UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Análisis de redes sociales en Twitch

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Herrera Fernández, Francesc

Tutor/a: Rebollo Pedruelo, Miguel

CURSO ACADÉMICO: 2021/2022

Resumen

En torno a las retransmisiones de Twitch, se generan comunidades de espectadores auspiciadas por los diferentes mecanismos de interacción que dan vida a la plataforma. Empleando una de estas herramientas, el chat que acompaña a cada emisión en vivo, podemos explorar las similitudes entre los 200 *streamers* con más seguidores en la comunidad hispanohablante. Actualmente, existe escasa literatura científica que explore Twitch desde la perspectiva del análisis de redes. Mediante la propuesta de nuevas metodologías, este trabajo plantea un paso hacía delante en el estudio de la plataforma y de las nuevas formas que de producir y consumir contenido en vivo que representa, obteniendo una representación novedosa que permite capturar las similitudes entre *streamers* tanto a nivel de contenido como de otros factores culturales.

Palabras clave: Twitch, análisis de redes, detección de comunidades, clusterización.

Abstract

Communities of viewers emerge around Twitch broadcasts thanks to the different interaction mechanisms that give life to the platform. Using one of these tools, the chat that accompanies all live broadcasts, we can explore the similarities between 200 streamers with the most followers in the Spanish-speaking community. Currently, there is little scientific literature exploring Twitch from the perspective of network analysis. By proposing new methodologies, this work takes a step forward in the study of the platform and the new ways of producing and consuming live content that it represents, obtaining a new representation that allows us to capture the similarities between streamers at different levels.

Keywords: Twitch, network analysis, community detection, clustering

Tabla de contenidos

Tabla de figuras

1. Introducción

Twitch es uno de los grandes nombres en el mercado actual de contenido audiovisual en *streaming*, compitiendo de tú a tú con las grandes empresas del entretenimiento tradicionales. Esta plataforma de emisión de vídeo en vivo reúne diariamente a más de 31 millones de usuarios de todo el mundo, según cifras de la propia compañía (Twitch Interactive, Inc., 2022).

Este nuevo paradigma de producción y consumo de contenidos en directo propicia escenarios inéditos, de necesaria exploración y discusión. Es relevante no solo en cuanto a una posible migración de la audiencia más joven de la televisión tradicional a Twitch, sino que, de manera profunda y estructural, propone una nueva forma en la que consumir contenidos audiovisuales en vivo, poniendo especial importancia en los elementos sociales que dan vida a la plataforma (Gutiérrez Lozano, 2020).

Dentro del ecosistema de plataformas de emisión de vídeo en directo, o *streaming,* distinguiremos dos figuras principales: la persona emisora o *streamer* y aquella que consume dicho contenido, es decir, el o la espectadora. Estas emisiones, se denominan directos o *streams*. Consumir contenido en Twitch se caracteriza por el factor social y comunitario de la plataforma, siendo el chat integrado en cada *stream* uno de los elementos más importantes de Twitch (James Dux, 2018). La posibilidad de interactuar con la comunidad que surge en torno a un *streamer* es uno de los principales factores que motivan a un usuario conectarse a un directo (William A. Hamilton, 2014).

1.1 Motivación

La escueta bibliografía científica que estudia Twitch desde la perspectiva del análisis de redes choca con el rápido crecimiento y adopción de la plataforma. El presente trabajo nace de la voluntad de aportar nuevos puntos de vista a la discusión técnica y científica sobre esta cuestión. Mediante la propuesta de nuevas metodologías, este trabajo explorará la escena hispanohablante de Twitch, centrándonos en los y las *streamers* con mayor número de seguidores y que acumulan grandes audiencias.

1.2 Objetivos

Este trabajo pretende arrojar un poco de luz sobre el ecosistema hispanohablante en Twitch. Para ello, el objetivo principal es el desarrollo de una metodología que nos permita analizar las relaciones y comunidades existentes entre diferentes *streamers*. Para estructurar este objetivo plantearemos diferentes subobjetivos o tareas más materiales y evaluables:

1. Establecer un punto de partida en cuanto metodologías empleadas y resultados obtenidos a partir de la literatura científica existente en torno a Twitch.
2. Determinar qué *streamers* debemos analizar y como seleccionarlos para preservar las características poblacionales en nuestra muestra.
3. Desarrollar una solución integrada para la captura y almacenamiento de los datos necesarios para el estudio.
4. Estudiar qué tipo de contenido emiten estos *streamers* y la existencia de patrones temporales.
5. Explorar la existencia de comunidades de *streamers* según el tipo de contenido emitido u otras relaciones.

1.3 Estructura

Para alcanzar los objetivos propuestos primeramente ha sido necesario establecer un punto de partida base mediante una revisión bibliográfica previa, el estudio de las API públicas de Twitch y el planteamiento de un sistema acorde a ellas y que nos permita recoger y almacenar todos los datos necesarios. Posteriormente podremos proceder a analizar estos datos y extraer las conclusiones pertinentes.

Estas líneas generales se han visto plasmadas de manera orgánica en el trabajo en la siguiente estructura de capítulos y secciones:

- El capítulo 2 sirve como punto de partida mediante una revisión bibliográfica y la crítica al estado del arte que motiva este trabajo (sección 2.1).
- En el tercer capítulo (Materiales y métodos) definimos el alcance de nuestro análisis al establecer qué *streamers* serán objeto de estudio, qué datos vamos a recoger y cómo los vamos a capturar y almacenar.
- El capítulo 4 ofrece una caracterización de la plataforma mediante el análisis de diferentes aspectos de las emisiones realizadas por el conjunto de *streamers* seleccionados.
- Los capítulos 0 y 6 constituyen el grueso del presente trabajo. En estas secciones se estudiará las similitudes entre *streamers* según sus emisiones y se construirá y analizará una red que represente nuestra muestra bajo estudio.
- Finalmente, en el capítulo 7 finalizamos este trabajo con una exposición de las conclusiones extraídas y de posibles líneas de trabajo futuras.

