estos datos, se utilizó el lenguaje R con las librerías *rvest*, *stringr*, *dplyr*, y *igraph*. El flujo general del procesamiento fue el siguiente:

1. **Lectura de archivos HTML**: Se definió una lista con los nombres de los archivos descargados, y se utilizó la función read_html() para abrir cada archivo. A través de html_table(), se extrajeron las tablas relevantes y se almacenaron como data frames.

```r
# Función para extraer la tabla 3 (tweets)
extract_table3 <- function(file_path) {
  html_data <- read_html(file_path, encoding = "UTF-8")
  tables <- html_data %>% html_table(fill = TRUE)
  table3 <- as.data.frame(tables[[3]])
  return(table3)
}

# Función para extraer la tabla 5 (menciones)
extract_table5 <- function(file_path) {
  html_data <- read_html(file_path, encoding = "UTF-8")
  tables <- html_data %>% html_table(fill = TRUE)
  table5 <- as.data.frame(tables[[5]])
  return(table5)
}
```

Código 3.1. *Funciones para extraer tablas HTML desde archivos (Tweets y menciones desde código en R)*

2. **Unificación y limpieza de datos**: Las tablas extraídas de todos los archivos fueron unificadas en un solo dataset para los tweets y otro para las menciones. En el caso de los tweets, se eliminaron las entradas duplicadas utilizando el identificador único de cada mensaje (ID), de manera que cada tweet tuviera solo una vez.

```r
# ---- PROCESAR TABLA 3: TWEETS ----

# Unir datos de tweets de todos los archivos
tweets_data <- bind_rows(lapply(files, extract_table3))
# Filtrar IDs únicos (eliminando repetidos)
tweets_data <- tweets_data[!duplicated(tweets_data$ID), ]
```

Código 3.2*. Limpieza de duplicados en los tweets*

Para las menciones, los datos se agruparon por pares de usuario (Source, Target) y se sumaron las frecuencias de aparición cuando una mención se repetía en varios tweets.

```r
# ---- PROCESAR TABLA 5: MENCIONES ----

# Unir datos de menciones de todos los archivos
mentions_data <- bind_rows(lapply(files, extract_table5))
# Fusionar menciones repetidas sumando pesos
mentions_data <- mentions_data %>%
  group_by(Source, Target) %>%
  summarise(Weight = sum(Weight, na.rm = TRUE), .groups = "drop")
```

Código 3.3. *Agrupación y suma de menciones entre usuarios*

3. **Generación de menciones desde el campo Mentions de los tweets**: Aparte de la tabla 5, cada tweet también incluye un campo llamado Mentions, donde aparecen los usuarios mencionados dentro del contenido textual. Para asegurarse de capturar todas las relaciones, se analizaron manualmente estos campos, descomponiéndolos en pares fuente-destino mediante funciones de separación (str_split()) y agregándolos al conjunto general de menciones.

```r
# Crear un bucle que vaya de 1 a nrow(tweets_data)
# si menciones está vacío, next, sino:
# m <- unlist(str_split(tweets_data[tweets_data$Author=="fsorel","Mentions"], " "))

mentions_data <- data.frame(SOURCE = 0, TARGET = 0)
for (i in 1:nrow(tweets_data)) {
  if (tweets_data$Mentions[i] == "") {
    next
  } else {
    m <- unlist(str_split(tweets_data$Mentions[i], " "))
    for (j in 1:length(m)) {
      mentions_data <- rbind(mentions_data, data.frame(SOURCE = tweets_data$Author[i], TARGET = m[j]))
    }
  }
}
```

Código 3.4. *Generación manual de menciones desde el campo Mentions*

4. **Filtrado de menciones inválidas:** Para garantizar la validez de los datos, se eliminaron todas las relaciones en las que alguno de los extremos (SOURCE o TARGET) estuviera vacío, fuera igual a "0" o no correspondiera a un usuario real.

```r
# Eliminar conexiones donde SOURCE o TARGET son vacíos o "0"
mentions_data <- mentions_data %>%
  filter(SOURCE != "" & TARGET != "") %>%
  filter(SOURCE != "0" & TARGET != "0")
```

Código 3.5*. Filtrado de menciones inválidas en los datos procesados*

Como resultado de este proceso, se obtuvo un conjunto de datos limpio, estructurado y consolidado, listo para ser transformado en una red de interacciones.

Esta estrategia de procesamiento ofrece una visión más completa y robusta de cómo se estructura la conversación digital en torno a los modelos de inteligencia artificial OpenAI y DeepSeek dentro de la plataforma. Al incluir tanto las menciones capturadas por WDRA como aquellas identificadas manualmente en el texto, se mejora la fidelidad del grafo y se facilita el análisis posterior de comunidades, centralidad y polarización.

3.3 Construcción del grafo

A partir del conjunto de datos procesado en la sección anterior, se construyó un grafo no dirigido que representa las relaciones de mención entre usuarios. En este grafo,

cada nodo corresponde a un usuario de Twitter y cada arista representa al menos una mención explícita entre dos cuentas durante el periodo analizado.

1. **Modelo de red no dirigida:** Aunque las menciones en Twitter son interacciones dirigidas (un usuario menciona a otro), se optó por construir un grafo no dirigido con el objetivo de capturar la estructura global de interacción sin considerar el sentido de cada vínculo. Esta decisión responde a una motivación analítica, al centrarse en los patrones de agrupamiento y cohesión, la dirección específica de la mención resulta secundaria frente a la existencia del contacto.
   La red fue creada en R utilizando la función graph_from_data_frame() de la librería *igraph*, a partir del conjunto consolidado de menciones:

```r
# Crear el grafo con todas las menciones
mention_graph <- graph_from_data_frame(mentions_data, directed = FALSE)
```

Código 3.6. *Creación del grafo a partir del conjunto de menciones con igraph*

2. **Visualización inicial y detección de comunidades:** Como primer paso de exploración, se realizó una detección de comunidades naturales utilizando el algoritmo Louvain, que permite identificar agrupaciones densamente conectadas sin necesidad de definir el número de grupos a priori:

```r
# Detectar comunidades con Louvain
communities <- cluster_louvain(mention_graph)
```

Código 3.7. *Detección de comunidades mediante el algoritmo Louvain*

   A continuación, se generaron los data frames necesarios para una visualización interactiva de la red mediante *visNetwork*, asignando a cada nodo su comunidad detectada y su grado de conexión (número de enlaces).

```r
# Crear el dataframe de nodos con la información de comunidad
nodes <- data.frame(
  id = V(mention_graph)$name,   # ID del nodo (nombre del usuario)
  label = V(mention_graph)$name,  # Etiqueta del nodo (para visualización)
  group = membership(communities),  # Número de comunidad a la que pertenece el nodo
  value = degree(mention_graph)  # Tamaño del nodo basado en el grado (cantidad de conexiones)
)
# Crear el dataframe de aristas
edges <- data.frame(
  from = as.character(mentions_data$SOURCE),  # Usuario que menciona
  to = as.character(mentions_data$TARGET)  # Usuario mencionado
  #value = mentions_data$Weight  # Peso de la relación (frecuencia de menciones)
)
```

Código 3.8*. Generación de dataframes de nodos y aristas para visualización con visNetwork*

   La visualización resultante permite explorar dinámicamente la estructura de la red y las agrupaciones emergentes en torno a los distintos focos de conversación:

```r
# Crear el grafo interactivo con visNetwork
visNetwork(nodes, edges) %>%
  visOptions(highlightNearest = TRUE, nodesIdSelection = TRUE) %>%
  visPhysics(stabilization = FALSE) %>%
  visInteraction(dragNodes = TRUE, zoomView = TRUE) %>%
  visLayout(randomSeed = 42)
```

Código 3.9*. Visualización interactiva del grafo mediante la librería visNetwork*

3. **Estadísticas básicas del grafo:** Antes de pasar al análisis estructural detallado, se calcularon algunas métricas básicas del grafo, y para garantizar la consistencia entre la lista de nodos y las menciones efectivamente presentes en

los datos, se compararon los nodos del grafo con los usuarios únicos identificados en el conjunto mentions_data. Esta validación asegura que la red refleja con fidelidad las relaciones de mención reales y que no hay nodos desconectados introducidos por error.

```
> cat("Total de nodos en el grafo:", vcount(mention_graph), "\n")
Total de nodos en el grafo: 2827
> unique_nodes <- unique(c(mentions_data$SOURCE, mentions_data$TARGET))
> cat("Total de nodos únicos en mentions_data:", length(unique_nodes), "\n")
Total de nodos únicos en mentions_data: 2827
```

Figura 3.1. *Salida de consola que muestra la consistencia entre el número de nodos únicos en el grafo y en el conjunto mentions_data.*

Con esta estructura de grafo, se establece la base para el análisis posterior. La red de menciones reconstruida constituye una representación robusta de las interacciones entre usuarios durante el periodo de estudio.

4. ANÁLISIS DE LA RED GLOBAL

4.1 Estructura general de la red

La red global de menciones resultante está compuesta por 2.827 nodos (usuarios únicos) y 3.711 aristas (menciones entre ellos). Esta red representa todas las interacciones registradas durante el período de análisis, sin aplicar aún filtros de polarización o partición forzada.

Pese a su densidad relativamente baja (0.00029), la red exhibe una estructura compleja con 615 componentes conexas, lo cual indica la existencia de múltiples "satélites" o subgrupos desconectados entre sí. La componente gigante concentra a 1.224 usuarios, es decir, alrededor del 43% de la red, y es donde se concentra la actividad principal.

Las métricas básicas permiten describir mejor su estructura:

```
> # Métricas básicas de la red global
> cat("Número total de nodos:", vcount(mention_graph), "\n")
Número total de nodos: 2827
> cat("Número total de aristas:", ecount(mention_graph), "\n")
Número total de aristas: 3711
> cat("Número de componentes conexas:", components(mention_graph)$no, "\n")
Número de componentes conexas: 615
> cat("Tamaño de la componente gigante:", max(components(mention_graph)$csize), "\n")
Tamaño de la componente gigante: 1224
> cat("Densidad de la red:", graph.density(mention_graph), "\n")
Densidad de la red: 0.0009290156
> cat("Diámetro de la red:", diameter(mention_graph, unconnected = TRUE), "\n")
Diámetro de la red: 15
> cat("Distancia media:", mean_distance(mention_graph, directed = FALSE), "\n")
Distancia media: 5.190338
> cat("Clustering global:", transitivity(mention_graph, type = "global"), "\n")
Clustering global: 0.003947099
```

Figura 4.1. *Resultados de las métricas estructurales básicas calculadas sobre el grafo global*

4.2 Detección de comunidades naturales

Con el fin de identificar agrupaciones dentro del grafo de menciones, se aplicó el algoritmo de Louvain, ampliamente utilizado en análisis de redes por su capacidad para detectar comunidades densamente conectadas de forma eficiente. Este método no

requiere fijar a priori el número de comunidades y se basa en la optimización de la modularidad.

```
> # Número de comunidades detectadas con Louvain
> communities <- cluster_louvain(mention_graph)
> cat("Número de comunidades detectadas (Louvain):", length(communities), "\n")
Número de comunidades detectadas (Louvain): 642
```

Figura 4.2. *Número de comunidades detectadas utilizando el algoritmo de Louvain*

El resultado fue la detección de 642 comunidades distintas, lo que revela un patrón altamente fragmentado en la conversación sobre inteligencia artificial. Pueden que sea muchas de estas comunidades son pequeñas o incluso de un solo nodo (componentes satélites). Este elevado número de comunidades sugiere una estructura policéntrica, en la que múltiples grupos discuten sobre distintos aspectos del tema, sin un único núcleo dominante. Además la diversidad temática y la dispersión de usuarios influyentes por distintas comunidades refuerzan la idea de una conversación distribuida, con interacciones en subgrupos especializados, como divulgadores, perfiles técnicos, instituciones o medios.

4.3 Métricas de centralidad

Para identificar a los usuarios más influyentes en la red, se calcularon cuatro métricas clásicas de centralidad:

- **Grado (Degree)** es el número de conexiones directas de un nodo.
- **Intermediación (Betweenness)** es la frecuencia con la que un nodo aparece en los caminos más cortos entre otros.
- **Cercanía (Closeness)** que evalúa la facilidad con la que un nodo puede alcanzar a los demás en la red.
- **Autovector (Eigenvector)** es la medida de prestigio, que valora más los vínculos con nodos influyentes.

Los resultados se resumen en la siguiente tabla con los 5 usuarios más destacados:

```
> # Mostrar los top 5 usuarios en cada métrica de centralidad
> cat("Top 5 usuarios por Grado (Degree Centrality):")
Top 5 usuarios por Grado (Degree Centrality):
> print(top_degree)
deepseek_ai betamoroney      tut_ml      openai   giga_labs
        309         296         234         186         147
> cat("Top 5 usuarios por Intermediación (Betweenness Centrality):")
Top 5 usuarios por Intermediación (Betweenness Centrality):
> print(top_betweenness)
    deepseek_ai          openai aureliecoudouel     betamoroney  anthonyrochand
     0.10565796      0.07075961      0.07021335      0.02229217      0.02077456
> cat("Top 5 usuarios por Cercanía (Closeness Centrality):")
Top 5 usuarios por Cercanía (Closeness Centrality):
> print(top_closeness)
   aralleida riffreporter  stanfordhai       igakuj   rubixchain
           1           1           1           1           1
> cat("Top 5 usuarios por Autovector (Eigenvector Centrality):")
Top 5 usuarios por Autovector (Eigenvector Centrality):
> print(top_eigen)
       tut_ml      antgrasso ronald_vanloon     kirkdborne      giga_labs
   1.00000000     0.58213340     0.57996741     0.57918759     0.08131342
```

Figura 4.3. *Principales usuarios según métricas de centralidad (grado, intermediación, cercanía, autovector)*

Estas métricas muestran distintos tipos de influencia. Por ejemplo, deepseek_ai aparece como nodo central en varias medidas, mientras que tut_ml destaca especialmente por su peso en la red según el autovector. Además, openai también cumple un rol importante como puente entre grupos.

4.4 Visualización interactiva de la red global

Para explorar la estructura de la red de menciones, se generó una visualización interactiva utilizando la librería *visNetwork*. Los nodos han sido agrupados por comunidad (según el algoritmo Louvain) y coloreados en función de esta pertenencia.

La visualización revela una estructura altamente fragmentada, en la que destaca una componente central densa rodeada de múltiples satélites dispersos. Estos satélites corresponden a pequeños grupos de usuarios que no se conectan con el núcleo principal de la conversación.

A pesar de haber resaltado manualmente a los usuarios más influyentes (como openai, deepseek_ai o tut_ml), la densidad y el tamaño de la red impiden que sus etiquetas sean difíciles de visualizar en el grafo completo. Esto es normal en redes con alto número de nodos y bajo clustering, como la que se analiza. Para hacer visibles estas dinámicas, se propone en la siguiente sección trabajar con una subred más enfocada, centrada en la componente gigante.

Más allá de su valor visual, esta representación resulta útil como herramienta exploratoria, ya que permite detectar patrones generales de agrupamiento, comunidades destacadas, e incluso anomalías estructurales.

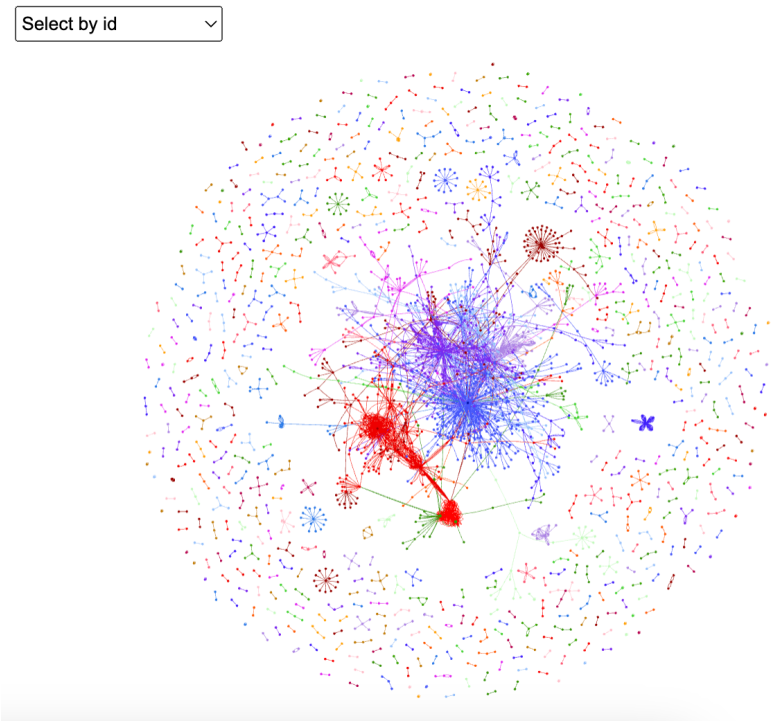


Figura 4.4. *Visualización completa del grafo de menciones con comunidades detectadas*

5. COMUNIDADES FORZADAS Y POLARIZACIÓN

Para garantizar la coherencia estructural del análisis y evitar ruido procedente de nodos periféricos, se restringe la red a su componente conexa gigante. Esta subred agrupa al 43 % de los nodos y concentra la mayor parte de las interacciones registradas, lo que la convierte en el núcleo del debate en torno a los modelos OpenAI y DeepSeek. Al trabajar sobre esta estructura más densa y conectada, se pueden aplicar algoritmos de detección comunitaria con mayor precisión y robustez.

5.1 Aplicación del algoritmo Fast-Greedy

Una vez aislada la componente gigante, se aplicó el algoritmo **Fast-Greedy** para detectar comunidades estructurales en la red de menciones. Este método aglomerativo construye jerárquicamente agrupaciones de nodos que maximizan la **modularidad**, una métrica que mide la calidad de la partición en términos de densidad interna de conexiones frente a las externas.

Se compararon dos enfoques:

```
> modularity(mention_graph_gc,forced_membership) # modularidad de la partición restringida a 2 comunidades
[1] 0.4348189
> modularity(mention_graph_gc,fg_comm$membership) # modularidad de la partición que Fast-Greedy encontró sin restricciones
[1] 0.7561309
```

Figura 5.1. *Comparación de modularidad entre partición libre y forzada con el algoritmo Fast-Greedy*

- **Partición libre** (no forzada): el algoritmo detectó de forma automática varias comunidades, alcanzando una modularidad de **0.756**, lo cual indica una estructura interna fuertemente modular.
- **Partición forzada a dos comunidades**: se impuso una división binaria sobre la red, obteniendo una modularidad de **0.435**, significativamente menor.

En otras palabras, el método Fast-Greedy, sin restricciones, decidió que había más de dos grupos (quizá cuatro, cinco o más), y esa segmentación “óptima” reflejaba mejor la estructura real de la red. Al forzar exactamente dos comunidades, se pierde parte de esa fidelidad estructural, lo que se refleja en la caída de la modularidad (**0.756 → 0.435**).

No obstante, es importante señalar que, aunque la modularidad disminuye, la partición forzada puede ser conceptualmente útil si el objetivo es estudiar **polarización entre dos bloques**. A la hora de calcular JDJ, se utiliza el algoritmo *makefuzzy*, por lo que se entiende que un nodo puede pertenecer a ambas comunidades a la vez en cierto grado. En este contexto, una partición crispy (nítida) con baja modularidad no necesariamente implica una segmentación incorrecta, sino una simplificación deliberada con fines analíticos. Cabe señalar que existen versiones de modularidad difusa, más adecuadas para estos escenarios.

En resumen, una modularidad alta indica que los grupos encontrados reflejan fielmente la estructura conectiva de la red (muchas conexiones dentro de cada grupo,

pocas entre ellos). Al forzar solo dos grupos, estamos imponiendo una partición menos alineada con esta estructura natural, por lo que la modularidad disminuye.

5.2 Análisis de centralidad

Tras forzar la red a una partición binaria (dos comunidades), se aplica un análisis de centralidad para identificar a los usuarios más relevantes dentro de cada grupo. Se han utilizado cuatro métricas clásicas de análisis de redes: **grado**, **cercanía**, **intermediación** y **autovector**. Estas métricas permiten evaluar tanto la visibilidad directa de un nodo (grado), como su capacidad de alcance (cercanía), de conexión entre otros (intermediación), o su importancia en función del prestigio de sus vecinos (autovector).