2. Estado del arte

Existe una extensa literatura científica que, desde muy diferentes disciplinas y enfoques, estudia y explora Twitch. Debido a la diversidad de campos y metodologías que se han empleado para estudiar Twitch ,se decidió realizar una revisión bibliográfica de forma holística, para poder valorar el estado de la cuestión desde diferentes perspectivas y, finalmente, realizar un análisis más acotado al análisis de redes.

Tomando como punto de partida la revisión bibliográfica de Erik Harpstead (2019), podemos dibujar una primera imagen de la escena científica en torno a la plataforma. A partir del metaanálisis de 46 artículos científicos, revisados por pares y publicados en inglés entre 2011 y mayo de 2018, Harpstead et al. describen las metodologías y enfoques de la literatura científica hasta la fecha. De este estudio podemos extraer diferentes conclusiones:

1. La producción de textos científicos ha ido en aumento de manera continuada desde 2011, debiéndose esto a la creciente popularidad de Twitch en todo el mundo.

2. El objeto de estudio de los diferentes artículos se ha ido desplazando con el tiempo, inicialmente centrándose en la plataforma de manera macroscópica y, en los últimos años, tratando específicamente aspectos concretos de los sujetos que conforman la plataforma: *streamers* y espectadores.

3. Solo se recogen dos artículos en los que se empleen técnicas de análisis de redes sociales, siendo más utilizadas metodologías derivadas de la etnografía, el uso de encuestas o la utilización generalista de datos extraídos de las API públicas de Twitch.

Cómo ya se ha comentado, disciplinas muy diversas han investigado los diferentes aspectos de Twitch. A continuación, destacaremos algunas referencias de especial interés. Podemos empezar hablando del trabajo de Rachel Kowert (2021) y de las nuevas formas de relaciones parasociales que Twitch acoge. La interactividad que ofrece el *chat* y los mensajes añadidos a las subscripciones rompen con la visión tradicional de relación parasocial entendida como *one-sided, non-reciprocal relationships with media figures* (relaciones unilaterales no reciprocas con figuras públicas) para ser reinterpretadas de forma que son *one-and-a-half sided parasocial relationships* (relaciones parasociales semi-bilaterales). Fruto de los diferentes elementos de interacción entre *streamer* y espectador, tanto propios de la plataforma como externos mediante redes sociales, surgen situaciones en las que algunos espectadores se pueden sentir identificados o interpelados de manera directa y personal por el *streamer* cuando realmente no es así, ya que estos realmente forman parte de una audiencia. La capacidad de que los espectadores sientan cercanía y familiaridad con el *streamer* es uno de los principales factores por los que un espectador decide consumir uno u otro canal.

Destacamos también la aportación de García (2021), en la que, mediante el análisis del chat integrado en Twitch, se puede explorar la adopción de términos procedentes

del inglés y la adaptación de los mismos a las formas gramaticales típicas del castellano. Así mismo, se plantea una explicación a la necesidad de emplear términos directamente en inglés, y es que más allá de una cuestión de prestigio lingüístico o de carácter jerárquico: «*algunos factores a tener en cuenta en el lenguaje que emplean los jugadores de videojuegos MOBA […], serían tanto "la identificación de la comunidad de hablantes y la necesidad comunicativa de carácter internacional", como el hecho de que "la mayoría de las posibles traducciones o correspondencias en español están formadas por más de un elemento, lo que obstaculiza al jugador en tanto que la comunicación se retrasa, algo que va en contra de la dinámica del videojuego a tiempo real"*» (García (2021) citando a Ariza (2015)).

Si seguimos hablando de lingüística computacional o procesamiento de lenguaje natural, una referencia de especial interés es la investigación de Pavel Dolin (2021). Los emoticonos juegan un papel muy importante dentro del ecosistema de Twitch, facilitan la comunicación en chats de miles de personas y son, en muchos casos, seña de identidad de las comunidades creadas en torno a un determinado canal. Estos emoticonos son creados por la propia comunidad; cada canal, según su tamaño, puede disponer de un mayor o menor número de ellos y pueden ser utilizados por los subscriptores de dicho canal. Según Dolin, existen más de 8 millones de emoticonos. La continua incorporación de nuevos y la evolución de sus significados hace que llevar a cabo tareas de procesamiento de lenguaje natural sea muy difícil dada la escasez de conjuntos de datos disponibles. Mediante la aplicación de técnicas de aprendizaje automático (*word-embeddings* y k-nn), Dolin et al. (2021) plantean una solución para obtener un pseudo-diccionario actualizado de la semántica de cada uno de los emoticonos.

Conociendo el papel central que tiene el chat de Twitch, continuamos ahora con la disección planteada por Colin Ford et al. (2017) de los mecanismos lingüísticos mediante los cuales los participantes de chats en los que se reúnen decenas de miles de personas pueden mantener una coherencia discursiva y comunicarse de manera grupal. Los participantes de dichos chats emplean diversas técnicas como la reducción, el bricolaje o la adopción de voz (*shorthanding*, *bricolage* y *voice-taking* respectivamente en inglés según los autores). Estas técnicas permiten la contracción de palabras, la recombinación de emoticonos o expresiones compartidas y que los participantes adopten puntos de vista y formalismos compartidos de manera emergente, sin la necesidad de unos códigos o normas preestablecidas. A pesar del gran volumen de mensajes por segundo que aparecen en estos chats, al emplear las técnicas mencionadas los usuarios pueden mantener una coherencia discursiva común, desapareciendo la comunicación interpersonal para actuar como lo haría una masa de aficionados en un estadio deportivo convencional.

Seguidamente, revisaremos tres referencias acotadas al estudio de redes sociales aplicado a Twitch. Primeramente, Dux (2018), partiendo de una revisión de la bibliografía existente hasta el momento, construye y analiza una red de 167 *streamers* interconectados según el número de seguidores en común. Mediante la aplicación de la teoría de *Use and Gratification*, la cual explora porqué los usuarios consumen determinados contenidos en relación con sus necesidades sociales y qué tipo de gratificación reciben por hacerlo, discute las diversas implicaciones de aplicar esta teoría al contexto de Twitch. El autor procede a analizar la red aplicando diversas

particiones según diversos atributos de cada *streamer*. De este estudio se desprende cómo diferentes comunidades de streamers coexisten de forma que los usuarios pueden ver suplidas sus diferentes necesidades de entretenimiento, así como de participación social y aprendizaje, siendo los canales de tipo casual (emisión de un videojuego sin un carácter competitivo, educativo o de *speedrun*[1]) los que componen el grueso de la red y sirven de enlace entre comunidades más específicas como las competitivas.

Hohyun Jung (2021) propone un nuevo modelo para estudiar el fenómeno de la popularidad de Barbasi y Albert. Se escogen Youtube y Twitch como plataformas a partir de las cuales extraer redes en las que se presupone que se cumple el efecto de Mateo, es decir, aquellos usuarios con más subscriptores son los más proclives a seguir acumulando más subscriptores. En este caso, se extienden los modelos habituales para capturar cómo de popular es un nodo, no solamente mediante el grado de entrada, y se utiliza también el número de visitas a un vídeo o canal en un lapso de tiempo determinado.

Por último, Rozemberczki (2021) desarrolla un conjunto de datos para evaluar la calidad de diferentes (y futuros) modelos de *node embeddings* (representación de los nodos de una red en un espacio vectorial). La red se construye mediante relaciones de seguimiento mutuas entre usuarios. Con 168.000 nodos y 6,79 millones de ejes, se presenta una red no dirigida de tamaño mediano en la cual todos los nodos cuentan con ciertos atributos de diferentes tipos que pueden ser empleados por modelos de aprendizaje automático para realizar tareas de clasificación o regresión.

Antes de cerrar esta sección, me gustaría discutir una referencia que ha motivado en gran medida este trabajo. Claudia Flores-Saviaga (2019) realiza una clasificación de diferentes canales o *streamers* en cinco grupos (pequeñas comunidades, *streamers* prometedores, habladores, *streamers* famosos y celebridades o competiciones) mediante la aplicación de técnicas de clusterización. A partir del número medio espectadores concurrentes, número de participantes en el chat, número de mensajes y su sentimiento asociado y una revisión manual de la actuación de los *streamers* de cada clase, se presenta una clasificación y se realiza una comparativa. Este artículo me resultó de gran interés al inicio de la revisión bibliográfica del presente trabajo ya que presentaba la aplicación de diversas técnicas de análisis y presentaba unos resultados muy enriquecedores a la hora de entender el ecosistema de Twitch y los diferentes roles que juegan los usuarios. Así mismo, el análisis de la actuación de los streamers y de cómo estos interactúan con los espectadores mediante las diferentes interfaces, reafirma lo expuesto anteriormente en referencia a las relaciones parasociales y cómo estas forman parte de la dinámica de monetización de la plataforma:

> *Si bien las fuentes de interacción entre los streamers y la audiencia incluyen mensajes de chat, suscripciones y donaciones, observamos que las suscripciones y las donaciones no parecen conducir típicamente a interacciones prolongadas. En cambio, las transacciones financieras parecen ofrecer un "espacio" de reacción posible más restringido para los streamers, es decir, responder con un reconocimiento de la acción de la audiencia junto con una valoración positiva de alguna forma ("¡gracias por la suscripción!"). Como parte de esto, los streamers parecen mucho más propensos a*

[1] Acabar un videojuego en el menor tiempo posible.

> *incorporar elementos de identificación (es decir, nombres de usuario) en el reconocimiento de las suscripciones y donaciones que en los mensajes de chat. Esta práctica no parece estar motivada por la necesidad de desambiguación (es decir, identificar claramente a un miembro de la audiencia de otro), ya que las suscripciones y donaciones se entregan directamente en el vídeo de la transmisión y no a través del chat. En cambio, podría servir más para motivar a otros miembros de la audiencia a realizar acciones similares.* (Claudia Flores-Saviaga, 2019).

2.1 Crítica al estado del arte

La literatura científica disponible que explora Twitch desde la perspectiva del análisis de redes es muy escasa. Encontramos mucho material que analiza el medio desde otras disciplinas como las expuestas anteriormente, pero las referencias que exploten las posibilidades que representa Twitch en el contexto del análisis de redes sociales son sorprendentemente pocas. Recordemos que la revisión bibliográfica (Erik Harpstead, 2019) solamente encontró 2 artículos que trataran esta cuestión. A partir de todo lo expuesto anteriormente planteamos dos críticas:

1. Los artículos que realizan análisis de redes construyen sus grafos a partir de las relaciones de seguido/no seguido entre los diferentes nodos, aspecto que puede no ser lo suficientemente representativo del conjunto real de usuarios que consumen el contenido de un determinado *streamer*. Para paliar esa diferencia entre seguidores y personas que realmente consumen un canal, en el presente trabajo se construirá una red a partir de los usuarios que participan en el chat de una retransmisión y, por tanto, se tratará un grafo en el que los *streamers* estén relacionados entre sí por audiencia real, no por seguidos y seguidores.

2. La plataforma está sujeta a cambios constantes y su evolución ha sido muy rápida en los últimos años. Gran parte de la literatura existente se centra en estudiar el ecosistema de Twitch desde la perspectiva de la comunidad de videojuegos. Sin embargo, aunque el *streaming* relacionado con videojuegos sigue ocupando la mayor parte de la plataforma, con el paso de los años han surgido nuevos contenidos y, recientemente, se ha incorporado una cantidad considerable de nuevo público no relacionado directamente con el mundo de los videojuegos. Si realizamos una comparativa de las horas consumidas según tipo de contenido entre abril de 2018, año en el que James Dux (2018) se escribió, y marzo de 2022, observaremos un notable cambio. Teniendo en cuenta los 50 tipos de contenido más emitidos en la comunidad hispanohablante, en 2018 encontramos un total de 1.030.078.620 de horas consumidas (Tabla 1), debiéndose un 10,39% de ellas con contenidos no relacionados con videojuegos. Sin embargo, en 2022 encontramos que las horas consumidas han aumentado hasta 9.200.119.440 (una cantidad casi 9 veces mayor) y, además, ha aumentado el porcentaje de horas consumidas de contenido no relacionado con videojuegos hasta el 34.6% (Figura 1). Es por tanto necesario replantear el estudio de la

plataforma desde una nueva perspectiva más amplia que abandone una visión excesivamente fijada en los videojuegos ya que esta ha sufrido grandes cambios en los últimos años.