La siguiente tabla recoge el top 5 de usuarios por cada métrica en ambas comunidades. Esta información resulta clave para interpretar el perfil de cada grupo:

Comunidad 1

Métrica	Usuario 1	Valor	Usuario 2	Valor	Usuario 3	Valor	Usuario 4	Valor	Usuario 5	Valor
Grado	openai	130	betamoroney	46	elonmusk	45	sama	43	youtube	43
Cercanía	openai	0.2932	huzefaaziz4	0.2622	ghcest1	0.2599	nbdpress	0.2587	michaelcree7	0.2584
Intermediación	openai	0.6329	sama	0.2126	ronald_vanloon	0.2092	elonmusk	0.1938	giga_labs	0.1573
Autovector	openai	1.0000	sama	0.2178	ghcest1	0.1201	huzefaaziz4	0.1182	nbdpress	0.1155

Comunidad 2

Métrica	Usuario 1	Valor	Usuario 2	Valor	Usuario 3	Valor	Usuario 4	Valor	Usuario 5	Valor
Grado	deepseek_ai	223	chatgptapp	51	aureliecoudouel	31	grok	22	jeffsheehan	13
Cercanía	deepseek_ai	0.4387	aureliecoudouel	0.3427	forleoliang	0.3287	mtechtian	0.3271	rahulsutariya	0.3267
Intermediación	deepseek_ai	0.8977	aureliecoudouel	0.2862	chatgptapp	0.1380	numerama	0.0838	techopedia	0.0693
Autovector	deepseek_ai	1.0000	chatgptapp	0.1714	mtechtian	0.0821	forleoliang	0.0814	rahulsutariya	0.0812

Tabla 5.1. *Top 5 de usuarios por métrica de centralidad en la Comunidad 1 y Comunidad 2*

En la **comunidad 1 (nodos rojos)**, destacan de forma consistente los usuarios openai, sama, elonmusk, nbdpress, y giga_labs, todos ellos estrechamente vinculados al ecosistema OpenAI. En todas las métricas, openai aparece como nodo más central, lo cual indica que esta comunidad se articula alrededor de dicha organización. Por este motivo, denominamos esta **comunidad** como el bloque **OpenAI**.

En la **comunidad 2 (nodos azules)**, se observa una estructura liderada por deepseek_ai, acompañado por otros actores como deepseek, aureliecoudouel, chatgptapp, y mtechitian, que ocupan posiciones destacadas en múltiples métricas. En este caso, la concentración de centralidad en torno a deepseek_ai permite identificar este grupo como la **comunidad DeepSeek**.

Junto a la tabla, se presenta una gráfica comparativa que permite visualizar de forma inmediata la distribución de centralidad entre comunidades. Para cada métrica se muestran los cinco usuarios con mayor valor, diferenciando en color los pertenecientes

a la comunidad 1 (rojo) y a la comunidad 2 (azul). Este enfoque visual permite apreciar no solo quién domina cada métrica, sino también las diferencias internas entre comunidades.

Por ejemplo, en **grado**, deepseek_ai y openai se destacan claramente, aunque la comunidad azul muestra mayor dispersión. En cuanto a la **intermediación**, openai y sama lideran en la comunidad roja, mientras que deepseek_ai y aureliecoudouel lo hacen en la azul, son nodos que sirven de puente. Y en **autovector**, se refuerza la posición central de openai y deepseek_ai, indicando que están conectados con otros nodos también influyentes.

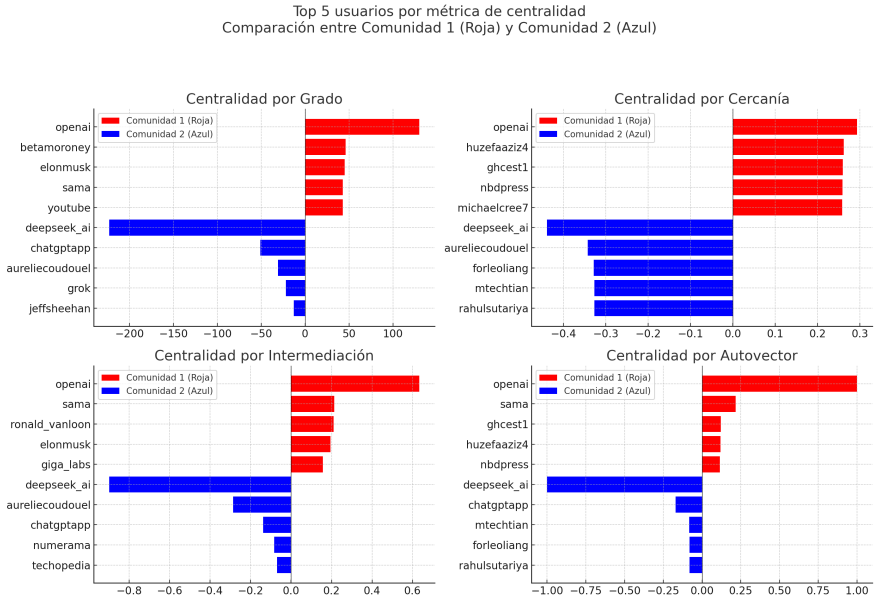


Figura 5.2. *Comparación visual de centralidades por comunidad (Gráfica comparativa en Python: grado, cercanía, intermediación, autovector)*

La tabla y los gráficos no solo refuerzan la asignación temática de las comunidades, sino que revelan estructuras distintas, ya que la comunidad roja aparece más centralizada y jerárquica, mientras que la azul presenta un liderazgo más distribuido entre varios actores.

Estas diferencias ayudarán a contextualizar mejor el grafo visual que se presenta en el siguiente apartado, en el que se destacan los nombres de los principales nodos para facilitar la lectura e interpretación.

5.3 Visualización del grafo con partición forzada

Para complementar el análisis de centralidad, se presenta el grafo correspondiente a la componente gigante de la red de menciones, forzado a una partición binaria (dos comunidades). Los nodos han sido coloreados según la comunidad asignada por el algoritmo Fast-Greedy: **rojo** para la comunidad 1 (identificada con el ecosistema **OpenAI**) y **azul** para la comunidad 2 (asociada a **DeepSeek**).

En la visualización se han destacado con etiquetas los usuarios más centrales según las métricas analizadas previamente. Esta representación permite observar de forma directa la estructura de la polarización, donde se aprecian claramente dos núcleos densos con interacciones internas predominantes y una zona intermedia donde se sitúan nodos conectores entre ambas comunidades.

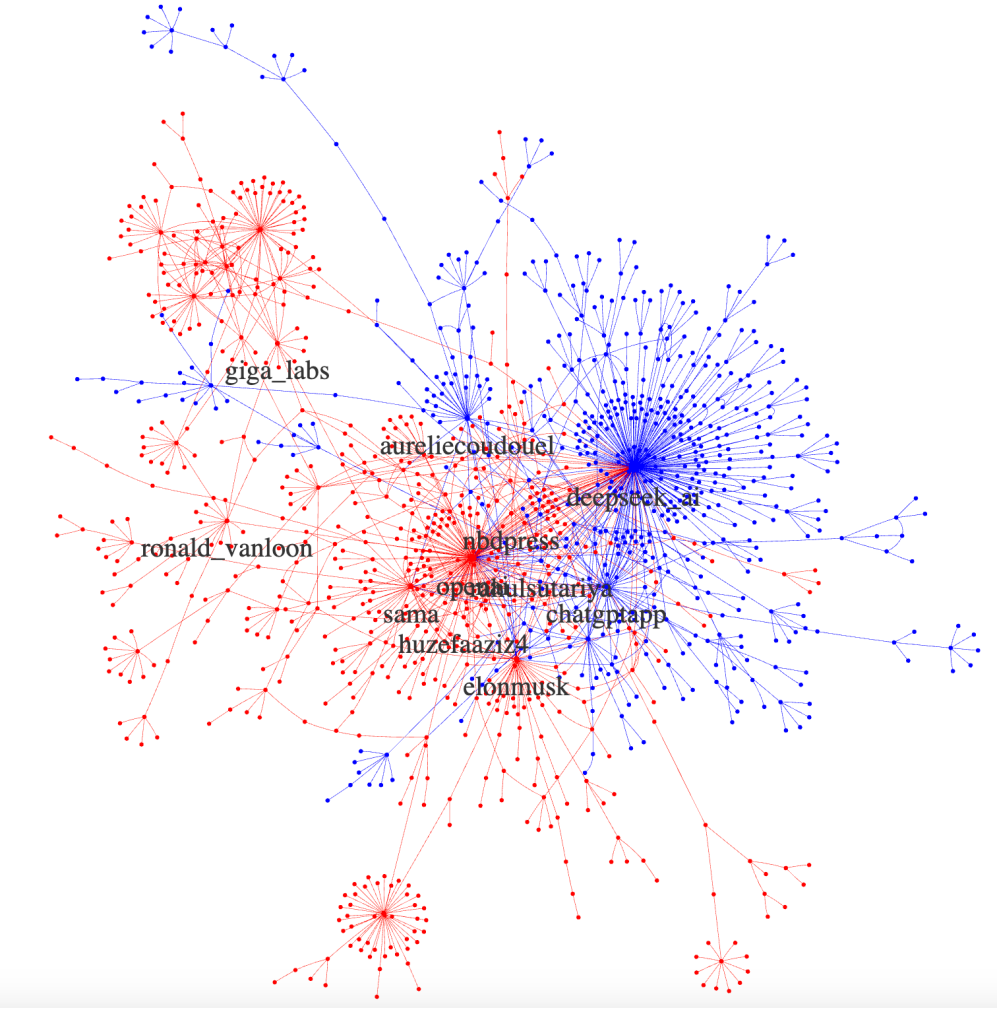


Figura 5.3. *Grafo del componente gigante coloreado por comunidad (Roja – OpenAI, Azul – DeepSeek)*

En definitiva, el grafo refuerza la interpretación de que, aunque la partición a dos comunidades no maximiza la modularidad, sí permite capturar una división significativa entre los principales actores del debate.

5.4 Análisis de hashtags diferenciadores

Para comprender los temas predominantes y enfoques discursivos en cada comunidad, se analizaron los hashtags más frecuentes tras aplicar la partición forzada a dos comunidades (OpenAI en rojo y DeepSeek en azul). En la siguiente tabla se presentan los 20 hashtags más utilizados por cada grupo.

Comunidad 1 (Roja – OpenAI)	Frecuencia	Comunidad 2 (Azul – DeepSeek)	Frecuencia
#chatgpt	396	#deepseek	292
#ai	309	#chatgpt	289
#deepseek	248	#ai	209
#openai	228	#openai	113
#deeplearning	160	#artificialintelligence	48
#genai	111	#deepseekai	45
#machinelearning	94	#ia	42
#ml	94	#technology	26
#datascience	84	#china	26
#bigdata	82	#grok	22
#artificialinteligence	79	#samaltman	20
#healthtech	79	#google	19
#datavisualization	78	#jeffsaipost	18
#softwareengineering	78	#meta	17
#python	77	#tech	16
#generativeai	63	#deepseekr1	38
#artificialintelligence	61	#domains	14
#keras	49	#chatgpt, (con coma)	15
#tech	39	#deepseek's	16
#data	37	#gemini	14

Tabla 5.2. *Top 20 hashtags más utilizados por comunidad*

En la **Comunidad 1 (Roja – OpenAI)**, los hashtags dominantes están fuertemente ligados al ámbito técnico y académico de la inteligencia artificial. Sobresalen etiquetas como **#chatgpt**, **#ai**, **#openai**, **#deeplearning**, **#genai**, y **#machinelearning**, que sugieren a tecnologías generativas y procesos de aprendizaje automático. También aparecen referencias frecuentes a **#python**, **#datascience**, **#bigdata**, y **#keras**, lo que refuerza una orientación hacia perfiles desarrolladores, científicos de datos y comunidades tecnológicas especializadas.

En cambio, la **Comunidad 2 (Azul – DeepSeek)** mantiene algunas etiquetas comunes (como **#chatgpt**, **#ai**, **#openai**), pero introduce elementos distintivos como **#deepseek**, **#deepseekai**, **#deepseekr1**, y otras etiquetas más periféricas como **#china**, **#technology**, **#jeffsaipost**, o **#domains**. Esto sugiere un bloque de usuarios posiblemente vinculado a iniciativas emergentes, presencia institucional o campañas de posicionamiento.

En conjunto, este análisis sugiere que, si bien ambos grupos giran en torno al debate sobre IA generativa, cada comunidad expresa intereses, públicos y estrategias comunicativas distintas.

Además de la tabla, se generaron nubes de palabras para ambas comunidades, que permiten visualizar de forma intuitiva la prominencia relativa de los términos más repetidos. En la comunidad **OpenAI (roja)**, el foco se centra en tecnologías de IA, modelos de lenguaje y herramientas de desarrollo. En cambio, en la comunidad **DeepSeek (azul)**, los hashtags reflejan un esfuerzo de posicionamiento de marca y ecosistemas emergentes, con una menor diversidad temática.

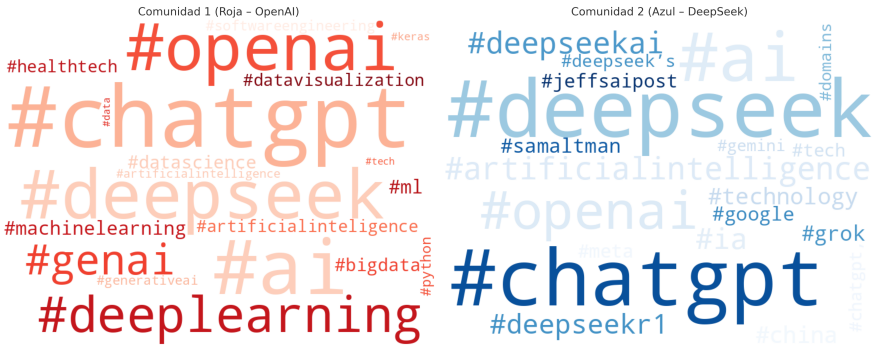


Figura 5.4. *Nube de palabras por comunidad (Wordcloud generada en Python con hashtags por grupo)*

Estas visualizaciones y conteos refuerzan la hipótesis de polarización, cada comunidad no solo presenta distinta estructura de red, sino también un lenguaje distintivo que las identifica semánticamente.

5.5 Análisis de Idiomas por Comunidad

Para comprender mejor la dimensión lingüística del debate, se analizó el idioma de los tuits presentes en la componente gigante de la red. Dado que los textos en redes sociales suelen incluir ruido como enlaces, menciones, hashtags o signos de puntuación, se realizó primero una limpieza exhaustiva del contenido textual para facilitar la detección del idioma. Esta limpieza incluyó la eliminación de URLs, menciones, hashtags, signos de puntuación, palabras sueltas y caracteres innecesarios.

Una vez depurado el texto, se aplicó el paquete *cld2* para identificar el idioma principal de cada tuit. Para los casos en los que *cld2* no logró una clasificación (quedando como NA), se utilizó una estrategia alternativa con la librería *textcat*, la cual logró cubrir parte de los vacíos.

Como resultado, se obtuvieron distribuciones lingüísticas bastante contrastadas entre las dos comunidades detectadas:

- **La Comunidad 1 (Roja – asociada a OpenAI)** presenta una clara predominancia del inglés, con una proporción muy alta de tuits en ese idioma. Además, aunque en menor medida, se encuentran otros idiomas como el español y el alemán.
- **La Comunidad 2 (Azul – asociada a DeepSeek)** también se expresa principalmente en inglés, pero destaca por una mayor diversidad lingüística, especialmente con una presencia significativa del francés.

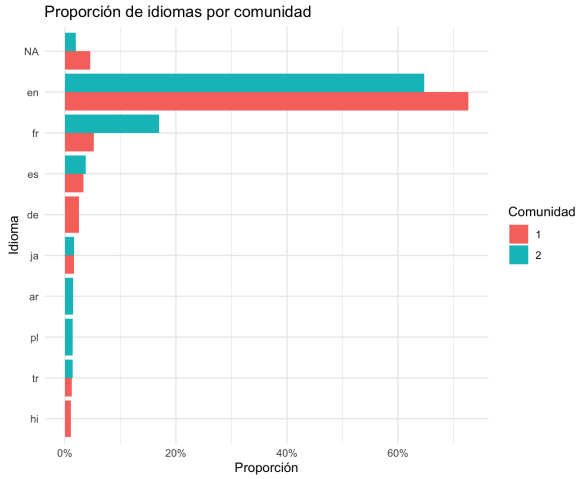


Figura 5.5. *Proporción de idiomas por comunidad*

Esto se visualiza claramente en el gráfico de barras, donde se representan los idiomas con más del 1% de proporción por comunidad. En particular, el francés representa un **17.3%** de los tuits en la Comunidad 2, frente a solo un **2.7%** en la Comunidad 1, lo que sugiere una posible comunidad francófona más activa alrededor de DeepSeek.

Cabe mencionar que, a pesar de observar la presencia de caracteres chinos en algunos tuits, no se logró identificar correctamente el idioma chino en muchas ocasiones. Esto podría deberse a la escasa cantidad de texto útil en esos tuits o a las limitaciones de los detectores automáticos, lo que se refleja en la proporción de casos marcados como NA.

En resumen, aunque ambas comunidades se comunican predominantemente en inglés, la Comunidad 2 muestra una mayor diversidad idiomática, con una base francófona destacada. Esta información aporta contexto sobre los públicos que interactúan en torno a cada polo del debate y puede vincularse a diferencias geográficas o culturales.

5.6 Análisis descriptivo por comunidad

Además de las diferencias estructurales y de idioma, se analizaron algunas variables que ayudan a perfilar mejor el comportamiento de las dos comunidades detectadas: la comunidad **1 (Roja – OpenAI)** y la comunidad **2 (Azul – DeepSeek)**.

Indicador	Comunidad 1 (Roja – OpenAI)	Comunidad 2 (Azul – DeepSeek)
Nº de usuarios (nodos)	661	563
Promedio de conexiones por usuario	2.73	2.58
Media de tweets por autor	2.05	2.12
Cuentas con 'news|daily|press'	25	28

Cuentas con 'ai|tech|robot|ml|data'	171	90
Hashtags únicos	#deeplearning, #ml, #datavisualization, #softwareengineering, #keras, #digitaltransformation	#deepseekai, #domainsforsale, #jeffsaipost, #llms, #stargateproject, #seo

Tabla 5.3. *Indicadores descriptivos por comunidad*

Hashtags exclusivos

Al comparar los 20 hashtags más frecuentes en ambas comunidades, se identificaron términos que aparecen exclusivamente en una de ellas, revelando intereses o enfoques particulares:

- **Hashtags únicos en la Comunidad 1 (Roja – OpenAI)**: destacan temas como #deeplearning, #ml, #datavisualization, #softwareengineering, #keras o #digitaltransformation, todos asociados a aspectos técnicos y científicos del desarrollo de IA.
- **Hashtags únicos en la Comunidad 2 (Azul – DeepSeek)**: aparecen etiquetas como #deepseekai, #domainsforsale, #jeffsaipost, #llms, #stargateproject, o incluso términos de marketing como #seo, lo que sugiere una comunidad más heterogénea, con foco tanto en IA como en promoción, mercado de dominios o divulgación.

Tamaño y conectividad

Ambas comunidades son de tamaño similar, la comunidad roja contiene 661 nodos (usuarios), mientras que la azul agrupa a 563. Sin embargo, el promedio de conexiones (grado) es superior en la comunidad roja (2.73 frente a 2.57), lo que sugiere una red más densa en ese grupo.

Actividad por autor

El promedio de tweets por autor también revela diferencias leves, los usuarios de la comunidad azul publican más mensajes en promedio (2.12 tweets por usuario) que los de la comunidad roja (2.05). Esto puede reflejar un grado mayor de participación individual en la comunidad DeepSeek.

Presencia institucional vs. automatizada

Para explorar el tipo de actores presentes en cada comunidad, se analizaron los nombres de usuarios que incluyen ciertos patrones:

- Cuentas con referencias a medios de comunicación (“news, daily, press”): ambas comunidades muestran una presencia equilibrada de este tipo de usuarios, con 25 en la comunidad 1 y 28 en la comunidad 2.
- Cuentas con referencias a IA, tecnología o automatización ("ai, tech, robot, ml, data"): este tipo de cuentas están mucho más presentes en ambas comunidades,

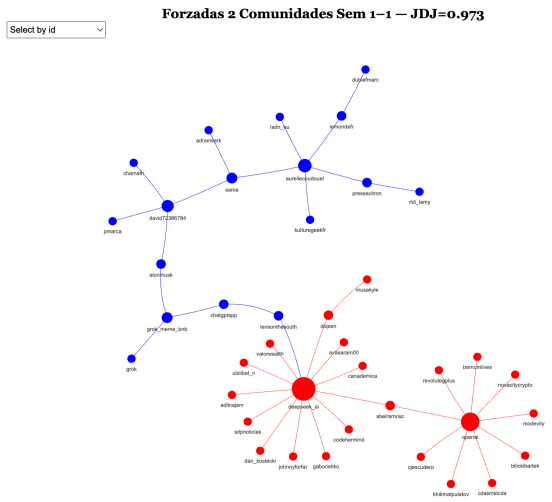
con un ligero predominio en la comunidad roja (171 frente a 90). Esto puede estar asociado a bots, cuentas automatizadas o usuarios centrados específicamente en contenidos técnicos sobre IA.

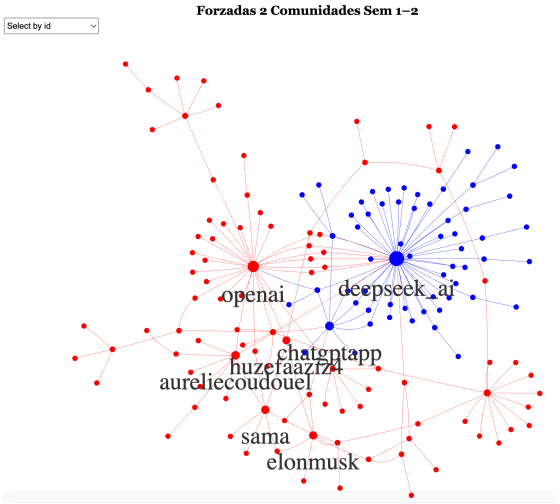
6. MEDICIÓN DE LA POLARIZACIÓN Y EVOLUCIÓN DE LA RED

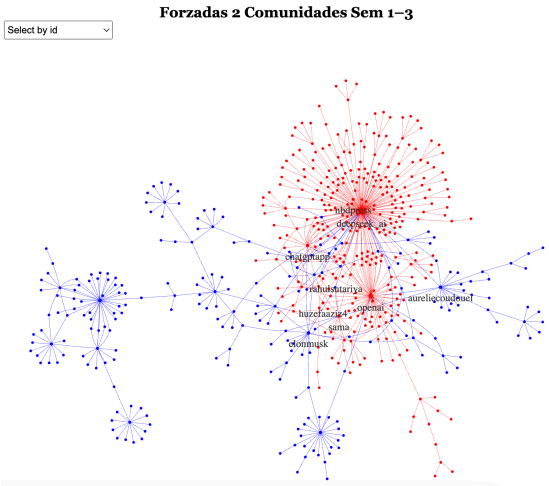
6.1 Visualización de grafos semanales

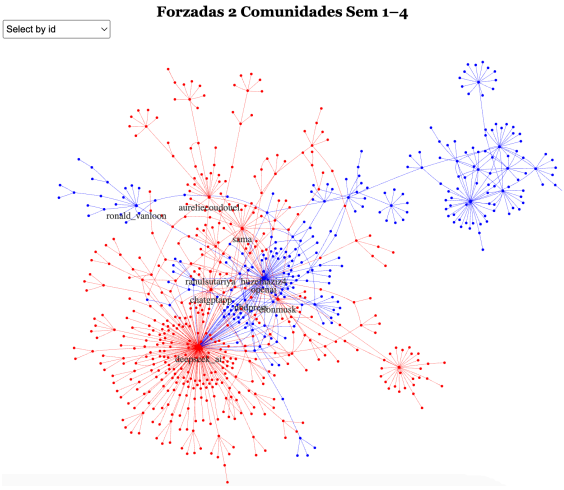
Con el objetivo de analizar la evolución de la red y su posible polarización ideológica, se visualizan los grafos correspondientes a cada semana del periodo analizado. Estos grafos se construyen a partir de las interacciones en Twitter entre usuarios que mencionan a modelos de lenguaje (como GPT y DeepSeek) y se representan como redes no dirigidas. En cada caso, se ha forzado una partición en dos comunidades, asignadas visualmente en rojo y azul, para facilitar la lectura comparativa entre semanas.

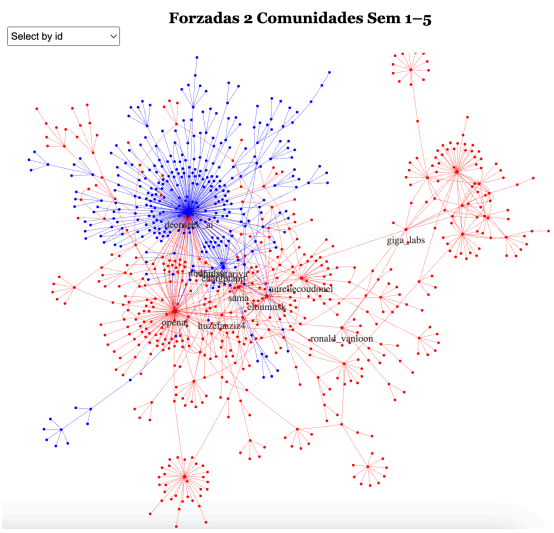
Estas visualizaciones permiten observar tanto el crecimiento estructural de la red (número de nodos y aristas), como la evolución de las comunidades, su grado de cohesión interna y la progresiva aparición de estructuras polarizadas.

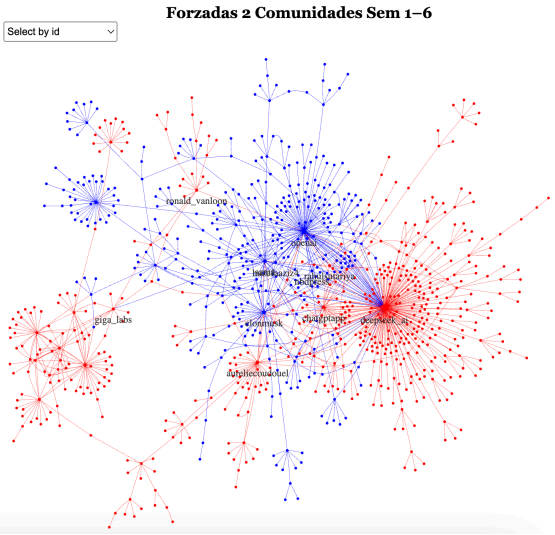


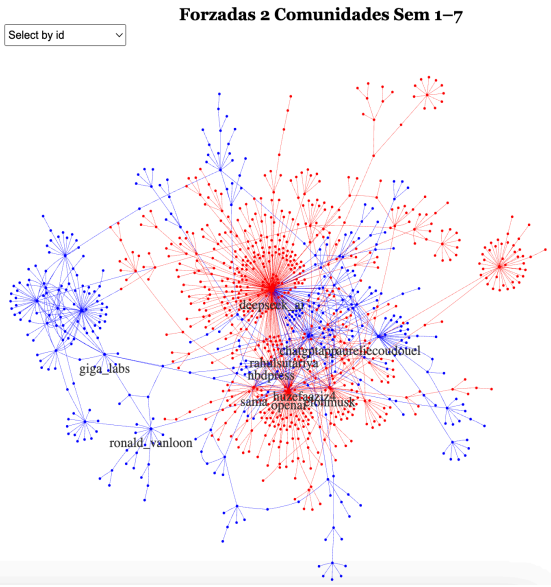


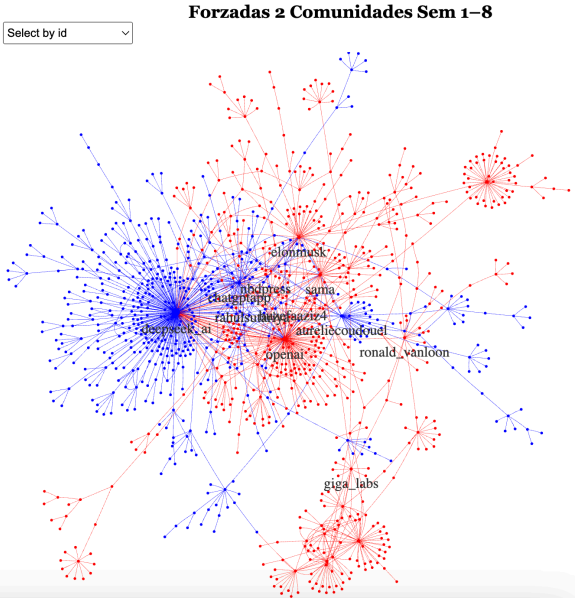


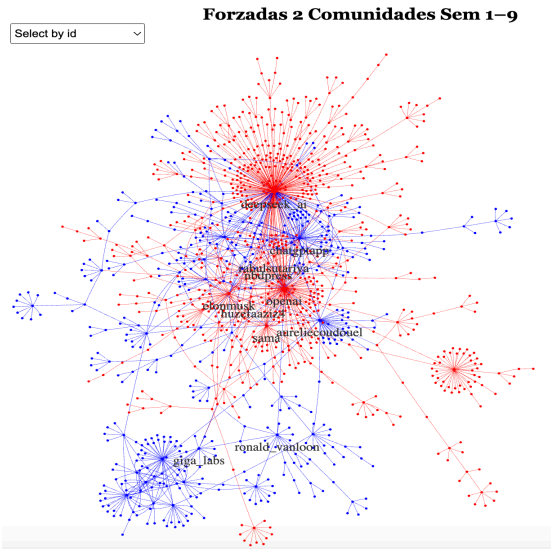


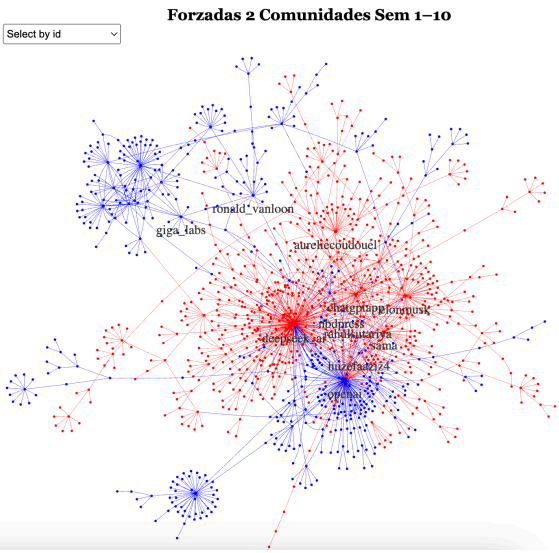

Forzadas 2 Comunidades Sem 1–5
Select by id
giga_labs
aurelieccoudouel
sama
elonmusk
openai
huzefaaziz4
ronald_vanloon


Forzadas 2 Comunidades Sem 1–6
Select by id
ronald_vanloon
openai
giga_labs
elonmusk
deepseek_ai
aurelieccoudouel


Forzadas 2 Comunidades Sem 1–7
Select by id
deepseek_ai
giga_labs
rahulsutariya
hbdpress
sama
openai
elonmusk
huzefaaziz4
aurelieccoudouel
ronald_vanloon


Forzadas 2 Comunidades Sem 1–8
Select by id
elonmusk
hbdpress
sama
deepseek_ai
aurelieccoudouel
openai
ronald_vanloon
giga_labs


Forzadas 2 Comunidades Sem 1–9
Select by id
deepseek_ai
hbdpress
elonmusk
openai
huzefaaziz4
sama
aurelieccoudouel
ronald_vanloon
giga_labs


Forzadas 2 Comunidades Sem 1–10
Select by id
ronald_vanloon
giga_labs
aurelieccoudouel
elonmusk
hbdpress
deepseek_ai
rahulsutariya
sama
huzefaaziz4
openai

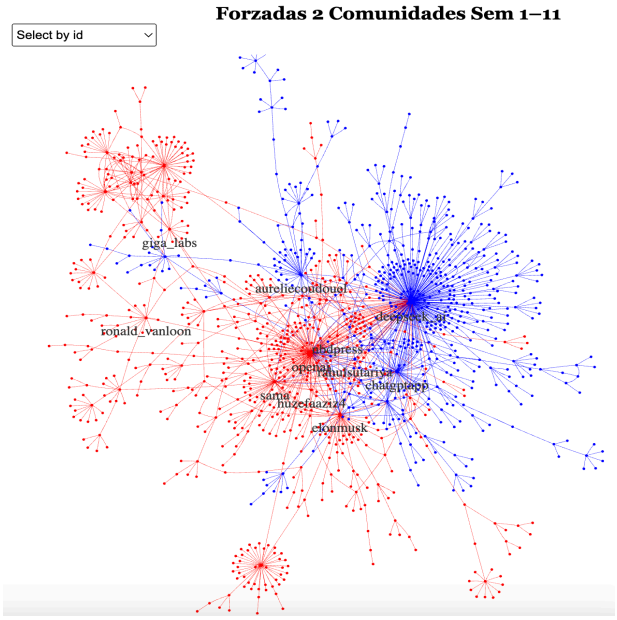


Figura 6.1. *Grafo de comunidades forzadas en 2 grupos (Semanas 1–11)*

A lo largo de las once semanas analizadas, se observa una transformación notable en la red de usuarios vinculados a los modelos de lenguaje de OpenAI y DeepSeek. Desde una red incipiente y dispersa (Semana 1), se pasa gradualmente a una red más compleja, densa y modular.

A partir de la semana 3 aparecen dos grandes bloques, generalmente asociados a los entornos discursivos de DeepSeek y OpenAI. Sin embargo, esta división no es estricta ni estable, en semanas como la 1, 3, 7 o 9, ambos polos aparecen fusionados en una única comunidad o muestran una conectividad elevada entre sí.

Este comportamiento sugiere momentos de convergencia discursiva o ambigüedad estructural, donde los límites ideológicos entre comunidades no están claramente definidos. Actores como deepseek_ai, openai, elonmusk o sama aparecen en ocasiones dentro de la misma comunidad, lo que evidencia que el alineamiento no es puramente binario. Esta característica es coherente con redes reales, donde las afinidades pueden solaparse o fluctuar temporalmente.

Con el paso de las semanas, las comunidades adquieren mayor cohesión interna y una separación más visible. Emergen hubs influyentes en cada comunidad (como deepseek_ai, chatgptapp o betamoroney), y la estructura de la red se aproxima a un tipo *small-world*, caracterizado por una alta agrupación local y trayectorias cortas entre nodos. Esta evolución visual anticipa los valores crecientes del índice JDJ, que capturan la intensificación de la polarización difusa.

En conjunto, la visualización dinámica de la red revela no solo el crecimiento de la actividad y la emergencia de comunidades, sino también las transiciones entre momentos de integración y momentos de separación ideológica. Este análisis permite observar que la polarización no es un fenómeno estático, sino un proceso dinámico y matizado.

La tabla siguiente resume esta evolución desde una perspectiva cuantitativa. Se observa cómo el número de comunidades detectadas de forma natural aumenta progresivamente, alcanzando más de 600 al final del periodo. Al mismo tiempo, la modularidad natural crece de forma constante (de 0.66 a 0.75), lo que indica una mayor cohesión interna dentro de cada comunidad. Por otro lado, la modularidad forzada en dos comunidades (columna "modularidad_2") se mantiene relativamente estable, oscilando entre 0.40 y 0.46, lo que sugiere una separación estructural clara pero no extrema entre los dos bloques ideológicos forzados.

semana	comunidades	modularidad_natural	modularidad_2
1	31	0.6663379	0.4625247
2	88	0.6545633	0.4159837
3	322	0.7225231	0.4336316
4	425	0.7389013	0.4102247
5	497	0.7310979	0.4118259
6	519	0.7412064	0.4223702
7	551	0.7568523	0.4057557
8	596	0.7581332	0.4376329
9	613	0.7539239	0.4077775
10	633	0.7527753	0.4170991
11	642	0.7561309	0.4348189

***Tabla 6.1**. Evolución semanal del número de comunidades detectadas y valores de modularidad*

Esto será crucial para entender cómo evoluciona la polarización difusa, medida mediante el índice JDJ en el siguiente apartado.

6.2 Construcción del índice JDJ

Una vez definida la partición forzada de la red en dos comunidades, se procedió al cálculo del índice JDJ con el objetivo de medir el grado de polarización estructural. El enfoque adoptado consistió en analizar la afinidad local de cada nodo, observando la proporción de sus vecinos inmediatos que pertenecen a cada una de las dos comunidades.

Para cada nodo i, se identificaron sus vecinos (incluyéndose a sí mismo) y se computaron dos proporciones:

- μ_{iA}: fracción de vecinos que pertenecen a la comunidad roja.
- $\mu_{iB} = 1 - \mu_{iA}$: fracción de vecinos que pertenecen a la comunidad azul.

Con esta información se construyó un data frame con tres columnas, el identificador del nodo (ID), su grado de pertenencia con la comunidad A (A), y su grado con la comunidad B (B).

A continuación, se muestra el fragmento de código en R que permitió obtener dicha matriz:

```r
# Crear un dataframe vacío
polarizacion_df <- data.frame(
  ID = character(),
  A = numeric(),  # Proporción de vecinos rojos
  B = numeric(),  # Proporción de vecinos azules
  stringsAsFactors = FALSE
)

# Calcular proporciones para cada nodo
for (i in 1:vcount(mention_graph_gc)) {
  nodo <- V(mention_graph_gc)[i]$name
  vecinos <- unlist(neighborhood(mention_graph_gc, order = 1)[[i]])  # incluye al nodo
  nombres_vecinos <- V(mention_graph_gc)[vecinos]$name
  comunidades_vecinos <- nodes_gc$color[match(nombres_vecinos, nodes_gc$id)]

  mu_a <- sum(comunidades_vecinos == "red", na.rm = TRUE) / length(comunidades_vecinos)
  mu_b <- 1 - mu_a

  polarizacion_df <- rbind(polarizacion_df, data.frame(ID = nodo, A = mu_a, B = mu_b))
}
```

Código 6.1. *Cálculo de proporciones por comunidad*

Posteriormente, el índice JDJ se calculó a partir de esta matriz de pertenencias, comparando cómo de diferentes son las distribuciones de afinidad entre nodos conectados. El cálculo se implementó con la siguiente función:

```r
# ──────────────────────────────────────────────
# CÁLCULO CON JDJ
# ──────────────────────────────────────────────

JDJ <- function(data, r = 4) {
  # data = data frame con ID, A y B.
  # r = redondeo de decimales de mu_a y mu_b
  t <- proc.time()
  data[, 2:3] <- round(data[, 2:3], r)
  # Tabla de frecuencias relativas de combinaciones (mu_a, mu_b)
  a <- data.frame(prop.table(table(data[, 2:3])))
  a <- a[a$Freq != 0, ]
  a$A <- as.numeric(as.character(a$A))
  a$B <- as.numeric(as.character(a$B))
  # Convertir a matriz para operaciones vectorizadas
  a <- as.matrix(a)
  # Matrices cruzadas: muA_i * muB_j y muB_i * muA_j
  AB <- outer(a[, "A"], a[, "B"])
  BA <- outer(a[, "B"], a[, "A"])

  # Tomamos el máximo entre los dos para cada par (i, j)
  JDJ_Full <- pmax(AB, BA) * outer(a[, "Freq"], a[, "Freq"])

  # Calcular valor final de JDJ
  JDJ_value <- sum(JDJ_Full) * 2
  #return(cat(" JDJ =", JDJ_value,
  #           "\nTiempo de computación:", (proc.time() - t)[[3]]))
  return(JDJ_value)
}

# Llamar a la función con el dataframe de polarización
JDJ(polarizacion_df, r = 4)
```

Código 6.2*: Cálculo del Índice JDJ*

Fuente*: Juan Antonio Guevara Gil*

6.3 Cálculo de la proporción de vecinos por comunidad

En la tabla se muestra un fragmento del dataframe generado, donde cada fila representa un nodo de la red (un usuario) y las columnas A y B reflejan el grado de pertenencia a la comunidad roja y azul.

Este desglose resulta clave para observar cómo se distribuyen las afinidades ideológicas de cada nodo. La tabla será utilizada en los apartados posteriores para visualizar dinámicas temporales y construir histogramas de pertenencia, así como para calcular la polarización semanal mediante el índice JDJ.

	ID	A	B
1	coinzonia	1.0000000	0.00000000
2	fsorel	0.0000000	1.00000000
3	dr_m_davis	0.0000000	1.00000000
4	china__focus	1.0000000	0.00000000
5	tut_ml	1.0000000	0.00000000
6	googlesuperbug	0.8333333	0.16666667
7	andygbannister	1.0000000	0.00000000
8	tirotg	0.0000000	1.00000000
9	gauravh1	0.0000000	1.00000000
10	nbdpress	0.8571429	0.14285714
11	moneycontrolcom	1.0000000	0.00000000
12	glengilmore	1.0000000	0.00000000
13	bamitav	1.0000000	0.00000000
14	nirmalyaseo	0.0000000	1.00000000
15	asiapolicy	0.0000000	1.00000000
16	dralkheraiji	0.0000000	1.00000000
17	fabriziobustama	1.0000000	0.00000000
18	anishasircar	0.0000000	1.00000000
19	iphelipevr	1.0000000	0.00000000
20	ronald_vanloon	1.0000000	0.00000000
21	alien71_btc	0.0000000	1.00000000
22	abhishekjha157	1.0000000	0.00000000
23	thecube	0.8571429	0.14285714

Showing 1 to 23 of 1,224 entries, 3 total columns

Tabla 6.2. *Proporciones de vecinos por comunidad para cada nodo*

6.4 Resultados del índice JDJ y análisis de polarización

Los valores obtenidos para los grados de afinidad muestran una distribución altamente polarizada. Como se aprecia en los histogramas a continuación, la gran mayoría de los nodos se sitúan en los extremos del eje (cercanos a 0 o a 1), lo que indica que están rodeados casi exclusivamente por miembros de su propia comunidad.