Abril - **2018**	**Categorías**	**Horas consumidas**	**Horas emitidas**
Relacionadas con videojuegos	47	923.017.500	30.875.340
No relacionadas	3	107.061.120	1.668.480
Total	50	1.030.078.620	32.543.820
Marzo - **2022**	**Categorías**	**Horas consumidas**	**Horas emitidas**
Relacionadas con videojuegos	43	6.010.301.880	263.448.000
No relacionadas	7	3.189.817.560	46.437.120
Total	50	9.200.119.440	309.885.120

Tabla 1. Evolución del contenido no relacionado con videojuegos. Fuente (Boyd, 2022).

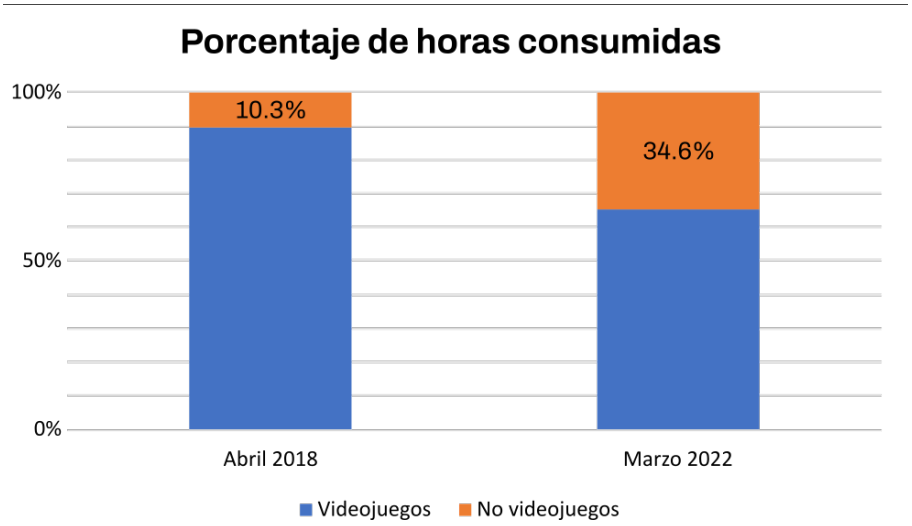

Figura 1. Comparativa del porcentaje de horas consumidas de contenido no relacionado con videojuegos entre 2018 y 2022. Punto separador decimal. Elaboración propia.

3. Materiales y métodos

3.1 Selección de *streamers*

Como en todo experimento, debemos definir previamente las condiciones en las que este se va a desarrollar y los objetos (o sujetos) de estudio. El primer paso a realizar es la selección de los *streamers* de los cuales recogeremos datos de sus directos. Como ya se ha comentado previamente, nos centraremos en la comunidad de streamers hispanohablantes. Empleando el sitio web *TwitchTracker.com* (TwitchTracker, 2022), un servicio web en el que se ofrecen estadísticas actualizadas de la plataforma Twitch, podremos obtener un listado de los 500 canales con mayor número de seguidores a fecha de 23/03/2022. De este primer listado, se eliminan 79 *streamers* por no haber emitido en los últimos 30 días, reduciendo la lista por tanto a 421.

A continuación, resulta de interés llevar a cabo un pequeño análisis exploratorio de las diferentes métricas que *TwitchTracker* ofrece. El listado de campos disponible es el siguiente:

Campo	Descripción
Avg viewers	Número medio de espectadores concurrentes en los últimos 30 días.
Time streamed	Horas emitidas en directo en los últimos 30 días.
All time peak viewers	Número máximo de espectadores concurrentes
Hours Watched	Horas consumidas durante los últimos 30 días
Followers gained	Número de seguidores ganados o perdidos en los últimos 30 días
Total followers	Número total de seguidores
Total views	Número total de reproducciones del canal

Tabla 2 Campos disponibles en TwitchTracker

Es importante destacar una limitación de esta fuente de datos: los campos marcados en azul claro en la Tabla 2 no se proporcionan como números con resolución completa. Es decir, las cantidades para esos campos se redondean a un valor y se sufijan con una letra ("K" para miles o "M" para millones) que indica el multiplicador a aplicar. Por ejemplo, podemos tomar un *streamer* cualquiera y encontraremos que tiene "966K" seguidores, es decir, podemos afirmar que tiene del orden de 966.000 seguidores, pero no podemos asegurar que sean una cantidad exacta diferente a esa, por ejemplo 966.001 o 966.999. Esta limitación no se ha demostrado importante a la hora de llevar a cabo los análisis aquí presentados.

Comenzaremos analizando el número total de seguidores por *streamer*, ya que es el criterio que hemos empleado para ordenar y, por tanto, escoger los *streamers* que

formaran parte del estudio. Observamos una distribución donde la gran mayoría de *streamers* (>89%) se sitúan en el intervalo de 158.000 a un millón de seguidores. El resto de los *streamers* se dispersan entre el millón y los 10 millones de seguidores, situándose el grueso de estos en el intervalo de un millón a tres millones con un ~9% del total. Aquellos que alcanzan cantidades más elevados son muy pocos: solamente 8 *streamers* tienen 3 millones de seguidores o más (Figura 2).

Figura 2. Histograma del número de seguidores de cada streamer. Elaboración propia.

Es evidente que es una distribución muy desigual, con una cola relativamente larga a la derecha. Si nos fijamos en el subconjunto de *streamers* con menos de un millón de seguidores (Figura 3) podemos ver como se reproduce la misma situación, aunque de una manera no tan aguda. El ~70% de estos *streamers* tienen como máximo, 408.000 seguidores.