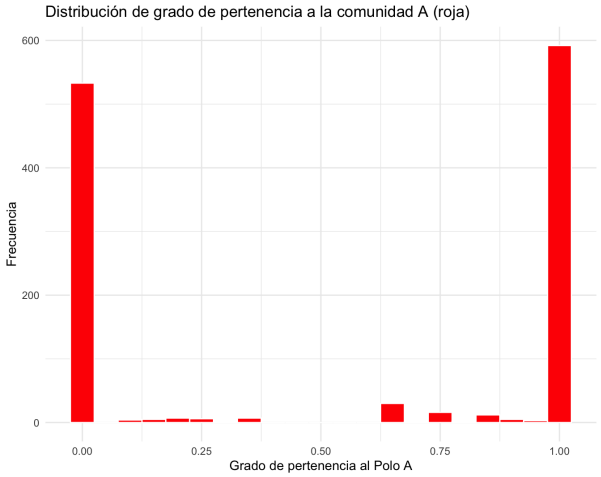


Figura 6.2. *Distribución del grado de pertenencia a la comunidad A (roja)*

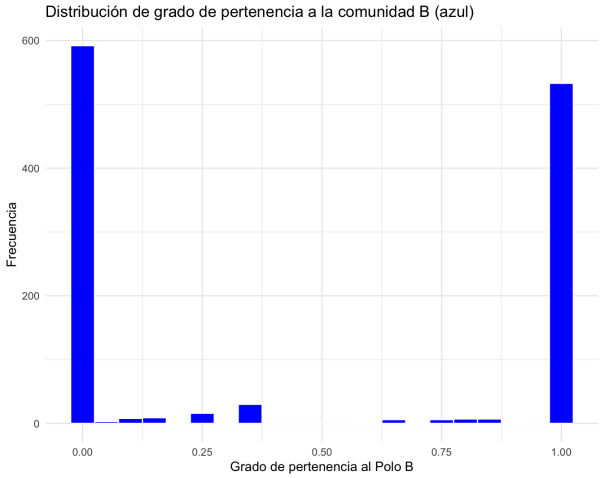


Figura 6.3. *Distribución del grado de pertenencia a la comunidad B (azul)*

Este patrón de concentración extrema en ambos polos sugiere una estructura fuertemente segmentada, donde la mayoría de las interacciones se dan dentro de la misma comunidad, reduciendo la exposición a nodos del grupo opuesto.

Finalmente, el valor de **JDJ = 0.9946** confirma numéricamente este escenario de alta polarización. Un valor tan cercano a 1 indica que, para la mayoría de los pares de nodos conectados, sus entornos inmediatos (sus vecinos) están compuestos por comunidades opuestas, lo cual evidencia una clara disimilitud estructural entre los bloques ideológicos representados en la red.

6.5 Análisis estructural de la red

Con el objetivo de observar la evolución estructural de la red a lo largo del tiempo, se construyeron grafos acumulados para cada semana, considerando exclusivamente el componente gigante (es decir, la subred conexa de mayor tamaño). Este enfoque permite analizar cómo se desarrolla la estructura de interacción entre los usuarios a medida que progresa el debate en redes sociales.

Cada grafo semanal fue analizado mediante una serie de métricas topológicas que permiten caracterizar la estructura de la red desde múltiples dimensiones, número de nodos y aristas, densidad, diámetro, coeficientes de agrupamiento, medidas de cercanía y distribución de distancias mínimas. Además, se calculó el coeficiente de pequeño mundo (*sigma*), que compara la estructura observada con la de un grafo aleatorio equivalente.

6.5.1 Métricas descriptivas semanales

En la tabla se presentan los valores de las principales métricas estructurales para el componente gigante de la red, acumulado semanalmente. Como se puede observar, el número de nodos crece de forma constante hasta estabilizarse hacia la semana 11, con más de 1200 usuarios conectados. La densidad de la red disminuye

progresivamente conforme se incorporan nuevos nodos, lo cual es consistente con redes sociales a gran escala, donde los enlaces no crecen al mismo ritmo que los nodos.

El diámetro de la red se mantiene estable, oscilando entre 12 y 16 pasos, mientras que la distancia media de las rutas entre pares de nodos se estabiliza en torno a los 5.2 pasos. Esta estabilidad sugiere la existencia de rutas cortas entre usuarios, incluso cuando la red crece, un rasgo característico de las redes del tipo *small-world*.

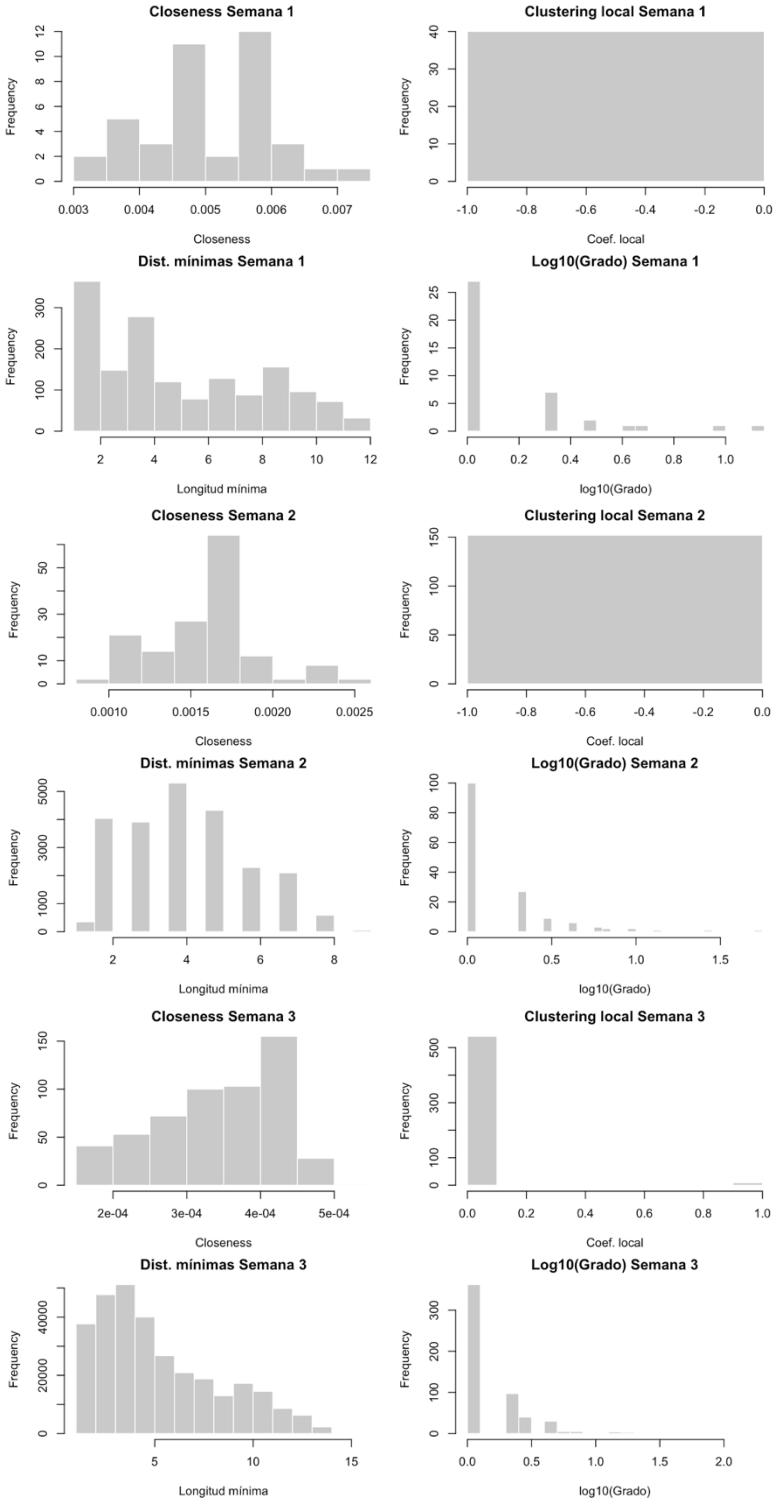
El coeficiente de agrupamiento global (*clustering global*) presenta valores bajos aunque crecientes, lo que indica una tendencia leve a la formación de triángulos o círculos sociales a medida que la red se consolida. A su vez, el coeficiente de agrupamiento local medio, aunque inicialmente nulo en las semanas 1 y 2, también tiende a incrementarse con el tiempo. Finalmente, el parámetro σ (sigma_sw) —que compara la red observada con un grafo aleatorio equivalente— supera el umbral de 1 a partir de la semana 4. Esto confirma que la red satisface las condiciones del modelo de *small-world* propuesto por Watts y Strogatz (1998), al presentar simultáneamente alta cohesión local y trayectorias globales cortas.

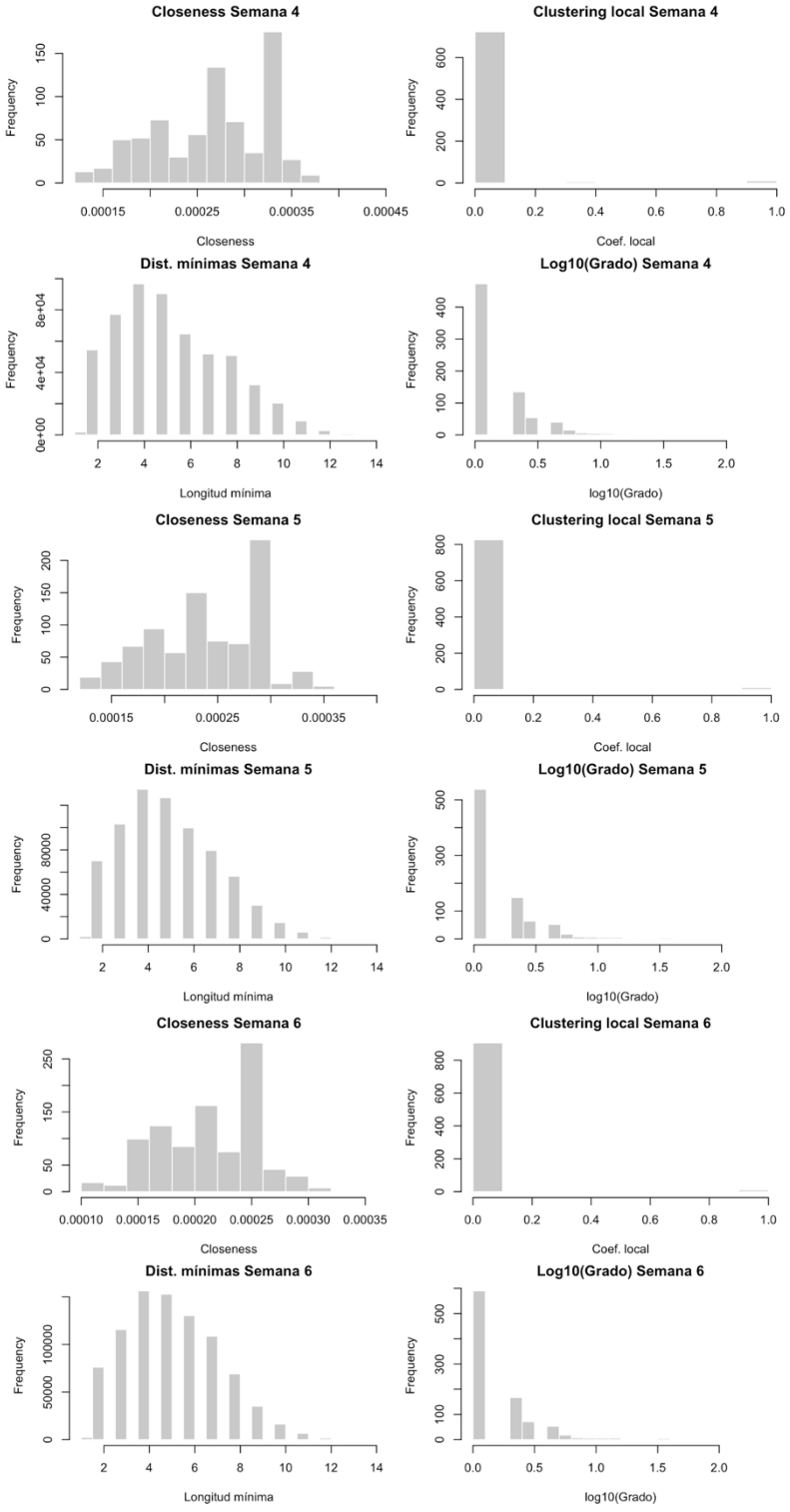
semana	nodos	aristas	densidad	diametro	clust_global	dist_media	C_rand	L_rand	sigma	closeness	clust_local	min_dist
1	40	39	0.05	12	0.0	5.39	0.04348	4.81	0.0	0.0049	0.0	5.39
2	152	175	0.01525	9	0.0	4.21	0.01523	5.39	0.0	0.0016	0.0	4.21
3	553	651	0.00427	16	0.00186	5.73	0.00987	6.97	0.22	0.0003	0.0189	5.73
4	744	908	0.00329	14	0.00255	5.38	0.00271	7.09	1.24	0.0003	0.0190	5.38
5	852	1070	0.00295	14	0.00369	5.15	0.00434	6.95	1.15	0.0002	0.0206	5.15
6	934	1168	0.00268	14	0.00409	5.24	0.00202	7.15	2.77	0.0002	0.0204	5.24
7	1041	1306	0.00241	13	0.00388	5.25	0.00366	7.29	1.47	0.0002	0.0191	5.25
8	1139	1430	0.00221	14	0.00341	5.3	0.00085	7.57	5.75	0.0002	0.0174	5.3
9	1180	1489	0.00214	15	0.00323	5.17	0.00485	7.38	0.95	0.0002	0.0161	5.17
10	1211	1531	0.00209	15	0.00343	5.18	0.00078	7.39	6.27	0.0002	0.0169	5.18
11	1224	1546	0.00207	15	0.00349	5.2	0.00385	7.37	1.29	0.0002	0.0179	5.2

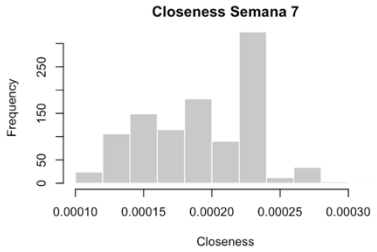
Tabla 6.3. *Evolución de métricas estructurales del componente gigante por semana*

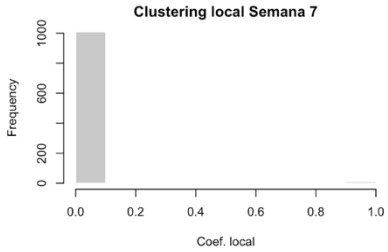
6.5.2 Clasificación del tipo de red por semana

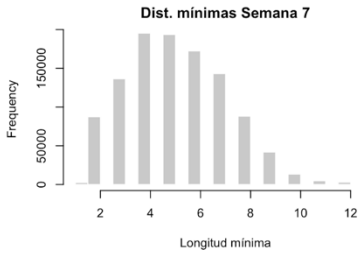
Con el objetivo de comprender la evolución estructural de la red a lo largo del tiempo, se realizó una clasificación cualitativa del tipo de red observada en cada semana. Esta clasificación se basó en una combinación de métricas estructurales y en el análisis visual de histogramas generados para cada período. Las variables representadas son, el coeficiente de agrupamiento (global y local), la distancia media, la distribución del grado (log10), la distribución de las distancias mínimas, y el coeficiente de pequeño mundo σ (Watts & Strogatz, 1998).

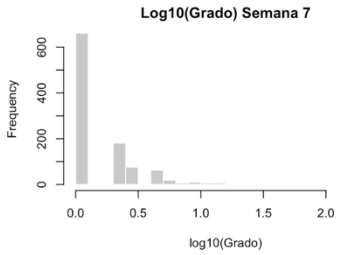

Closeness Semana 1
Frequency
Closeness
Clustering local Semana 1
Coef. local
Dist. mínimas Semana 1
Longitud mínima
Log10(Grado) Semana 1
log10(Grado)
Closeness Semana 2
Clustering local Semana 2
Dist. mínimas Semana 2
Log10(Grado) Semana 2
Closeness Semana 3
Clustering local Semana 3
Dist. mínimas Semana 3
Log10(Grado) Semana 3

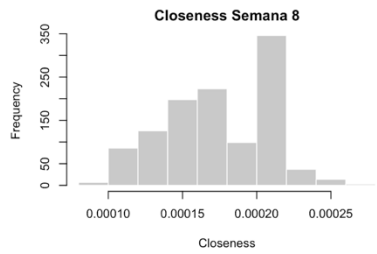

Closeness Semana 4
Frequency
Closeness
Clustering local Semana 4
Coef. local
Dist. mínimas Semana 4
Longitud mínima
Log10(Grado) Semana 4
log10(Grado)
Closeness Semana 5
Clustering local Semana 5
Dist. mínimas Semana 5
Log10(Grado) Semana 5
Closeness Semana 6
Clustering local Semana 6
Dist. mínimas Semana 6
Log10(Grado) Semana 6

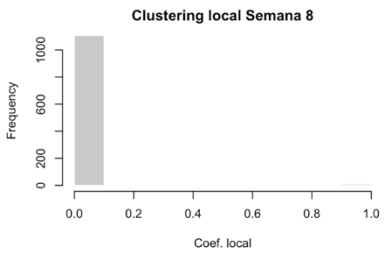

Closeness Semana 7
Frequency
0
50
150
250
0.00010
0.00015
0.00020
0.00025
0.00030
Closeness

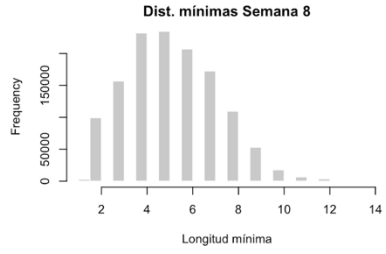

Clustering local Semana 7
Frequency
0
200
600
1000
0.0
0.2
0.4
0.6
0.8
1.0
Coef. local

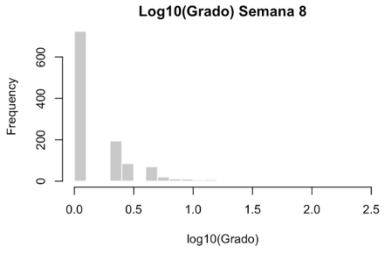

Dist. mínimas Semana 7
Frequency
0
50000
150000
2
4
6
8
10
12
Longitud mínima

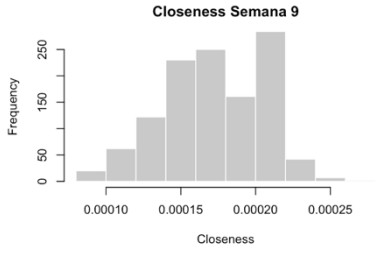

Log10(Grado) Semana 7
Frequency
0
200
400
600
0.0
0.5
1.0
1.5
2.0
log10(Grado)

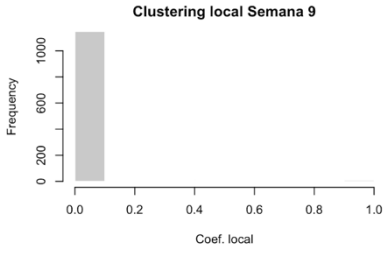

Closeness Semana 8
Frequency
0
50
150
250
350
0.00010
0.00015
0.00020
0.00025
Closeness

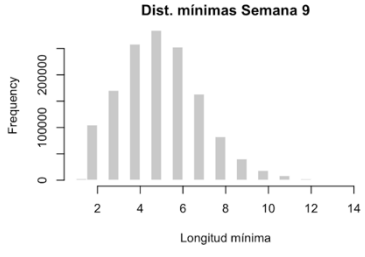

Clustering local Semana 8
Frequency
0
200
600
1000
0.0
0.2
0.4
0.6
0.8
1.0
Coef. local

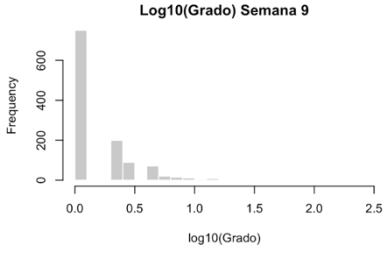

Dist. mínimas Semana 8
Frequency
0
50000
150000
2
4
6
8
10
12
14
Longitud mínima


Log10(Grado) Semana 8
Frequency
0
200
400
600
0.0
0.5
1.0
1.5
2.0
2.5
log10(Grado)


Closeness Semana 9
Frequency
0
50
150
250
0.00010
0.00015
0.00020
0.00025
Closeness


Clustering local Semana 9
Frequency
0
200
600
1000
0.0
0.2
0.4
0.6
0.8
1.0
Coef. local


Dist. mínimas Semana 9
Frequency
0
100000
200000
2
4
6
8
10
12
14
Longitud mínima


Log10(Grado) Semana 9
Frequency
0
200
400
600
0.0
0.5
1.0
1.5
2.0
2.5
log10(Grado)

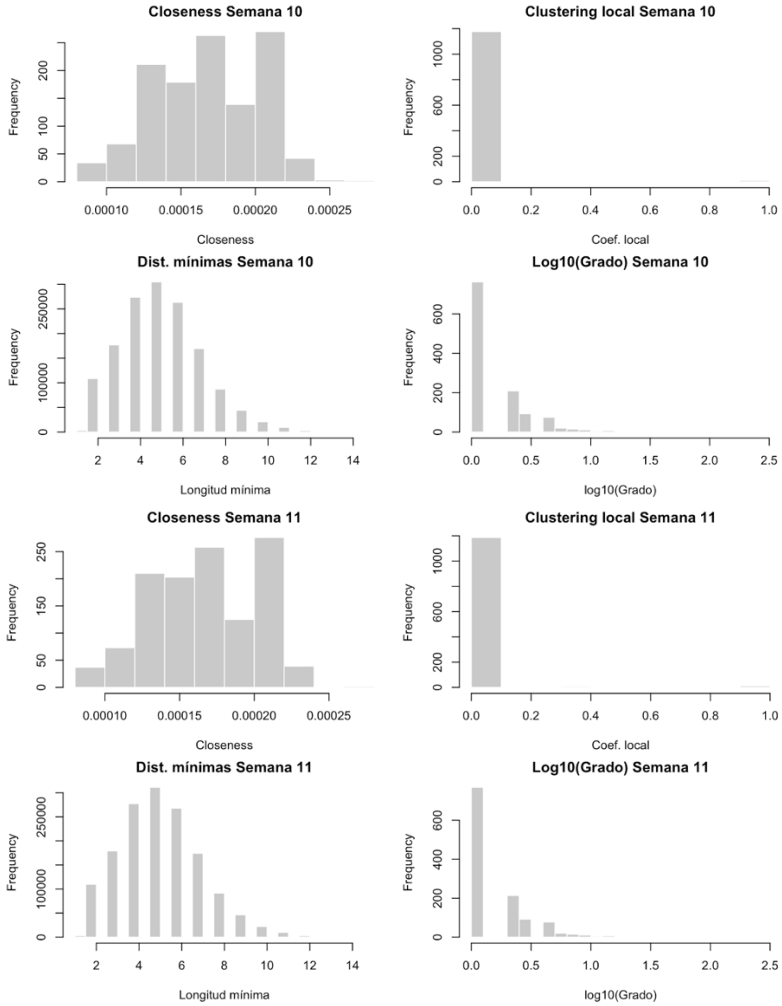


Figura 6.4. Histogramas de métricas estructurales (closeness, clustering, grado y dist. mínimas)

En particular, la detección de estructuras tipo *small-world* se basó en la combinación de dos condiciones, un valor de σ superior a 1 y un coeficiente de agrupamiento (global o local) significativamente mayor al de una red aleatoria equivalente. Además, el análisis de la forma de la distribución del grado (ya sea normal, sesgada o con cola larga) permitió identificar la aparición de *hubs* y el surgimiento de una organización jerárquica en la red.