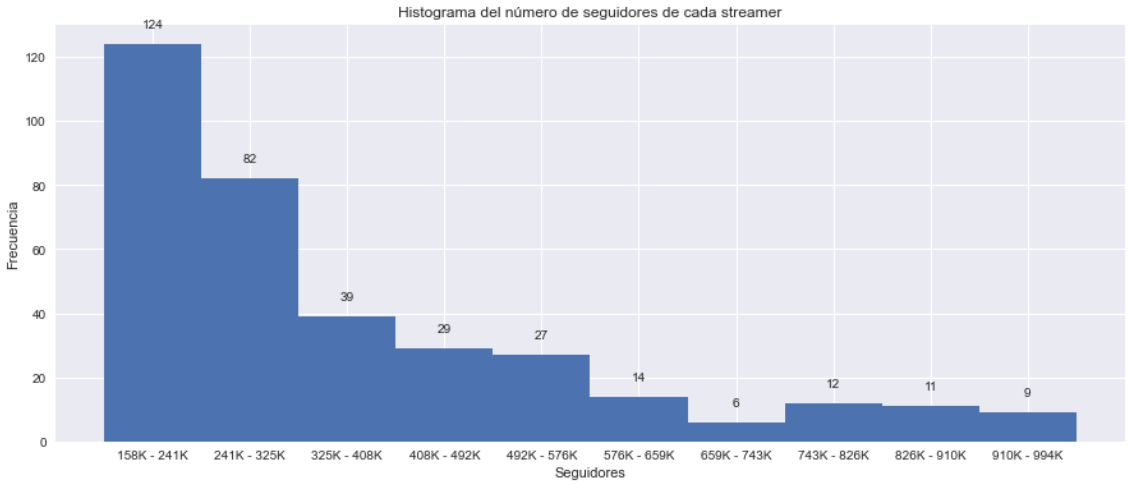

Figura 3. Histograma del número de seguidores de cada streamer (con menos de un millón de seguidores). Elaboración propia.

Podemos, cautelosamente, suponer que esta dinámica se reproduce a lo largo de todo el espectro de *streamers* de la plataforma, de manera que la gran mayoría de ellos reunirán muy pocos seguidores, mientras que una larga cola de pocos *streamers* acumularán cantidades millonarias de seguidores. Es importante, por tanto, ser conscientes de que este estudio se sitúa en esta larga cola donde pocos *streamers* tienen

muchos seguidores, y presumiblemente, grandes audiencias, algo alejado de la realidad macroscópica de la plataforma, donde muchos otros usuarios y usuarias emiten en directo para pequeñas audiencias. Podemos presuponer que muchos, quizás no todos, los *streamers* objetos de este estudio serán profesionales en este campo, recibiendo su principal sustento del *streaming* en Twitch.

Para continuar indagando en este desigual reparto de seguidores, podemos prestar atención ahora a otras dos variables: horas de visualización y número medio de espectadores concurrentes. Si visualizamos estas dos variables de manera cumulativa, podemos observar como la gran mayoría de *streamers* contribuyen muy poco a la suma total.

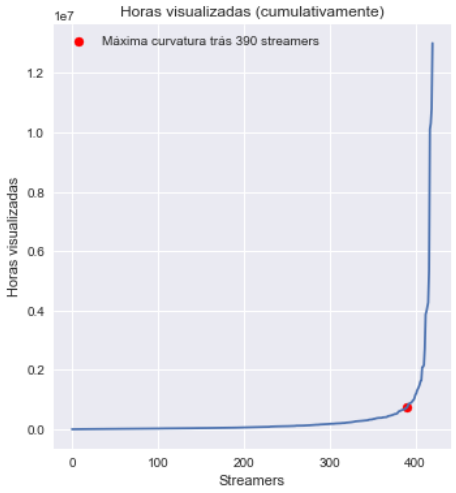

Figura 4. Horas visualizadas cumulativamente. Punto separador decimal. Elaboración propia.

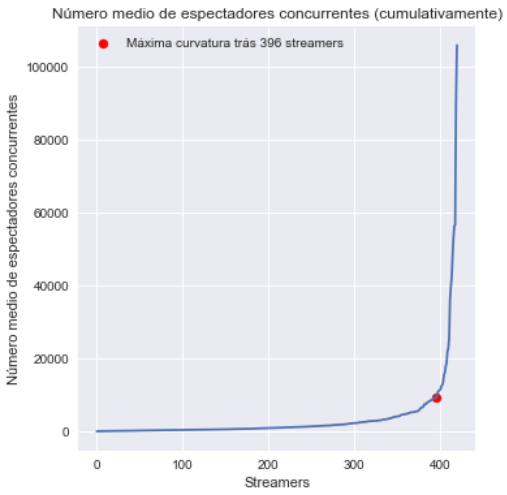

Figura 5. Número medio de espectadores concurrentes cumulativamente. Punto separador decimal. Elaboración propia.

En la Figura 4 y la Figura 5 podemos observar como, en ambos casos, encontramos una subida muy abrupta tras ~390 *streamers*, es decir, el ~92% de los *streamers* contribuyen muy poco a la suma total, siendo el 8% restante, los que concentran las grandes cifras de seguidores y espectadores, los que acumulan la mayoría de horas visualizadas y espectadores concurrentes. Los puntos de inflexión se definen como los puntos de máxima curvatura en la curva que representa la suma cumulativa de cada variable.