Cabe destacar que en las semanas 1 y 2 el coeficiente de agrupamiento local aparece cercano a cero para la mayoría de los nodos. Esto no representa un error, sino que refleja una característica propia de redes emergentes en etapas iniciales, donde la conectividad es aún baja. En estos casos, la mayoría de los nodos tienen un grado igual o menor a uno, lo que impide la formación de triángulos. Por tanto, el valor promedio del *clustering local* se define como 0. Esta situación también se manifiesta en histogramas con escalas atípicas debido a la falta de variabilidad estructural.

La tabla siguiente resume esta clasificación, identificando para cada semana el tipo de red predominante, junto con una breve explicación basada en las métricas y visualizaciones:

Semana	σ	C global	Dist. media	Grado (log10)	Clust. local	Dist. mínimas	Tipo de Red	Explicación
1	0.0	0.0	5.39	Poisson-like	Casi todo 0	Pico en 2–6	Estrella / Árbol disperso	Un solo nodo central, sin ciclos ni grupos → grado bajo y uniforme.
2	0.0	0.0	4.21	Poisson-like	Casi todo 0	Distribución más compacta	Estrella (más grande)	Igual que semana 1 pero con más nodos; sigue sin hubs ni clustering.
3	0.22	0.0018	5.73	Cola leve	Empieza a subir	Amplia derecha	Red semiestructurada	Aparecen primeros ciclos, clustering > 0, leve desviación del grado.
4	1.24	0.0026	5.38	Cola clara	Pico en 0 + cola	Distribución amplia	Small-world (incipiente)	σ > 1 + clustering ≫ 0 → estructura social empieza.
5	1.15	0.0037	5.15	Heavy-tail	Distribución parecida a sem 5	Más corta	Small-world + Hubs emergentes	Hubs definidos, grados desiguales, caminos cortos.
6	2.77	0.0041	5.24	Heavy-tail	Igual que 6	Pico en 4–6	Small-world fuerte	σ > 2.5, fuerte estructura social con hubs.
7	1.47	0.0039	5.25	Heavy-tail	Igual patrón	Pico en 5–6	Small-world consolidada	Hubs persistentes, red estabilizada, estructura densa.
8	5.75	0.0034	5.3	Heavy-tail	Igual que 7–8	Pico en 5–6	Ultra small-world (con superhubs)	σ récord, hubs muy centrales y conectados.
9	0.95	0.0032	5.17	Heavy-tail	Igual que 9	Pico en 5–6	Small-world estable	σ ligeramente bajo pero estructura idéntica. No es aleatoria.
10	6.27	0.0034	5.18	Heavy-tail	Idéntica forma	Pico en 5–6	Small-world con hubs marcados	σ vuelve a subir → sigue consolidada con hubs fuertes.
11	1.29	0.0035	5.2	Heavy-tail	Idéntica forma	Pico en 5–6	Small-world estable final	Sin cambios relevantes. Red madura, polarizada y con comunidades.

***Tabla 6.4**. Clasificación del tipo de red por semana*

Esta interpretación se sustenta no solo en los valores agregados, sino también en los histogramas individuales por métrica. Estos permiten identificar desviaciones, asimetrías o concentraciones locales que podrían pasar desapercibidas si solo se consideran promedios. Por ejemplo, aunque en las semanas 1 y 2 se observe un valor puntual de *clustering* local igual a 0.5 en algunos histogramas, este valor corresponde a una minoría de nodos aislados. El valor medio global permanece cercano a 0, coherente con estructuras sin ciclos ni triángulos.

En conjunto, el análisis muestra un patrón claro de consolidación estructural a lo largo del tiempo, desde redes dispersas y sin cohesión en las primeras semanas (estructuras tipo estrella o árbol), hasta redes densas y organizadas con *hubs* estables

y trayectorias cortas. Este comportamiento es característico de redes sociales maduras con propiedades de *small-world*.

6.6 JDJ semanal: evolución de la polarización acumulada

El índice JDJ permite medir el grado de polarización estructural en la red, capturando cómo de separados están los entornos inmediatos de los nodos en términos de afinidad con las dos comunidades forzadas (roja y azul). A lo largo de las once semanas analizadas, se observa que los valores del JDJ son consistentemente elevados, oscilando entre 0.90 y 0.99. Esto confirma una segmentación pronunciada, donde la mayoría de los usuarios interactúa principalmente con miembros de su propia comunidad.

semana	JDJ
1	0.9729799
2	0.9654153
3	0.9657758
4	0.9242206
5	0.9585359
6	0.9575028
7	0.908427
8	0.9961973
9	0.9253643
10	0.9404482
11	0.9946591

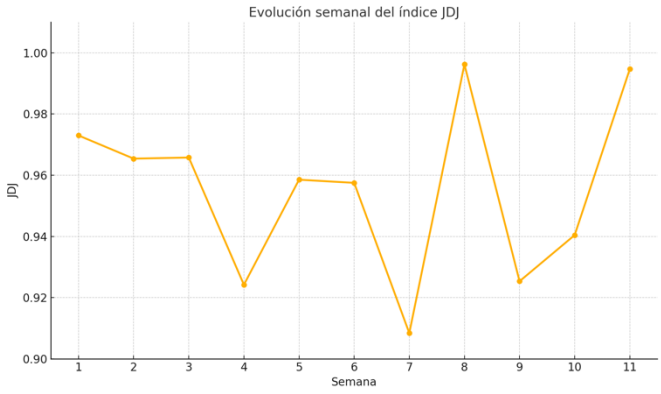


***Tabla 6.5 y gráfico 6.1**. Evolución semanal del índice JDJ acumulada*

No obstante, el índice no se mantiene completamente estable, sino que presenta ciertas oscilaciones significativas. En las semanas 4, 7 y 9 se detectan descensos notables en el valor del JDJ, alcanzando mínimos relativos de 0.92, 0.90 y 0.92 respectivamente. Estos descensos indican momentos de menor polarización estructural, posiblemente asociados a fases de convergencia discursiva o mayor interacción entre comunidades.

Este patrón encuentra respaldo en la visualización de los grafos semanales. En las semanas 7 y 9 —que en parte coinciden con los descensos en JDJ— se observa cómo los nodos de OpenAI y DeepSeek aparecen en la misma comunidad forzada, o al menos muestran vínculos significativos entre sí. La partición visual es menos nítida, y los hubs de cada lado (deepseek_ai, openai, chatgptapp) aparecen conectados o incluso integrados en un mismo bloque.

Asimismo, el análisis de los Top 3 usuarios por comunidad refleja un comportamiento híbrido, en semanas como la 4 o la 9, OpenAI figura en ambas comunidades forzadas, lo cual sugiere una posición estructural intermedia o ambigua dentro de la red.

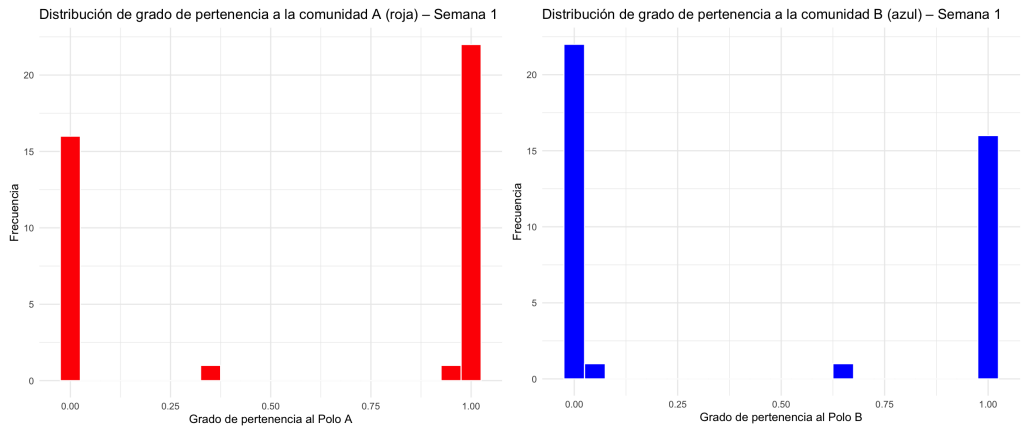
Pese a estas fluctuaciones, la mayoría de los valores de JDJ se mantienen por encima de 0.94, e incluso se alcanzan picos de 0.996 (semana 8) y 0.994 (semana 11), lo cual indica una estructura fuertemente polarizada en la que los usuarios tienden a agruparse en comunidades claramente diferenciadas. Estos picos coinciden con momentos de consolidación de hubs (e.g., deepseek_ai, chatgptapp, betamoroney), reflejando una cohesión interna más fuerte y una menor interacción intercomunitaria.

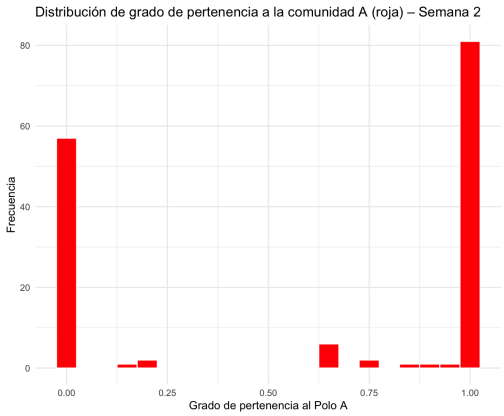
Cabe destacar que los picos de polarización observados en las semanas 8 y 11 coinciden con eventos noticiosos relevantes. La semana 8 (9-16 de marzo) se observó con el anuncio del juicio acelerado entre Elon Musk y OpenAI, relacionado con la conversión de la organización en una entidad con fines de lucro. Esta disputa generó amplias reacciones en redes sociales, donde se intensificaron las posturas tanto a favor como en contra de la dirección actual de OpenAI. Además, comenzaron a circular discusiones técnicas sobre la inminente adopción del *Model Context Protocol (MCP)*, lo que añadió otra capa de polarización entre defensores de la apertura tecnológica y críticos del modelo cerrado. En la semana 11 (30 de marzo-2 de abril), se refleja el foco principal de la integración oficial del *Model Context Protocol* por parte de OpenAI en su ecosistema de herramientas, lo cual reactivó el debate sobre la interoperabilidad y el control sobre los modelos. Aunque DeepSeek no realizó anuncios relevantes durante este periodo, la intensidad del intercambio generado por las decisiones estratégicas de OpenAI fue suficiente para consolidar una segmentación fuerte entre comunidades.

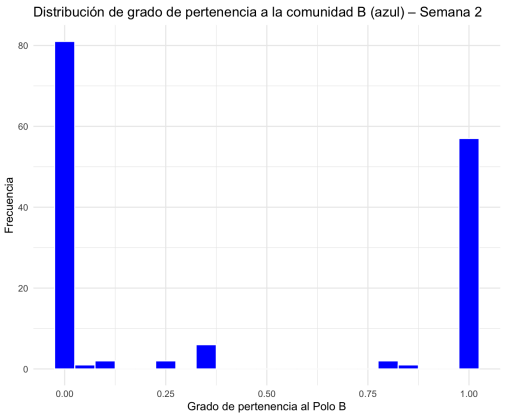
En resumen, aunque el índice JDJ evidencia una alta polarización estructural constante, existen fluctuaciones significativas que reflejan momentos de mayor ambigüedad o interacción entre polos. Estas transiciones no solo se manifiestan en las métricas cuantitativas, sino que también son visibles en los grafos, la composición de autores líderes en cada comunidad y la forma de las distribuciones de pertenencia.

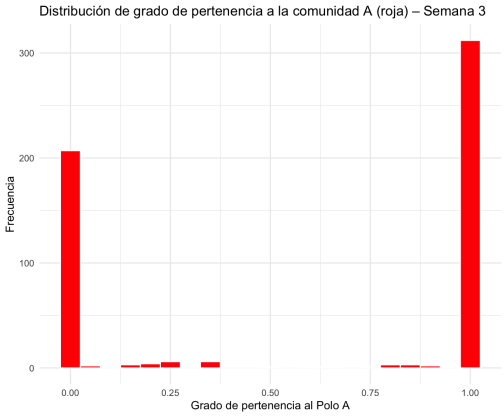
6.6.1 Visualización de histogramas de grado de pertenencia

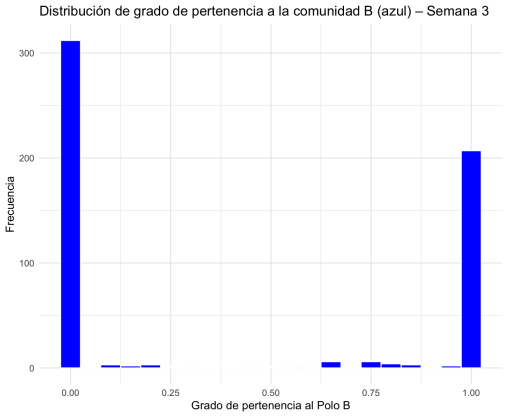
Los histogramas semanales de grado de pertenencia permiten observar con mayor detalle la distribución ideológica interna de cada comunidad, más allá de su asignación binaria a los polos rojo (OpenAI) o azul (DeepSeek). Estos gráficos muestran cómo de alineados están los nodos con cada uno de los polos, y permiten detectar momentos de ambigüedad, transición o radicalización.

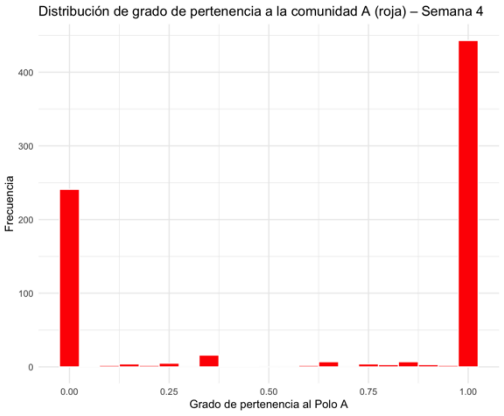


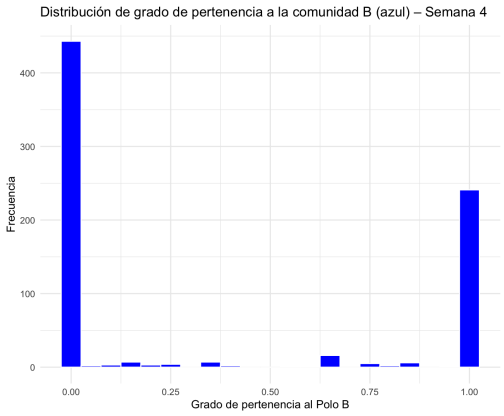

Distribución de grado de pertenencia a la comunidad A (roja) – Semana 2
Frecuencia
Grado de pertenencia al Polo A

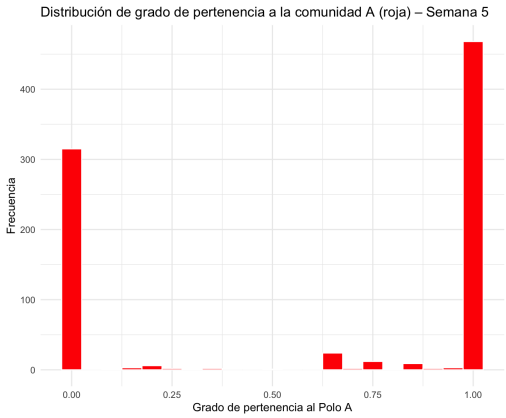

Distribución de grado de pertenencia a la comunidad B (azul) – Semana 2
Frecuencia
Grado de pertenencia al Polo B

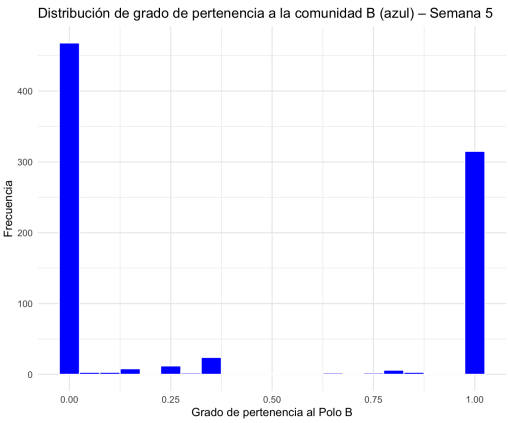

Distribución de grado de pertenencia a la comunidad A (roja) – Semana 3
Frecuencia
Grado de pertenencia al Polo A

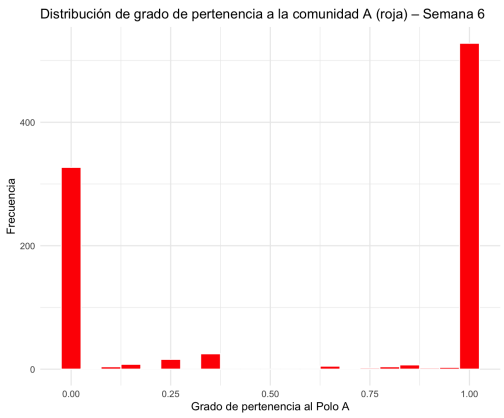

Distribución de grado de pertenencia a la comunidad B (azul) – Semana 3
Frecuencia
Grado de pertenencia al Polo B

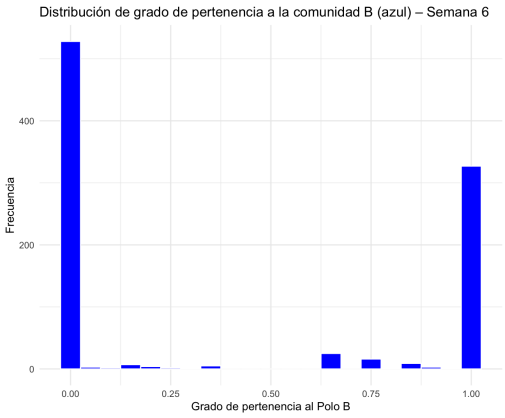

Distribución de grado de pertenencia a la comunidad A (roja) – Semana 4
Frecuencia
Grado de pertenencia al Polo A

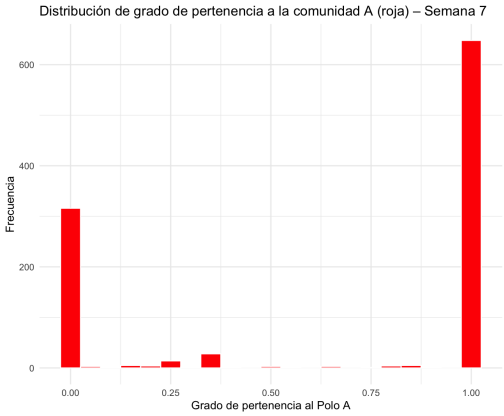

Distribución de grado de pertenencia a la comunidad B (azul) – Semana 4
Frecuencia
Grado de pertenencia al Polo B

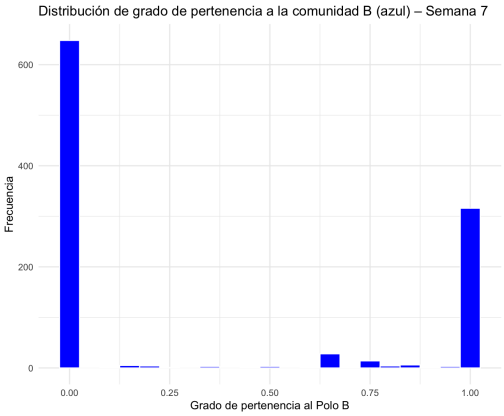

Distribución de grado de pertenencia a la comunidad A (roja) – Semana 5
Frecuencia
Grado de pertenencia al Polo A

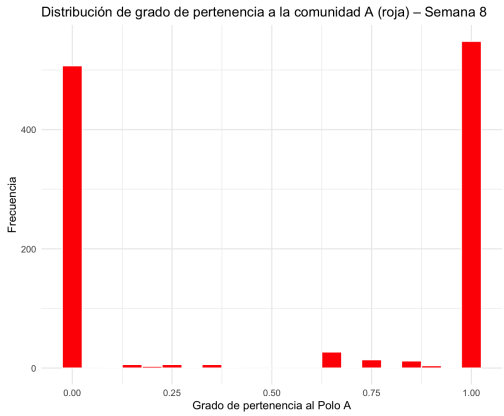

Distribución de grado de pertenencia a la comunidad B (azul) – Semana 5
Frecuencia
Grado de pertenencia al Polo B

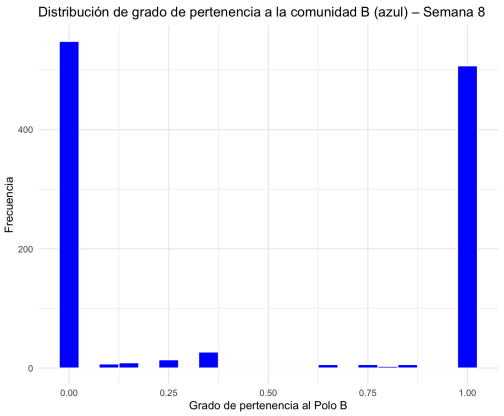

Distribución de grado de pertenencia a la comunidad A (roja) – Semana 6
Frecuencia
0
200
400
0.00
0.25
0.50
0.75
1.00
Grado de pertenencia al Polo A

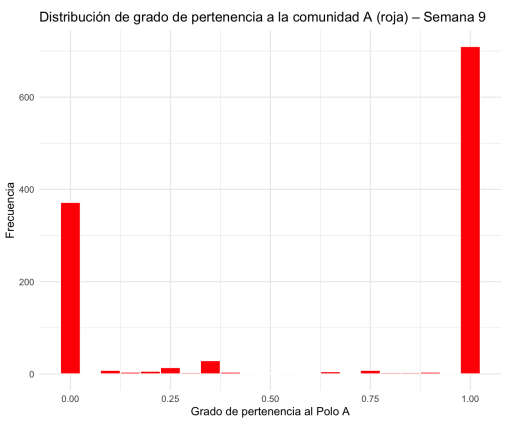

Distribución de grado de pertenencia a la comunidad B (azul) – Semana 6
Frecuencia
0
200
400
0.00
0.25
0.50
0.75
1.00
Grado de pertenencia al Polo B

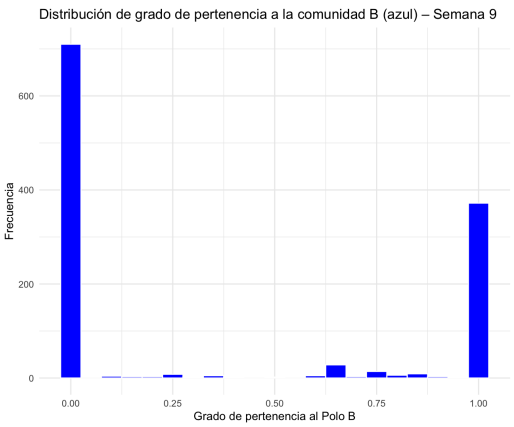

Distribución de grado de pertenencia a la comunidad A (roja) – Semana 7
Frecuencia
0
200
400
600
0.00
0.25
0.50
0.75
1.00
Grado de pertenencia al Polo A


Distribución de grado de pertenencia a la comunidad B (azul) – Semana 7
Frecuencia
0
200
400
600
0.00
0.25
0.50
0.75
1.00
Grado de pertenencia al Polo B


Distribución de grado de pertenencia a la comunidad A (roja) – Semana 8
Frecuencia
0
200
400
0.00
0.25
0.50
0.75
1.00
Grado de pertenencia al Polo A


Distribución de grado de pertenencia a la comunidad B (azul) – Semana 8
Frecuencia
0
200
400
0.00
0.25
0.50
0.75
1.00
Grado de pertenencia al Polo B


Distribución de grado de pertenencia a la comunidad A (roja) – Semana 9
Frecuencia
0
200
400
600
0.00
0.25
0.50
0.75
1.00
Grado de pertenencia al Polo A


Distribución de grado de pertenencia a la comunidad B (azul) – Semana 9
Frecuencia
0
200
400
600
0.00
0.25
0.50
0.75
1.00
Grado de pertenencia al Polo B

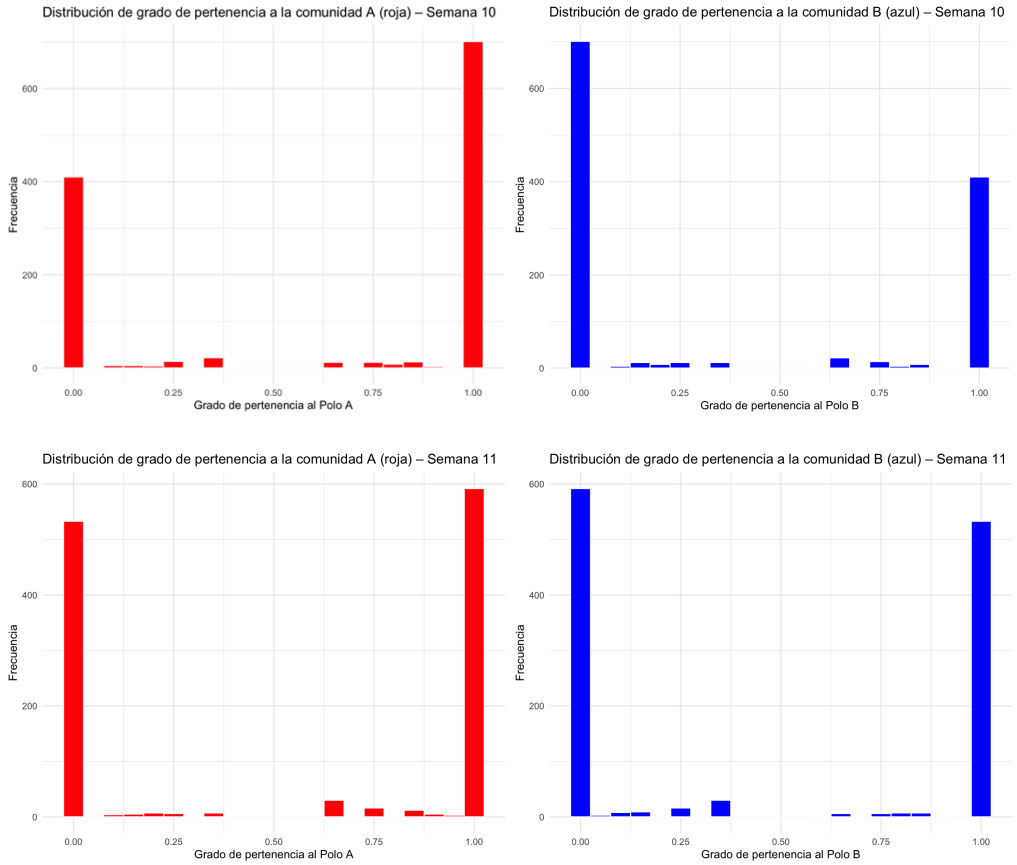


***Figura 6.5**. Evolución de grado de pertenencia de semana 1 al 11*

Desde la Semana 1, la mayoría de nodos presentan un grado de pertenencia cercano a 0 o 1, lo que sugiere una primera estructura polarizada. Sin embargo, aún existen usuarios con valores intermedios.

A partir de la Semana 3, el patrón se consolida. Los histogramas muestran una polarización más marcada, con un mayor número de nodos completamente alineados (grado ≈ 1) con su polo respectivo. No obstante, en semanas como la semana 4, semana 7 y semana 9, se observa un aumento en la proporción de nodos con pertenencia intermedia (entre 0.25 y 0.75). Estas semanas coinciden con los descensos en el índice JDJ y con los momentos donde los grafos muestran convergencia o mezcla entre comunidades, como se analizó anteriormente. Es decir, las distribuciones reflejan un mayor grado de ambigüedad.

En contraste, las semanas 8 y 11 muestran distribuciones altamente bimodales y extremas. La mayoría de nodos están claramente posicionados en los extremos del eje de pertenencia (cerca de 0 o de 1), tanto en la comunidad roja como en la azul. Este comportamiento indica un alto grado de cohesión ideológica y una menor ambigüedad, lo que se corresponde con los picos más altos del índice JDJ (0.996 y 0.994 respectivamente).

En conjunto, los histogramas permiten confirmar visualmente la evolución dinámica de la polarización, de una red inicialmente mixta y con ambigüedad estructural,

se transita hacia estructuras más segmentadas y polarizadas, con algunas semanas de relajamiento ideológico intermedio. Esta representación complementa y enriquece los hallazgos obtenidos mediante el índice JDJ y las visualizaciones de grafos.

6.7 Comparación entre comunidades (OpenAI vs DeepSeek)

Con el fin de caracterizar las dos comunidades principales detectadas (asociadas de forma predominante con OpenAI y DeepSeek), se analizan sus métricas estructurales semana a semana, como el número de nodos y aristas, densidad, diámetro y los actores más influyentes en cada comunidad. Esta comparación permite observar no solo el crecimiento interno de cada comunidad, sino también su evolución relativa y sus diferencias internas.

semana	N_nodos_OpenAI	N_aristas_OpenAI	Densidad_OpenAI	Diametro_OpenAI	Top3_Comunidad_A
1	23	22	0.08695652	5	deepseek_ai, openai, abelramosc
2	92	100	0.023889154	9	openai, sarbjeetjohal, aureliecoudouel
3	323	379	0.007288041	11	deepseek_ai, openai, chatgptapp
4	470	531	0.004817856	14	deepseek_ai, sama, chatgptapp
5	522	633	0.004655062	13	openai, betamoroney, elonmusk
6	552	662	0.004353087	20	deepseek_ai, betamoroney, chatgptapp
7	662	763	0.003487346	13	deepseek_ai, openai, youtube
8	608	740	0.004010232	15	openai, betamoroney, youtube
9	736	850	0.003142561	18	deepseek_ai, openai, youtube
10	749	883	0.003152153	15	deepseek_ai, chatgptapp, elonmusk
11	661	804	0.003685875	16	openai, betamoroney, elonmusk

Tabla 6.6. *Métricas estructurales de la comunidad A*

semana	N_nodos_DS	N_aristas_DS	Densidad_DS	Diametro_DS	Top3_Com_B
1	17	16	0.117647059	8	aureliecoudouel, david72386784, grok_meme_bnb
2	60	64	0.036158192	5	deepseek_ai, chatgptapp, alphaproxima17
3	230	243	0.009227264	20	betamoroney, youtube, aureliecoudouel
4	274	320	0.008555921	13	openai, betamoroney, bamitav
5	330	374	0.006889564	12	deepseek_ai, chatgptapp, grok
6	382	437	0.006005139	12	openai, elonmusk, youtube

7	379	456	0.006365959	20	betamoroney, chatgptapp, aureliecoudouel
8	531	607	0.004313684	14	deepseek_ai, chatgptapp, aureliecoudouel
9	444	535	0.005439977	18	chatgptapp, betamoroney, aureliecoudouel
10	462	556	0.005221098	15	openai, betamoroney, youtube
11	563	649	0.004102324	15	deepseek_ai, chatgptapp, aureliecoudouel

Tabla 6.7. *Métricas estructurales de la Comunidad B*

Tamaño y crecimiento

Ambas comunidades muestran una expansión progresiva en número de nodos y conexiones, aunque con algunas diferencias notables. La comunidad asociada a **OpenAI** (Comunidad A) comienza con 23 nodos en la semana 1 y alcanza un pico de 749 en la semana 10, estabilizándose luego en torno a los 660 nodos. Por otro lado, la comunidad de **DeepSeek** (Comunidad B) parte de una base aún más reducida (17 nodos) pero crece de forma constante hasta superar los 560 en la semana 11. Esta tendencia refleja una consolidación paralela de ambos polos discursivas, con tasas de crecimiento similares en términos generales.

Densidad

Las métricas de densidad informan que, en las primeras semanas, la comunidad de DeepSeek presenta una mayor cohesión interna. Por ejemplo, en la semana 1, su densidad es de 0.1176 frente al 0.0869 de OpenAI. Sin embargo, a medida que ambas redes se expanden, estas densidades disminuyen —como es típico en redes que crecen en tamaño—, y tienden a estabilizarse en valores bajos pero similares (en torno a 0.004 en las últimas semanas).

Diámetro de red

El análisis del diámetro —la distancia máxima entre nodos dentro de cada comunidad— muestra un comportamiento más variable. Por ejemplo, la comunidad de OpenAI alcanza su máximo diámetro en la semana 6 (20), mientras que la de DeepSeek presenta picos tanto en la semana 3 como en la 7. Estos cambios sugieren fluctuaciones en la forma en que se articulan las conexiones internas de cada grupo, con momentos de mayor dispersión o centralización.

Actores principales

El seguimiento de los tres actores más centrales en cada comunidad refuerza la interpretación anterior. A lo largo del tiempo, se repiten nodos clave en cada polo, **deepseek_ai**, **chatgptapp** y **aureliencoudouel** dominan la comunidad DeepSeek, mientras que **openai**, **betamorneoy**, **elonmusk** y **chatgptapp** son nodos claves en la comunidad OpenAI. Sin embargo, también se observan momentos de ambigüedad o solapamiento, como en las Semanas 3, 7 y 9, en las que algunos usuarios aparecen

como influyentes en ambas comunidades o en roles intercambiables. Este fenómeno coincide con las semanas de menor polarización observadas en el índice JDJ, lo que sugiere que la estructura de las comunidades y su separación ideológica no son completamente estables ni dicotómicas.

En resumen, el análisis comparado de ambas comunidades presenta trayectorias de crecimiento similares, pero con dinámicas internas distintas. Mientras que DeepSeek mostró una mayor cohesión inicial, OpenAI desarrolló una estructura más extensa en etapas posteriores. Los líderes de opinión en cada polo consolidan sus posiciones con el tiempo, pero la existencia de conexiones cruzadas e influencias compartidas muestra que la polarización discursiva es un proceso dinámico y no estrictamente binario.

7. CONCLUSIONES

7.1 Principales hallazgos

Este trabajo ha analizado la evolución de una red social en Twitter/X en torno al debate sobre los modelos de lenguaje desarrollados por OpenAI y DeepSeek. A través del uso combinado de teoría de grafos, lógica difusa y análisis estructural, se ha abordado el fenómeno de la polarización desde una doble perspectiva, una visión estructural, basada en modularidad y partición de comunidades, y una visión ideológica difusa, representada por el índice JDJ.

Los resultados muestran que la red crece de manera sostenida durante las once semanas analizadas, adoptando progresivamente una estructura tipo *small-world*, con mayor densidad interna, aparición de *hubs* (como @elonmusk, @deepseek_ai, @chatgptapp o @Openai) y trayectorias cortas entre nodos. A nivel comunitario, se observa una tendencia a la consolidación de dos grandes bloques, aunque con momentos de solapamiento o convergencia entre ellos.

El índice JDJ muestra niveles de polarización elevados desde las primeras semanas, confirmando una fuerte segmentación en dos comunidades diferenciadas. En particular, los valores más altos se registran en la semana 8 y la semana 11. Estas fechas coinciden con eventos clave que intensificaron el debate en redes. En la semana 8 (del 9 al 16 de marzo), se produjo un repunte en la atención pública tras el anuncio del juicio acelerado entre Elon Musk y OpenAI, en el marco de la disputa legal por la conversión de la organización en una entidad con fines de lucro. Este conflicto generó posturas enfrentadas en redes sociales. Paralelamente, comenzaron a circular discusiones técnicas sobre la adopción del *Model Context Protocol* (MCP), lo que amplificó la polarización entre defensores de la interoperabilidad abierta y sectores más críticos. Posteriormente, en la semana 11 (30 de marzo al 2 de abril), se concretó la integración oficial del MCP en las herramientas de desarrollo de OpenAI, lo que reactivó el debate técnico y estratégico sobre el control del ecosistema de modelos.

Estos picos de polarización también se reflejan en los histogramas de grado de pertenencia, que muestran distribuciones extremadamente bimodales, donde la mayoría

de los nodos se alinean hacia uno de los polos ideológicos. En los grafos semanales, los hubs centrales como @deepseek_ai, @elonmusk, @chatgptapp o @OpenAI aparecen fuertemente integrados en sus respectivas comunidades, reduciendo al mínimo los vínculos entre bloques opuestos.

Por el contrario, semanas como la 4, 7 y 9 presentan caídas significativas en el índice JDJ, lo que sugiere una relajación temporal de la polarización. Durante estas fases, los grafos revelan una mayor presencia de conexiones cruzadas y nodos con grados de pertenencia intermedios. En algunos casos, actores como OpenAI aparecen situados en ambas comunidades forzadas, lo que sugiere una posición estructural ambigua o híbrida. Estos momentos podrían interpretarse como ventanas de mayor interacción discursiva entre comunidades, reflejando una menor alineación ideológica.

En conjunto, los hallazgos indican que la polarización no es un fenómeno estático, sino dinámico y sensible tanto a la configuración estructural de la red como a eventos externos. La combinación del enfoque estructural y difuso permite capturar tanto la forma general de la segmentación como los matices de pertenencia ideológica, aportando una visión más completa del comportamiento colectivo en entornos digitales polarizados.

7.2 Limitaciones del estudio

Este trabajo presenta varias limitaciones que es importante reconocer. En primer lugar, la elección de una partición binaria puede no reflejar adecuadamente la complejidad del espacio ideológico real. Es posible que existan más de dos grupos relevantes o que algunos usuarios no se alineen claramente con ninguna comunidad.

Además, el supuesto de que los vínculos entre usuarios reflejan afinidad ideológica puede no cumplirse siempre. En redes sociales, las interacciones pueden surgir por desacuerdo, interés estratégico o simple visibilidad, lo que puede alterar las estimaciones del grado de pertenencia.

También debe considerarse que el análisis se ha centrado exclusivamente en la estructura de la red, sin incorporar datos sobre el contenido de los mensajes. Esta ausencia de análisis semántico limita la capacidad para identificar con mayor precisión las posiciones ideológicas de los usuarios.

Por último, el periodo de observación se restringe a once semanas y a una temática específica (Openai y Deepseek). Esto limita la generalización de los resultados, que podrían variar significativamente en otros contextos o debates.

7.3 Recomendaciones y posibles líneas futuras

En el futuro, sería interesante ampliar el análisis incluyendo el contenido textual de los mensajes, con el fin de contrastar los patrones estructurales con las opiniones expresadas directamente. Esto permitiría validar si los grupos detectados

estructuralmente reflejan diferencias temáticas reales. También resultaría útil extender el análisis a otros temas sociales o políticos, para comprobar si las dinámicas observadas en este estudio se repiten en otros contextos. Asimismo, observar redes durante periodos más largos o ante eventos relevantes podría arrojar información adicional sobre la estabilidad o transformación de las comunidades.

Finalmente, se podría investigar más sobre los eventos externos (como lanzamientos de productos o controversias públicas) afectan la estructura de la red y el índice JDJ, con el fin de comprender mejor la relación entre contexto y polarización.

8. REFERENCIAS BIBLIOGRÁFICAS

- Albornoz, M., & Alfaraz, C. (2006). Redes de conocimiento: construcción, dinámica y gestión.
- Barabási, A. L., & Albert, R. (1999). Emergence of scaling in random networks. *science*, *286*(5439), 509-512.
- Baumann, F., Lorenz-Spreen, P., Sokolov, I. M., & Starnini, M. (2020). Modeling echo chambers and polarization dynamics in social networks. *Physical review letters*, *124*(4), 048301.
- Blondel, V. D., Guillaume, J. L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, *2008*(10), P10008.
- Cano, N. R. (2018). Introducción a las redes complejas: El modelo del mundo pequeno. *Revista Entornos*, *31*(2), 60-64.
- Clauset, A., Newman, M. E., & Moore, C. (2004). Finding community structure in very large networks. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, *70*(6), 066111.
- de Blas, C. S., Guevara, J. A., Morillo, J., & Gómez González, D. (2022, July). Polarization measures in bi-partition networks based on fuzzy graphs. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems* (pp. 398-409). Cham: Springer International Publishing.
- Gregory, S. (2011). Fuzzy overlapping communities in networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2011*(02), P02017.
- Guevara, J. A. (2022). *Medición de la polarización ideológica en redes sociales mediante lógica difusa* (Tesis doctoral, Universidad Complutense de Madrid).
- Guevara, J. A., Gómez, D., Robles, J. M., & Montero, J. (2020, June). Measuring polarization: A fuzzy set theoretical approach. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems* (pp. 510-522). Cham: Springer International Publishing.
- Gutiérrez, I., Gómez, D., Castro, J., & Espínola, R. (2022). From fuzzy information to community detection: An approach to social networks analysis with soft information. *Mathematics*, *10*(22), 4348.