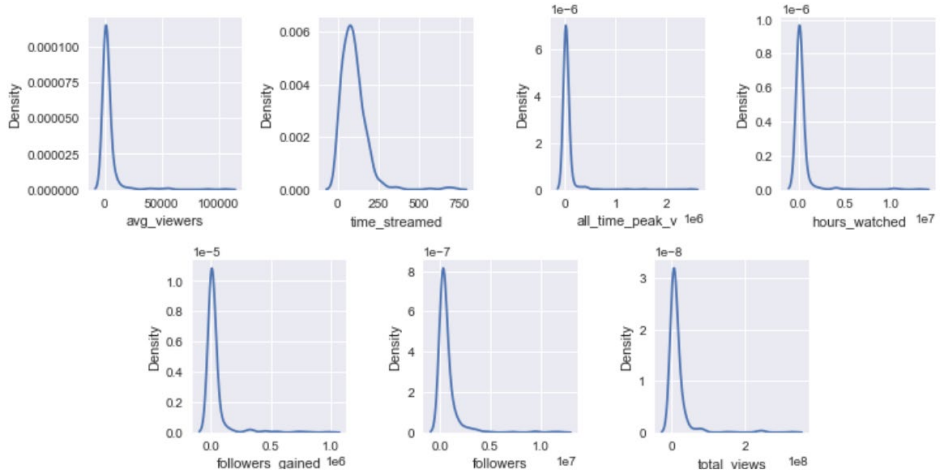

Figura 6. KDE para todas las variables. Punto separador decimal. Elaboración propia.

Calcular una estimación de la función de probabilidad utilizando kernels gausianos (Rosenblatt, 1956; Waskom, 2012) para todas las variables (Figura 6) es bastante elocuente en tanto que nos permite ver claramente como todas las variables siguen distribuciones muy similares, con largas colas a la derecha. La mayoría de *streamers* presentan valores pequeños, mientras que unos pocos *streamers* acumulan la mayoría de los seguidores, espectadores y visualizaciones. La variable que presenta mayor amplitud es la de tiempo emitido, siendo el resto muy agudas.

Por último, exploraremos la correlación que puede existir entre número de horas emitidas, el número medio de espectadores concurrentes y la variación en el número de seguidores. Para ello, transformaremos las tres variables aplicando el logaritmo neperiano, ya que, como hemos visto anteriormente, siguen distribuciones cercanas a exponenciales. De esta forma, obtenemos distribuciones más "normalizadas", permitiéndonos realizar análisis más robustos (Figura 7 y Figura 8).

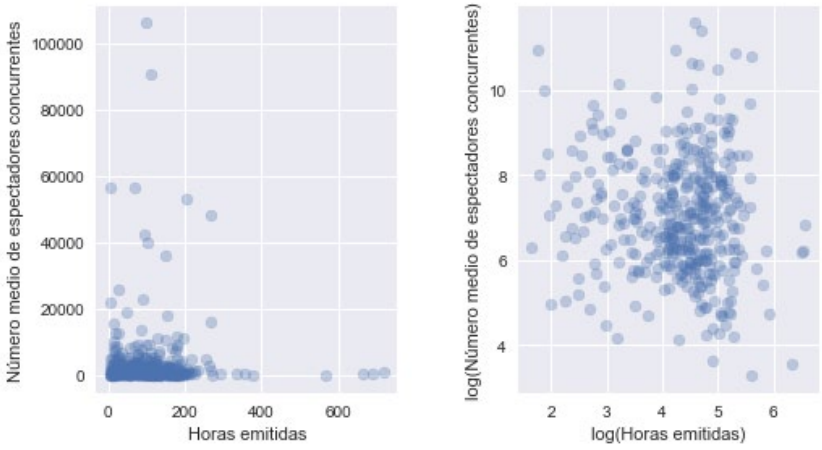

Figura 7. Distribución de las variables Horas emitidas y Número medio de espectadores concurrentes, antes y después de aplicar el logaritmo. Elaboración propia.

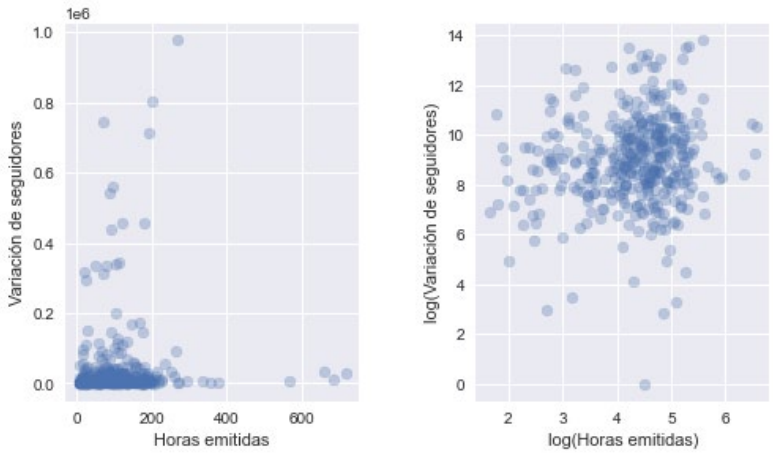

Figura 8. Distribución de las variables Horas emitidas y Variación de seguidores, antes y después de aplicar el logaritmo. Elaboración propia.

Tras transformar los datos, calcularemos diferentes coeficientes de correlación entre las dos variables. Para ello, seleccionamos los tres coeficientes más utilizados: Pearson, Spearman y Kendall. En las figuras Figura 9 y Figura 10 encontramos las variables en cuestión representadas, acompañadas de una regresión lineal y su área de

intervalo de confianza al 95%. De manera análoga, en las tablas Tabla 3. Coeficientes de correlación y p-value entre Horas emitidas y Número medio de espectadores concurrentes y Tabla 4. Coeficientes de correlación y p-value entre Horas emitidas y Variación de número de seguidores encontraremos los valores de los coeficientes acompañados de su p-valor.

El cálculo de los coeficientes de correlación para *Horas emitidas* según *Número medio de espectadores concurrentes* no arroja un resultado concluyente, ya que para Spearman y Kendall obtenemos p-valores elevados. El coeficiente de Pearson es igual a -0,10 con un p-valor inferior a 0.05, por lo que podríamos afirmar que existe una ligera correlación negativa. Sin embargo, al ser Pearson más sensible a valores atípicos y el resto de los coeficientes, más robustos frente a valores extremos y distribuciones no-nomales, devolviendo p-valores elevados, concluimos que no existen suficientes evidencias para afirmar que existe una correlación entre las dos variables.