- Gutiérrez, I., Guevara, J. A., Gomez, D., Castro, J., & Espinola, R. (2021). Community detection problem based on polarization measures: an application to Twitter: the COVID-19 case in Spain. *Mathematics*, *9*(4), 443.
- L. C. Freeman. Centrality in social networks conceptual clarification. Social networks, 1(3):215–239, 1978.
- L. C. Freeman. Spheres, cubes and boxes: graph dimensionality and network structure. Social Networks, 5(2):139–156, 1983.
- L. C. Freeman, S. P. Borgatti, and D. R. White. Centrality in valued graphs: A measure of betweenness based on network flow. Social networks, 13(2):141–154, 1991.
- L. C. Freeman, D. Roeder, and R. R. Mulholland. Centrality in social networks: Ii. experimental results. Social networks, 2(2):119–141, 1979.
- Obregón-Quintana, B., Hernández-Pérez, R., & Guzmán-Vargas, L. (2012). Algunas propiedades de transporte en redes de mundo-pequeno y libres de escala. *Revista Mexicana de Física*, *58*(1), 69-75.
- Robles, J. M., Rodríguez, J. T., Caballero, R., & Gómez, D. (2020). *Big data para científicos sociales. Una introducción* (Vol. 60). CIS.
- Wang, L. X. (2016). Modeling stock price dynamics with fuzzy opinion networks. *IEEE Transactions on Fuzzy Systems*, *25*(2), 277-301.
- Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of'small-world'networks. *Nature*, *393*(6684), 409-410.
- Zadeh, L. (1965). Fuzzy sets. *Inform Control*, *8*, 338-353.
- Reuters. (2025, marzo 15). *OpenAI, Musk agree to fast-tracked trial over for-profit shift*.
- Wiggers, K. (2025, marzo 26). *OpenAI adopts rival Anthropic's standard for connecting AI models to data*. *TechCrunch*.
- Jackson, F. (2025, marzo 28). *OpenAI Agents Now Support Rival Anthropic's Protocol*. *TechRepublic*.
- Smith, T. (2025, marzo 28). *How the Model Context Protocol simplifies AI development*. *TechTarget*.
- [Autor anónimo]. (2025, marzo). *Model-context protocol: the next big step in generating value from AI*. *Engineering.com*.

9. ANEXOS

```
# Cargar librerías necesarias
library(rvest)       # Para leer archivos HTML
library(dplyr)       # Para manipulación de datos
library(igraph)      # Para análisis y creación de grafos
library(visNetwork)  # Para crear grafos interactivos
library(stringr)
library(tidyr)
library(data.table)  # Para combinaciones rápidas
library(ggplot2)     # Para visualización)
library(lubridate)
library(tidytext)
library(cld2)        # Para detección de idioma
```

```
library(scales)       # Para formatear porcentajes
library(textcat)

# Función para extraer la tabla 3 (tweets)
extract_table3 <- function(file_path) {
  html_data <- read_html(file_path, encoding = "UTF-8")
  tables <- html_data %>% html_table(fill = TRUE)
  table3 <- as.data.frame(tables[[3]])
  return(table3)
}

# Función para extraer la tabla 5 (menciones)
extract_table5 <- function(file_path) {
  html_data <- read_html(file_path, encoding = "UTF-8")
  tables <- html_data %>% html_table(fill = TRUE)
  table5 <- as.data.frame(tables[[5]])
  return(table5)
}

# ---- PROCESAR TABLA 3: TWEETS ----

# Unir datos de tweets de todos los archivos
tweets_data <- bind_rows(lapply(files, extract_table3))
# Filtrar IDs únicos (eliminando repetidos)
tweets_data <- tweets_data[!duplicated(tweets_data$ID), ]

# ---- PROCESAR TABLA 5: MENCIONES ----

# Unir datos de menciones de todos los archivos
mentions_data <- bind_rows(lapply(files, extract_table5))
# Fusionar menciones repetidas sumando pesos
mentions_data <- mentions_data %>%
  group_by(Source, Target) %>%
  summarise(Weight = sum(Weight, na.rm = TRUE), .groups = "drop")

# Crear un bucle que vaya de 1 a nrow(tweets_Data)
# si mentiones está vacío, next, sino:
# m <- unlist(str_split(tweets_data[tweets_data$Author=="fsorel","Mentions"], " "))

mentions_data <- data.frame(SOURCE = 0, TARGET = 0)
for (i in 1:nrow(tweets_data)) {
  if (tweets_data$Mentions[i] == "") {
    next
  } else {
    m <- unlist(str_split(tweets_data$Mentions[i], " "))
    for (j in 1:length(m)) {
      mentions_data <- rbind(mentions_data, data.frame(SOURCE = tweets_data$Author[i],
TARGET = m[j]))
    }
  }
}
# Eliminar conexiones donde SOURCE o TARGET son vacíos o "0"
mentions_data <- mentions_data %>%
  filter(SOURCE != "" & TARGET != "") %>%
  filter(SOURCE != "0" & TARGET != "0")
```

```
# ---- CREACIÓN DEL GRAFO ----

# Crear el grafo con todas las menciones
mention_graph <- graph_from_data_frame(mentions_data, directed = FALSE)

# Comparación de nodos en `mention_graph` y `mentions_data`
cat("Total de nodos en el grafo:", vcount(mention_graph), "\n")
unique_nodes <- unique(c(mentions_data$SOURCE, mentions_data$TARGET))
cat("Total de nodos únicos en mentions_data:", length(unique_nodes), "\n")

# ---- VISUALIZACIÓN DEL GRAFO ----
# Detectar comunidades con Louvain
communities <- cluster_louvain(mention_graph)

# Crear el dataframe de nodos con la información de comunidad
nodes <- data.frame(
  id = V(mention_graph)$name,   # ID del nodo (nombre del usuario)
  label = V(mention_graph)$name,  # Etiqueta del nodo (para visualización)
  group = membership(communities),  # Número de comunidad a la que pertenece el nodo
  value = degree(mention_graph)  # Tamaño del nodo basado en el grado (cantidad de conexiones)
)
# Crear el dataframe de aristas
edges <- data.frame(
  from = as.character(mentions_data$SOURCE),  # Usuario que menciona
  to = as.character(mentions_data$TARGET)  # Usuario mencionado
  #value = mentions_data$Weight  # Peso de la relación (frecuencia de menciones)
)

# Crear el grafo interactivo con visNetwork
visNetwork(nodes, edges) %>%
  visOptions(highlightNearest = TRUE, nodesIdSelection = TRUE) %>%
  visPhysics(stabilization = FALSE) %>%
  visInteraction(dragNodes = TRUE, zoomView = TRUE) %>%
  visLayout(randomSeed = 42)

# Lista de usuarios destacados
destacados <- c("openai", "deepseek_ai", "tut_ml", "betamoroney", "antgrasso", "ronald_vanloon")

# Actualizar nodos con etiquetas más visibles
nodes$label <- ifelse(nodes$id %in% destacados, nodes$id, NA)
nodes$title <- nodes$id  # tooltip al pasar el mouse
nodes$font.size <- ifelse(nodes$id %in% destacados, 30, 0)  # solo mostrar texto grande si está
en la lista
nodes$color <- ifelse(nodes$id %in% destacados, "red", NA)  # opcional para marcar color rojo

# Visualización
visNetwork(nodes, edges) %>%
  visOptions(highlightNearest = TRUE, nodesIdSelection = TRUE) %>%
  visPhysics(stabilization = FALSE) %>%
  visInteraction(dragNodes = TRUE, zoomView = TRUE) %>%
  visLayout(randomSeed = 42)

# Métricas básicas de la red global
cat("Número total de nodos:", vcount(mention_graph), "\n")
cat("Número total de aristas:", ecount(mention_graph), "\n")
```

```
cat("Número de componentes conexas:", components(mention_graph)$no, "\n")
cat("Tamaño de la componente gigante:", max(components(mention_graph)$csize), "\n")
cat("Densidad de la red:", graph.density(mention_graph), "\n")
cat("Diámetro de la red:", diameter(mention_graph, unconnected = TRUE), "\n")
cat("Distancia media:", mean_distance(mention_graph, directed = FALSE), "\n")
cat("Clustering global:", transitivity(mention_graph, type = "global"), "\n")

# Número de comunidades detectadas con Louvain
communities <- cluster_louvain(mention_graph)
cat("Número de comunidades detectadas (Louvain):", length(communities), "\n")

# ---- ANÁLISIS DE CENTRALIDAD ----

# Calcular centralidad de grado
degree_centrality <- degree(mention_graph, mode = "all")
top_degree <- sort(degree_centrality, decreasing = TRUE)[1:5]

# Calcular centralidad de intermediación (betweenness)
betweenness_centrality <- betweenness(mention_graph, directed = FALSE, normalized = TRUE)
top_betweenness <- sort(betweenness_centrality, decreasing = TRUE)[1:5]

# Calcular centralidad de cercanía (closeness)
closeness_centrality <- closeness(mention_graph, normalized = TRUE)
top_closeness <- sort(closeness_centrality, decreasing = TRUE)[1:5]

# Calcular centralidad de autovector (eigenvector)
eigen_centrality <- eigen_centrality(mention_graph)$vector
top_eigen <- sort(eigen_centrality, decreasing = TRUE)[1:5]

# Mostrar los top 5 usuarios en cada métrica de centralidad
cat("Top 5 usuarios por Grado (Degree Centrality):")
print(top_degree)

cat("Top 5 usuarios por Intermediación (Betweenness Centrality):")
print(top_betweenness)

cat("Top 5 usuarios por Cercanía (Closeness Centrality):")
print(top_closeness)

cat("Top 5 usuarios por Autovector (Eigenvector Centrality):")
print(top_eigen)

# DETECCIÓN DE COMUNIDADES (forzar 2 comunidades)
# Aplicar Louvain para detección de comunidades
louvain_comm <- cluster_louvain(mention_graph)

# Mantener solo la componente gigante
componentes <- components(mention_graph)
componente_gigante <- which.max(componentes$csize)

# Filtrar solo nodos de la componente más grande
mention_graph_gc <- induced_subgraph(mention_graph,
                                     vids = V(mention_graph)[componentes$membership ==
componente_gigante])

# Simplificar la red (sin multi-aristas ni loops)
```

```
mention_graph_gc <- simplify(mention_graph_gc, remove.multiple = TRUE, remove.loops = TRUE)

# Forzar 2 comunidades con fast greedy
fg_comm <- cluster_fast_greedy(mention_graph_gc)
forced_membership <- cut_at(fg_comm, no = 2)

# Asignar comunidad al grafo
V(mention_graph_gc)$forced_comm <- as.character(forced_membership)

modularity(mention_graph_gc,forced_membership) # modularidad de la partición restringida a 2 comunidades
modularity(mention_graph_gc,fg_comm$membership) # modularidad de la partición que Fast-Greedy encontró sin restricciones

# Lista de usuarios influyentes a destacar en la visualización
nodos_destacados <- c(
  "openai", "deepseek_ai", "elonmusk", "sama", "chatgptapp",
  "aureliecoudouel", "ronald_vanloon", "giga_labs", "nbdpress",
  "huzefaaziz4", "mtechitian", "rahulsutariya"
)

# Preparar nodos
nodes_gc <- data.frame(
  id = V(mention_graph_gc)$name,
  label = ifelse(V(mention_graph_gc)$name %in% nodos_destacados,
                 V(mention_graph_gc)$name, ""),
  value = ifelse(V(mention_graph_gc)$name %in% nodos_destacados,
                 degree(mention_graph_gc) + 15,
                 degree(mention_graph_gc)),
  color = ifelse(V(mention_graph_gc)$forced_comm == "1", "red", "blue"),
  font.size = ifelse(V(mention_graph_gc)$name %in% nodos_destacados, 180, 0),
  font.face = ifelse(V(mention_graph_gc)$name %in% nodos_destacados, "bold", "arial")
)
# Preparar aristas
edges_gc <- as_data_frame(mention_graph_gc, what = "edges")
colnames(edges_gc) <- c("from", "to")

# Visualización
visNetwork(nodes_gc, edges_gc) %>%
  visOptions(highlightNearest = TRUE, nodesIdSelection = TRUE) %>%
  visPhysics(stabilization = TRUE) %>%
  visLayout(randomSeed = 42) %>%
  visInteraction(dragNodes = TRUE, zoomView = TRUE)

# ---- ANÁLISIS DE CENTRALIDAD ----
# 1) Extraer los vértices de cada comunidad
ids_A <- which(V(mention_graph_gc)$forced_comm == "1")  # comunidad 1 (rojo)
ids_B <- which(V(mention_graph_gc)$forced_comm == "2")  # comunidad 2 (azul)

# 2) Construir los subgrafos inducidos
g_A <- induced_subgraph(mention_graph_gc, vids = ids_A)
g_B <- induced_subgraph(mention_graph_gc, vids = ids_B)

# 3) Calcular centralidades para cada subgrafo
#    (a) Comunidad A
```

```
deg_A  <- degree(g_A,   mode = "all")
clo_A  <- closeness(g_A, normalized = TRUE)
bet_A  <- betweenness(g_A, directed = FALSE, normalized = TRUE)
eig_A  <- eigen_centrality(g_A)$vector

#    (b) Comunidad B
deg_B  <- degree(g_B,   mode = "all")
clo_B  <- closeness(g_B, normalized = TRUE)
bet_B  <- betweenness(g_B, directed = FALSE, normalized = TRUE)
eig_B  <- eigen_centrality(g_B)$vector

# —— Comunidad B (azul) ——
top5_deg_B  <- sort(deg_B,  decreasing = TRUE)[1:5]
top5_clo_B  <- sort(clo_B,  decreasing = TRUE)[1:5]
top5_bet_B  <- sort(bet_B,  decreasing = TRUE)[1:5]
top5_eig_B  <- sort(eig_B,  decreasing = TRUE)[1:5]

# ANÁLISIS DE HASHTAGS

# Obtener las comunidades forzadas
node_info_gc <- data.frame(
  Author = V(mention_graph_gc)$name,
  Community = as.character(V(mention_graph_gc)$forced_comm)
)

# Unir esa info con los tweets
tweets_data_gc <- tweets_data %>%
  left_join(node_info_gc, by = "Author") %>%
  filter(!is.na(Community))  # Eliminar autores que no están en el grafo

# Separar tweets por comunidad
tweets_rojos_gc <- tweets_data_gc %>% filter(Community == "1")
tweets_azules_gc <- tweets_data_gc %>% filter(Community == "2")

# Función para extraer hashtags
extract_hashtags <- function(text) {
  unlist(str_extract_all(text, "#\\S+"))
}

# Aplicar función a los textos
tweets_rojos_gc$hashtags <- lapply(tweets_rojos_gc$Text, extract_hashtags)
tweets_azules_gc$hashtags <- lapply(tweets_azules_gc$Text, extract_hashtags)

# Convertir a vectores y limpiar
hashtags_rojos_gc <- tolower(unlist(tweets_rojos_gc$hashtags))
hashtags_azules_gc <- tolower(unlist(tweets_azules_gc$hashtags))

hashtags_rojos_gc <- hashtags_rojos_gc[hashtags_rojos_gc != ""]
hashtags_azules_gc <- hashtags_azules_gc[hashtags_azules_gc != ""]

# Contar frecuencia
hashtags_freq_rojos_gc <- sort(table(hashtags_rojos_gc), decreasing = TRUE)
hashtags_freq_azules_gc <- sort(table(hashtags_azules_gc), decreasing = TRUE)

cat("Top 20 hashtags en Comunidad 1 (Roja):\n")
print(head(hashtags_freq_rojos_gc, 20))
```

```
cat("\nTop 20 hashtags en Comunidad 2 (Azul):\n")
print(head(hashtags_freq_azules_gc, 20))

# DESCRIPTIVOS QUE DIFERENCIEN ENTRE ROJOS Y AZULES
únicos_roja <- setdiff(names(hashtags_freq_rojos_gc), names(hashtags_freq_azules_gc))
únicos_azul <- setdiff(names(hashtags_freq_azules_gc), names(hashtags_freq_rojos_gc))

print(head(únicos_roja, 10))
print(head(únicos_azul, 10))

table(nodes_gc$color)  # Tamaño por comunidad
aggregate(value ~ color, data = nodes_gc, mean)  # Promedio de conexiones por comunidad

tweets_data_gc %>%
  group_by(Community) %>%
  summarise(
    media_tweets_autor = n() / n_distinct(Author)
  )

table(grepl("news|daily|press", tweets_data_gc$Author), tweets_data_gc$Community)

table(grepl("ai|tech|robot|ml|data", tweets_data_gc$Author), tweets_data_gc$Community)


# DETECCIÓN Y ANÁLISIS DE IDIOMAS EN TWEETS USANDO CLD2
# 1. Limpiar texto para mejorar detección

tweets_data_gc$clean_text <- tweets_data_gc$Text %>%
  str_remove_all("http\\S+") %>%              # Enlaces
  str_remove_all("#\\S+") %>%                 # Hashtags
  str_remove_all("@\\S+") %>%                 # Menciones
  str_remove_all("[\n\r]") %>%                # Saltos de línea
  str_remove_all("\\b[A-Za-z]\\b") %>%        # Palabras de una sola letra
  str_remove_all("[0-9]") %>%                 # Números
  str_remove_all("[[:punct:]]") %>%           # Signos de puntuación
  str_squish()                                # Espacios extra


# 2. Detectar idioma con texto limpio (cld2)
tweets_data_gc$idioma <- detect_language(tweets_data_gc$clean_text)

# 3. Ver tabla general de idiomas por comunidad
table(tweets_data_gc$idioma, tweets_data_gc$Community)

# 4. Ver ejemplos de NA tras detección
# Detectar idioma solo en NA usando textcat como alternativa
tweets_data_gc$idioma[textcat(tweets_data_gc$clean_text) %in% c("english", "spanish", "french",
"chinese")] -> fallback
tweets_data_gc$idioma[is.na(tweets_data_gc$idioma)] <- fallback[is.na(tweets_data_gc$idioma)]

sum(is.na(tweets_data_gc$idioma))

tweets_data_gc %>%
  filter(is.na(idioma)) %>%
  select(Text) %>%
```

```
  head(10)

# 5. Cuántos idiomas distintos por comunidad
tweets_data_gc %>%
  group_by(Community) %>%
  summarise(idiomas_distintos = n_distinct(idioma))

# 6. Calcular proporción de idiomas por comunidad
frecuencias_idioma <- tweets_data_gc %>%
  count(Community, idioma) %>%
  group_by(Community) %>%
  mutate(proporcion = n / sum(n)) %>%
  arrange(Community, desc(proporcion))

# 7. Gráfico: proporciones de idiomas (>1%)
frecuencias_idioma %>%
  filter(proporcion > 0.01) %>%  # Mostrar solo idiomas con más del 1%
  ggplot(aes(x = reorder(idioma, proporcion), y = proporcion, fill = Community)) +
  geom_col(position = "dodge") +
  coord_flip() +
  labs(title = "Proporción de idiomas por comunidad",
       x = "Idioma",
       y = "Proporción",
       fill = "Comunidad") +
  scale_y_continuous(labels = percent_format()) +
  theme_minimal()


# Proporción de tweets con palabras clave en francés
tweets_data_gc %>%
  mutate(frances = grepl("\\b(le|la|des|pour|une|avec)\\b", Text)) %>%
  group_by(Community) %>%
  summarise(fr_proporcion = mean(frances))

#####
# ANÁLISIS DE POLARIZACIÓN
# Elegir nodo específico
nodo_objetivo <- "googlesuperbug"
# Obtener vecinos (incluirse a sí mismo)
vecinos_idx <- unlist(neighborhood(mention_graph_gc, order =
1)[[which(V(mention_graph_gc)$name == nodo_objetivo)]])
# Obtener nombres de vecinos
vecinos_nombres <- V(mention_graph_gc)[vecinos_idx]$name
# Obtener colores de esos vecinos
colores_vecinos <- nodes_gc$color[match(vecinos_nombres, nodes_gc$id)]
# Verificar cantidades
table(colores_vecinos)
# Calcular proporciones reales
mu_a <- sum(colores_vecinos == "red", na.rm = TRUE) / length(colores_vecinos)
mu_b <- 1 - mu_a
# Mostrar
sprintf("mu_a (rojo): %.10f", mu_a)
sprintf("mu_b (azul): %.10f", mu_b)

# se necesita un data frame con 3 columnas: ID | A | B
# donde A es el grado depertenencia a la comunidad roja, y B a la azul.
```

```
# Crear un dataframe vacío
polarizacion_df <- data.frame(
  ID = character(),
  A = numeric(),  # Proporción de vecinos rojos
  B = numeric(),  # Proporción de vecinos azules
  stringsAsFactors = FALSE
)

# Calcular proporciones para cada nodo
for (i in 1:vcount(mention_graph_gc)) {
  nodo <- V(mention_graph_gc)[i]$name
  vecinos <- unlist(neighborhood(mention_graph_gc, order = 1)[[i]])  # incluye al nodo
  nombres_vecinos <- V(mention_graph_gc)[vecinos]$name
  comunidades_vecinos <- nodes_gc$color[match(nombres_vecinos, nodes_gc$id)]

  mu_a <- sum(comunidades_vecinos == "red", na.rm = TRUE) / length(comunidades_vecinos)
  mu_b <- 1 - mu_a

  polarizacion_df <- rbind(polarizacion_df, data.frame(ID = nodo, A = mu_a, B = mu_b))
}

# Histograma de grado de pertenencia a la comunidad A (roja)
ggplot(polarizacion_df, aes(x = A)) +
  geom_histogram(binwidth = 0.05, fill = "red", color = "white") +
  labs(title = "Distribución de grado de pertenencia a la comunidad A (roja)",
       x = "Grado de pertenencia al Polo A",
       y = "Frecuencia") +
  theme_minimal()

# Histograma de grado de pertenencia a la comunidad B (azul)
ggplot(polarizacion_df, aes(x = B)) +
  geom_histogram(binwidth = 0.05, fill = "blue", color = "white") +
  labs(title = "Distribución de grado de pertenencia a la comunidad B (azul)",
       x = "Grado de pertenencia al Polo B",
       y = "Frecuencia") +
  theme_minimal()

# CÁLCULO CON JDJ
JDJ <- function(data, r = 4) {
  # data = data frame con ID, A y B.
  # r = redondeo de decimales de mu_a y mu_b
  t <- proc.time()
  data[, 2:3] <- round(data[, 2:3], r)
  # Tabla de frecuencias relativas de combinaciones (mu_a, mu_b)
  a <- data.frame(prop.table(table(data[, 2:3])))
  a <- a[a$Freq != 0, ]
  a$A <- as.numeric(as.character(a$A))
  a$B <- as.numeric(as.character(a$B))
  # Convertir a matriz para operaciones vectorizadas
  a <- as.matrix(a)
  # Matrices cruzadas: muA_i * muB_j y muB_i * muA_j
  AB <- outer(a[, "A"], a[, "B"])
  BA <- outer(a[, "B"], a[, "A"])

  # Tomamos el máximo entre los dos para cada par (i, j)
```

```
  JDJ_Full <- pmax(AB, BA) * outer(a[, "Freq"], a[, "Freq"])

  # Calcular valor final de JDJ
  JDJ_value <- sum(JDJ_Full) * 2
  #return(cat(" JDJ =", JDJ_value,
  #           "\nTiempo de computación:", (proc.time() - t)[[3]]))
  return(JDJ_value)
}

# Llamar a la función con el dataframe de polarización
JDJ(polarizacion_df, r = 4)

# ANÁLISIS SEMANAL DE GRAFOS

# --- 1) Preparación de los datos temporales ----------

# 1.1 Convertir el timestamp a Date
tweets_data <- tweets_data %>%
  mutate(
    # Timestamp originalmente "YYYY-MM-DD HH:MM:SS"
    fecha = as.Date(Timestamp, format = "%Y-%m-%d %H:%M:%S")
  )

# 1.2 Definir los puntos de corte semanales
inicio <- min(tweets_data$fecha)          # p.ej. "2025-01-19"
fin    <- max(tweets_data$fecha)          # p.ej. "2025-04-02"
cortes <- seq(inicio, fin, by = "7 days") # 19-ene, 26-ene, 02-feb, …, 30-mar
cortes <- c(cortes, fin + 1)              # Añadir "2025-04-03" para cerrar último intervalo

# 1.3 Etiquetar cada tweet con su “periodo” de inicio de semana
tweets_data <- tweets_data %>%
  mutate(
    periodo = cut(
      fecha,
      breaks = cortes,
      right  = FALSE,                 # cada bin es [inicio, siguiente_inicio)
      labels = cortes[-length(cortes)] # etiquetas = fechas de inicio
    )
  )

# 1.4 Transformar el factor de periodos en número de semana (1, 2, …, 12)
tweets_data <- tweets_data %>%
  mutate(
    semana_num   = as.integer(periodo),
    semana_label = paste0("Semana ", semana_num)
  )

# Comprobación
table(tweets_data$semana_num)

# --- 2) Construcción de mentions_data_sem por semana ------------------------

# Creamos un data.frame vacío que contendrá las aristas (menciones)
mentions_data_sem <- data.frame(
  SOURCE     = character(),
  TARGET     = character(),
```

```
  semana_num = integer(),
  stringsAsFactors = FALSE
)

# Iteramos tweet a tweet para desanidar las menciones
for (i in seq_len(nrow(tweets_data))) {
  mencs <- tweets_data$Mentions[i]
  if (is.na(mencs) || mencs == "") next

  # Separar el string de menciones en vector de usuarios
  usuarios <- unlist(str_split(mencs, "\\s+"))
  usuarios <- usuarios[usuarios != ""]  # eliminar cadenas vacías

  # Añadir una fila por cada mención
  for (u in usuarios) {
    mentions_data_sem <- rbind(
      mentions_data_sem,
      data.frame(
        SOURCE     = tweets_data$Author[i],
        TARGET     = u,
        semana_num = tweets_data$semana_num[i],
        stringsAsFactors = FALSE
      )
    )
  }
}

max_sem <- max(mentions_data_sem$semana_num)

# Guardar todos los grafos completos en una lista, para no recalcarlos

lista_grafos_full <- vector("list", length = max_sem)

for (w in 1:max_sem) {
  menc_w <- mentions_data_sem %>% filter(semana_num <= w)
  todos_nodos <- unique(c(menc_w$SOURCE, menc_w$TARGET))
  g_full <- graph_from_data_frame(
    d        = menc_w %>% select(SOURCE, TARGET),
    directed = FALSE,
    vertices = data.frame(name = todos_nodos, stringsAsFactors = FALSE)
  )
  lista_grafos_full[[w]] <- g_full
}

#
# Grafos Semanales COMPLETOS (incluye satélites) — Visualización
#
# Lista para almacenar cada grafo coloreado por comunidades
redes_completas <- vector("list", length = max_sem)
# Vector para guardar cuántas comunidades naturales hay en cada semana
num_comunidades <- integer(max_sem)

for (w in 1:max_sem) {
  g_full <- lista_grafos_full[[w]]

  # Detectar comunidades “naturales” (Louvain) solo para colorear
```

```
  comm_w <- cluster_louvain(g_full)
  memb_w <- membership(comm_w)
  num_comunidades[w] <- length(comm_w)

  pal <- rainbow(num_comunidades[w])
  nodes_df <- data.frame(
    id     = V(g_full)$name,
    label  = V(g_full)$name,
    group  = as.factor(memb_w),
    color  = pal[memb_w],
    value  = degree(g_full),
    title  = paste0("<p><b>", V(g_full)$name, "</b><br>",
                    "Comunidad: ", memb_w, "<br>",
                    "Grado: ", degree(g_full), "</p>"),
    stringsAsFactors = FALSE
  )
  edges_df <- as_data_frame(g_full, what = "edges")

  redes_completas[[w]] <- visNetwork(nodes_df, edges_df,
                                main = paste0("Grafo COMPLETO Semana 1–", w,
                                              " (nComm=", num_comunidades[w], ")")) %>%
    visOptions(highlightNearest = TRUE, nodesIdSelection = TRUE) %>%
    visPhysics(stabilization = TRUE) %>%
    visLayout(randomSeed = 42)
}

df_com_nat <- tibble(
  semana      = 1:max_sem,
  comunidades = num_comunidades
)

print(df_com_nat)


# Modularidad natural y forzada en 2 comunidades — g_gc
# Pero número de comunidades reales sobre g_full

# Inicializar contenedores
num_comunidades     <- integer(max_sem)
modularidades_nat   <- numeric(max_sem)
modularidades_forz2 <- numeric(max_sem)

for (w in 1:max_sem) {
  g_full <- lista_grafos_full[[w]]

  # ——— Número de comunidades NATURALES sobre el grafo completo ———
  comm_full <- cluster_louvain(g_full)
  num_comunidades[w] <- length(comm_full)  # número total de comunidades (satélites incluidos)

  # ——— Modularidades sobre la componente gigante ———
  comp   <- components(g_full)
  idx_gc <- which(comp$membership == which.max(comp$csize))
  g_gc   <- induced_subgraph(g_full, vids = idx_gc) %>%
    simplify(remove.multiple = TRUE, remove.loops = TRUE)

  # Natural (sin restricción)
```

```
  fg_comm <- cluster_fast_greedy(g_gc)
  memb_nat <- membership(fg_comm)
  modularidades_nat[w] <- modularity(g_gc, memb_nat)

  # Forzada en 2 comunidades
  memb2 <- cut_at(fg_comm, no = 2)
  modularidades_forz2[w] <- modularity(g_gc, memb2)
}

# Data frame final
df_com_nat <- tibble(
  semana              = 1:max_sem,
  comunidades         = num_comunidades,       # ← basado en grafo completo
  modularidad_natural  = modularidades_nat,    # ← sobre g_gc
  modularidad_2        = modularidades_forz2   # ← sobre g_gc
)

print(df_com_nat)

#
# Grafos Semanales forzadas en 2 comunidades — Visualización y Métricas
#
# Componente Gigante (sin satélites) + Métricas e Histogramas

metrics_puro <- data.frame(
  semana        = integer(),
  n_nodos       = integer(),
  n_aristas     = integer(),
  densidad      = numeric(),
  diametro      = integer(),
  clust_global  = numeric(),
  dist_media    = numeric(),
  C_rand        = numeric(),    # clustering de grafo aleatorio
  L_rand        = numeric(),    # distancia media de grafo aleatorio
  sigma_sw      = numeric(),    # pequeño-mundo
  mean_closeness   = numeric(),
  mean_clust_local = numeric(),
  mean_min_dist    = numeric(),
  stringsAsFactors = FALSE
)

for (w in 1:max_sem) {
  cat("\n--- Semana 1–", w, " (Componente Gigante) ---\n", sep = "")

  # 1) Grafo acumulado hasta w y extraer componente gigante
  g_full <- lista_grafos_full[[w]]
  comp   <- components(g_full)
  idx_gc <- which(comp$membership == which.max(comp$csize))
  g_gc   <- induced_subgraph(g_full, vids = idx_gc) %>%
    simplify(remove.multiple = TRUE, remove.loops = TRUE)

  # 2) Métricas observadas
  nn   <- vcount(g_gc)
  ea   <- ecount(g_gc)
  den  <- graph.density(g_gc)
  dia  <- diameter(g_gc, directed = FALSE, unconnected = TRUE)
```

```
Cobs <- transitivity(g_gc, type = "global")
Lobs <- mean_distance(g_gc, directed = FALSE)

# 3) Grafo aleatorio de referencia y métricas
g_rand <- sample_gnm(nn, ea, directed = FALSE, loops = FALSE)
Crand   <- transitivity(g_rand, type = "global")
Lrand   <- mean_distance(g_rand, directed = FALSE)
sigma   <- (Cobs / Crand) / (Lobs / Lrand)

# 4) Estadísticas de distribución
clos_vals <- closeness(g_gc)
clust_loc <- transitivity(g_gc, type = "local", isolates = "zero")
dists_vec <- distances(g_gc)
dists_vec <- dists_vec[is.finite(dists_vec) & dists_vec > 0]

mean_clo    <- mean(clos_vals, na.rm = TRUE)
mean_clustl <- mean(clust_loc, na.rm = TRUE)
mean_dd     <- mean(dists_vec, na.rm = TRUE)

# 5) Degree distribution (para histograma log-log)
degs <- degree(g_gc)

# 6) Guardar en data.frame
metrics_puro <- bind_rows(metrics_puro,
                      tibble(
                        semana          = w,
                        n_nodos         = nn,
                        n_aristas       = ea,
                        densidad        = den,
                        diametro        = dia,
                        clust_global    = Cobs,
                        dist_media      = Lobs,
                        C_rand          = Crand,
                        L_rand          = Lrand,
                        sigma_sw        = sigma,
                        mean_closeness  = mean_clo,
                        mean_clust_local = mean_clustl,
                        mean_min_dist   = mean_dd
                      )
)

# 7) Dibujar histogramas en 2 filas x 2 columnas
par(mfrow = c(2, 2), mar = c(4, 4, 2, 1))

# 7a) Closeness
hist(clos_vals,
    main = paste("Closeness Semana", w),
    xlab = "Closeness",
    col  = "lightgray", border = "white")
# 7b) Clustering local
hist(clust_loc,
    main = paste("Clustering local Semana", w),
    xlab = "Coef. local",
    col  = "lightgray", border = "white")
# 7c) Distancias mínimas
hist(dists_vec,
```

```
      main = paste("Dist. mínimas Semana", w),
      xlab = "Longitud mínima",
      col  = "lightgray", border = "white")
  # 7d) Grado en escala log–log
  hist(log10(degs),
      breaks = 30,
      main = paste("Log10(Grado) Semana", w),
      xlab = "log10(Grado)",
      col  = "lightgray", border = "white")

  # 8) Pausa
  readline("Pulsa <Enter> para la siguiente semana...")
}

# Restaurar dispositivo gráfico
par(mfrow = c(1, 1))

# Imprimir tabla final
print(metrics_puro)

#
# Forzar 2 Comunidades + Métricas + Polarización + Visualización
#   A = OpenAI (rojo), B = DeepSeek (azul)

# Contenedores
comunidades_forzadas   <- data.frame(
  semana                = integer(),
  N_nodos_OpenAI        = integer(),
  N_aristas_OpenAI      = integer(),
  Densidad_OpenAI       = numeric(),
  Diametro_OpenAI       = integer(),
  Top3_Comunidad_A      = character(),
  N_nodos_DeepSeek      = integer(),
  N_aristas_DeepSeek    = integer(),
  Densidad_DeepSeek     = numeric(),
  Diametro_DeepSeek     = integer(),
  Top3_Comunidad_B      = character(),
  stringsAsFactors      = FALSE
)
polarizacion_semanal    <- data.frame(
  semana = integer(),
  JDJ    = numeric(),
  stringsAsFactors = FALSE
)
redes_forzadas         <- vector("list", length = max_sem)

for (w in 1:max_sem) {
  # ——————————— Obtén la componente gigante g_gc ———————————
  g_full <- lista_grafos_full[[w]]
  comp   <- components(g_full)
  idx_gc <- which(comp$membership == which.max(comp$csize))
  g_gc   <- induced_subgraph(g_full, vids = idx_gc) %>%
    simplify(remove.multiple = TRUE, remove.loops = TRUE)

  # ——————————— Forzar 2 comunidades ———————————
  fg    <- cluster_fast_greedy(g_gc)
```

```
memb2 <- cut_at(fg, no = 2)
V(g_gc)$group <- as.character(memb2)

# ——————————— Construye el df de nodos coloreado ———————————

nodes_df <- data.frame(
  id    = V(g_gc)$name,
  color = ifelse(V(g_gc)$group=="1", "red", "blue"),
  label = ifelse(V(g_gc)$name %in% nodos_destacados, V(g_gc)$name, ""),
  value = ifelse(V(g_gc)$name %in% nodos_destacados, degree(g_gc) + 15, degree(g_gc)),
  font.size = ifelse(V(g_gc)$name %in% nodos_destacados, 150, 0),
  font.face = ifelse(V(g_gc)$name %in% nodos_destacados, "bold", "arial"),
  stringsAsFactors = FALSE
)

# ——————————— Extrae subgrafos A y B y sus métricas ———————————
ids_A <- which(memb2==1);  g_A <- induced_subgraph(g_gc, vids = ids_A)
ids_B <- which(memb2==2);  g_B <- induced_subgraph(g_gc, vids = ids_B)

# Métricas A
nA    <- vcount(g_A);  eA    <- ecount(g_A)
densA <- if(nA>1) graph.density(g_A) else 0
diaA  <- if(nA>1) diameter(g_A, directed=FALSE, unconnected=TRUE) else 0
degA  <- degree(g_A)
top3A <- if(nA>0) paste(names(sort(degA,TRUE))[1:min(3,length(degA))], collapse=", ") else ""

# Métricas B
nB    <- vcount(g_B);  eB    <- ecount(g_B)
densB <- if(nB>1) graph.density(g_B) else 0
diaB  <- if(nB>1) diameter(g_B, directed=FALSE, unconnected=TRUE) else 0
degB  <- degree(g_B)
top3B <- if(nB>0) paste(names(sort(degB,TRUE))[1:min(3,length(degB))], collapse=", ") else ""

comunidades_forzadas <- bind_rows(comunidades_forzadas,
                          tibble(
                            semana            = w,
                            N_nodos_OpenAI    = nA,
                            N_aristas_OpenAI  = eA,
                            Densidad_OpenAI   = densA,
                            Diametro_OpenAI   = diaA,
                            Top3_Comunidad_A  = top3A,
                            N_nodos_DeepSeek  = nB,
                            N_aristas_DeepSeek = eB,
                            Densidad_DeepSeek  = densB,
                            Diametro_DeepSeek  = diaB,
                            Top3_Comunidad_B  = top3B
                          )
)

# ——————————— Calcular polarización JDJ ———————————
polar_df <- data.frame(ID=character(), A=numeric(), B=numeric(), stringsAsFactors=FALSE)
for (i in seq_len(vcount(g_gc))) {
  vecinos_idx    <- unlist(neighborhood(g_gc,1)[[i]])
  vecinos_nombres <- V(g_gc)[vecinos_idx]$name
  colores_vec    <- nodes_df$color[ match(vecinos_nombres, nodes_df$id) ]
  mu_a <- sum(colores_vec=="red", na.rm=TRUE) / length(colores_vec)
```

```
    mu_b <- 1 - mu_a
    polar_df <- rbind(polar_df,
                      data.frame(ID=V(g_gc)$name[i], A=mu_a, B=mu_b, stringsAsFactors=FALSE)
    )
  }
  jdj_val <- JDJ(polar_df, r=4)
  polarizacion_semanal <- bind_rows(polarizacion_semanal,
                                    tibble(semana=w, JDJ=jdj_val)
  )

  # ——————————— Visualizar con visNetwork ———————————

  edges_df <- as_data_frame(g_gc, what = "edges") %>% rename(from = from, to = to)
  redes_forzadas[[w]] <- visNetwork(
  data.frame(
    id = nodes_df$id,
    label = nodes_df$label,
    color = nodes_df$color,
    value = nodes_df$value,
    font.size = nodes_df$font.size,
    font.face = nodes_df$font.face
  ),
  edges_df,
  main = paste0("Forzadas 2 Comunidades Sem 1–", w)
) %>%
  visOptions(highlightNearest = TRUE, nodesIdSelection = TRUE) %>%
  visPhysics(stabilization = TRUE) %>%
  visLayout(randomSeed = 42)
}
# Mostrar resultados
print(comunidades_forzadas)
print(polarizacion_semanal)

# Extraer y verificar “polarizacion_df” para comprobación

# 1) Elegimos w = semana : del 1 al 11
w <- 1

# 2) Reconstruimos g_full para la semana que queremos analizar
g_full   <- lista_grafos_full[[w]]

# 3) Extraer la componente gigante (sin satélites)
comp     <- components(g_full)
idx_gc   <- which(comp$membership == which.max(comp$csize))
g_gc     <- induced_subgraph(g_full, vids = idx_gc) %>%
  simplify(remove.multiple = TRUE, remove.loops = TRUE)

# 4) Forzar 2 comunidades (Fast-Greedy + cut_at(..., 2))
fg       <- cluster_fast_greedy(g_gc)
memb2    <- cut_at(fg, no = 2)
V(g_gc)$group <- as.character(memb2)    # "1" o "2"

# 5) Construimos nodes_gc_sem con los colores “red” / “blue”
nodes_gc_sem <- data.frame(
  id    = V(g_gc)$name,
  color = ifelse(V(g_gc)$group == "1", "red", "blue"),
```

```
  stringsAsFactors = FALSE
)

# 6) Ahora creamos polarizacion_df_sem recorriendo cada vértice de g_gc
polarizacion_df_sem <- data.frame(
  ID = character(),
  A  = numeric(),   # proporción vecinos rojos
  B  = numeric(),   # proporción vecinos azules
  stringsAsFactors = FALSE
)

for (i in seq_len(vcount(g_gc))) {
  nodo           <- V(g_gc)$name[i]
  vecinos_idx    <- unlist(neighborhood(g_gc, order = 1)[[i]])
  vecinos_nombres <- V(g_gc)[vecinos_idx]$name

  # Extraemos colores de esos vecinos desde nodes_gc_sem1
  colores_vecinos <- nodes_gc_sem$color[
    match(vecinos_nombres, nodes_gc_sem$id)
  ]

  # Calculamos proporciones
  mu_a <- sum(colores_vecinos == "red", na.rm = TRUE) / length(colores_vecinos)
  mu_b <- 1 - mu_a

  polarizacion_df_sem <- rbind(
    polarizacion_df_sem,
    data.frame(ID = nodo, A = mu_a, B = mu_b, stringsAsFactors = FALSE)
  )
}

# 7) Mostramos el resultado y comprobamos a ojo unos nodos
print(polarizacion_df_sem)

# 8) Histograma de polarización para la semana seleccionada
ggplot(polarizacion_df_sem, aes(x = A)) +
  geom_histogram(binwidth = 0.05, fill = "red", color = "white") +
  labs(
    title = paste0("Distribución de grado de pertenencia a la comunidad A (roja) – Semana ", w),
    x = "Grado de pertenencia al Polo A",
    y = "Frecuencia"
  ) +
  theme_minimal()
# 9) Histograma de polarización para la semana seleccionada
ggplot(polarizacion_df_sem, aes(x = B)) +
  geom_histogram(binwidth = 0.05, fill = "blue", color = "white") +
  labs(
    title = paste0("Distribución de grado de pertenencia a la comunidad B (azul) – Semana ", w),
    x = "Grado de pertenencia al Polo B",
    y = "Frecuencia"
  ) +
  theme_minimal()

JDJ(polarizacion_df_sem, r = 4)
```