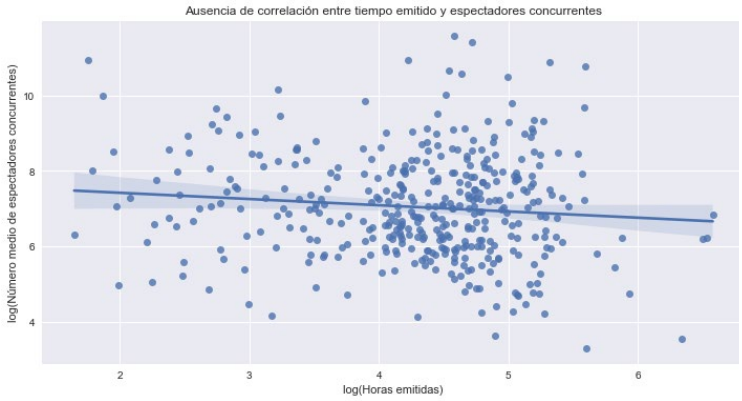

Figura 9. Gráfico de puntos y regresión lineal con intervalo de confianza entre Horas emitidas y Número medio de espectadores concurrentes . Elaboración propia.

Coeficiente	Valor	p-value
Pearson r	-0.1025	0.04
Spearman r	-0.073	0.13
Kendall tau	-0.051	0.12

Tabla 3. Coeficientes de correlación y p-value entre Horas emitidas y Número medio de espectadores concurrentes

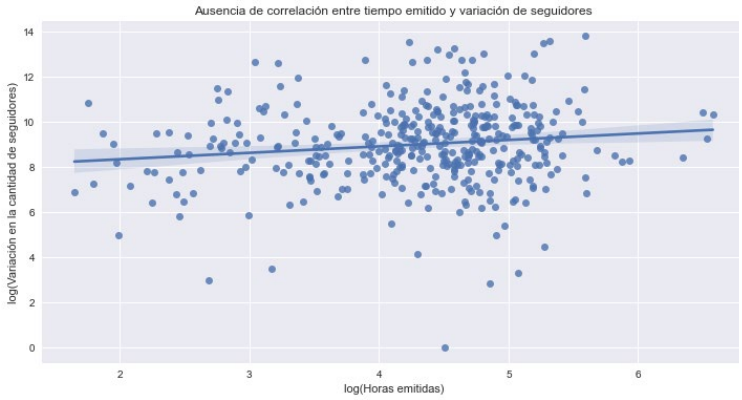

Figura 10. Gráfico de puntos y regresión lineal con intervalo de confianza entre Horas emitidas y Variación de número de seguidores. Elaboración propia.

Coeficiente	Valor	p-value
Pearson r	0.1385	0.005
Spearman r	0.1276	0.01
Kendall tau	0.0881	0.008

Tabla 4. Coeficientes de correlación y p-value entre Horas emitidas y Variación de número de seguidores

En cuanto a la posible correlación entre Horas emitidas y Variación de seguidores, encontramos que todos los coeficientes obtienen p-valores inferiores o iguales a 0.01.

Los tres coeficientes arrojan que existe una pequeña correlación positiva entre las dos variables, siendo la τ de Kendall la que devuelva el valor más pequeño (0.08) y Pearson el mayor (0.13).

Una vez explorada la naturaleza y distribución de las variables que caracterizan a los posibles sujetos de estudio, se procede a realizar un muestreo de la población, dado que no sería viable intentar recolectar datos para el conjunto completo de *streamers* debido al ingente volumen de los mismos y las limitaciones computacionales a las que se ciñe el proyecto. Tras cerciorarnos de la desigual distribución de todas las variables para la población, se decide muestrearla dividiendo los *streamers* en 10 tramos de igual longitud según su número de seguidores. De cada tramo se escoge aleatoriamente 20 *streamers*, obteniendo un total de 200, asegurando que la muestra con la que vamos a trabajar de aquí en adelante presenta distribuciones de las variables son similares a las exhibidas a nivel poblacional (Figura 11) y siendo así lo más representativa posible de la población. El listado de *streamers* que forman parte de la muestra puede encontrarse en el Anexo I.

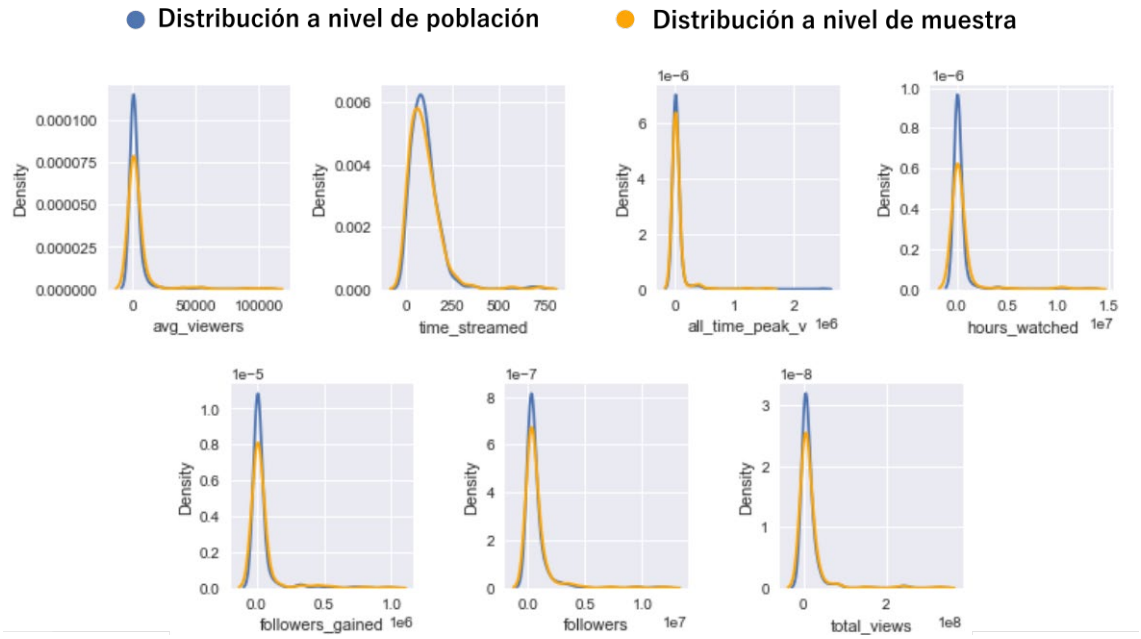

Figura 11. Comparativa de KDEs entre población y muestra. Elaboración propia.

3.2 Sistema de recogida de datos

Para la realización del presente trabajo, ha sido necesario recoger datos más allá de aquellos típicamente ofrecidos en los diferentes agregadores de estadísticas como *SullyGnome* (Boyd, 2022) o *TwitchTracker* (TwitchTracker, 2022). Dado que el objetivo de este proyecto es el análisis del ecosistema hispanohablante de Twitch desde la perspectiva del análisis de redes, es necesario preguntarnos qué clase de datos nos ofrece Twitch mediante sus API públicas que podamos emplear para construir redes. Además, se hace evidente la necesidad de ir un paso más allá de lo expuesto en el

capítulo del estado del arte, ya que es común encontrar diferentes trabajos en los que se construyen grafos basándose en la relación de seguido/no seguido, presente en esta plataforma como en tantas otras redes sociales.

La construcción y análisis de redes empleando la relación de seguido/ no seguido presenta diversas limitaciones que han sido expuestas en la crítica al estado del arte. Es por ello que se hace necesario buscar una alternativa. Idealmente, si queremos estudiar las comunidades de *streamers* según su audiencia, no habría mejor manera de hacerlo que pudiendo obtener un listado de los usuarios que en un momento dado se encuentran visualizando un directo. Sin embargo, es fácil y acertado asumir que Twitch no ofrece dicha información.

Por el contrario, aunque no podamos obtener un listado completo de los espectadores, sí que podemos obtener un subconjunto de ellos gracias al chat. Twitch ofrece un *endpoint*[2]*,* no documentado y fuera de su sistema de API públicas, en el que encontraremos aquellos usuarios conectados al chat del directo de un *streamer* dado. Según el portal de ayuda de Twitch (Twitch Interactive, Inc, 2022) un *chatter* es todo aquel usuario registrado que, habiendo iniciado sesión y visualizando un *stream,* está conectado al chat, es decir, este es visible en la interfaz. Esta definición resulta muy interesante ya que no se considera *chatter* solamente a aquellos que han participado en el chat en algún momento, sino que engloba a todos los usuarios, tanto aquellos que escriben como aquellos que no.

Por tanto, mediante este *endpoint,* podremos construir redes mucho más fidedignas que empleando simplemente la relación de seguido/no seguido, ya que podemos capturar de manera precisa la audiencia real y calcular similitudes y diferencias entre *streamers* y poder relacionarlos entre sí en un grafo. Sin embargo, esta fuente de datos también viene acompañada de ciertos inconvenientes, y es que al no formar parte de las API oficiales y mantenidas, esta puede cambiar o dejar de funcionar en cualquier momento sin aviso previo (Twitch Interactive, Inc., 2020; Twitch Interactive, Inc., 2019). Además, este *endpoint* puede estar sujeto a mucho *caching* (Twitch Interactive, Inc., 2020) por lo que la cantidad de *chatters* y espectadores puede variar.

Más allá de los usuarios presentes en el *stream,* resulta interesante poder capturar otros atributos de la emisión, como puede ser el número total de espectadores, el título, la categoría o la hora de comienzo. Twitch cuenta con una extensa documentación de sus API[3] que nos permiten recoger toda la información deseada. Dentro del ecosistema de desarrollo actual de Twitch, el *endpoint* que nos permite registrar a los *chatters* pertenece al servicio TMI (de las siglas en inglés para *Twitch Messaging Interface)*, mientras que el resto de los atributos serán capturados mediante la API *Helix*.

Implementar el sistema de recogida empleando directamente los *endpoints* hubiera añadido mucho tiempo de desarrollo. Es por ello que la opción más razonable es emplear una librería que implemente por nosotros las rutinas y llamadas a las API de Twitch. Existen multitud de librerías de código abierto que se podrían emplear para

[2] http://tmi.twitch.tv/group/user/{streamer}/chatters
[3] https://dev.twitch.tv/docs/

esta tarea (Twitch Interactive, Inc., 2022). Finalmente se escogió utilizar Twurple (Twurple, 2022), una librería para Node.js[4] que implementa la gran mayoría de las funcionalidades ofrecidas por Twitch, así como rutinas para facilitar la autentificación y la obtención de tokens necesarios para comunicarse con las API en cuestión.

Hasta ahora hemos expuesto la información que recopilaremos (*chatters*, número total de espectadores, título del directo, categoría y hora de comienzo), la librería que emplearemos para ello, Twurple, que nos permitirá hacer uso tanto de las API públicas oficiales como del *endpoint* no mantenido para obtener los usuarios presentes en el chat. Finalmente, hay que destacar que esta información no se capturará de manera estática o una sola vez por cada directo de los *streamers* seleccionados, si no que la recopilaremos de manera dinámica, consultando las fuentes de datos en un intervalo de tiempo determinado para poder obtener una radiografía durante toda la duración de los directos. Fijaremos este intervalo en 5 minutos, ya que parece un compromiso equilibrado entre la resolución de los datos y la capacidad de almacenamiento y procesamiento disponible. Además, escogiendo un intervalo menor podemos padecer del *caching* que Twitch aplica a sus *endpoints,* acabando con datos superfluos.

El procedimiento a seguir, a grandes rasgos, es el siguiente:

1. Esperar a que alguno de los *streamers* seleccionados comience a emitir.
2. Al comenzar un *stream*, realizar una primera captura de los datos indicados anteriormente y ejecutar de manera recurrente una rutina cada 5 minutos para realizar la captura de datos. Esta captura recurrente no incluirá la hora de comienzo y el título del directo dado que ya fueron consultados en la primera captura.
3. Al detectar que el directo ha finalizado cancelar la ejecución recurrente de la rutina y volver al primer paso.

Tenemos dos opciones a la hora de abordar el primer paso: siguiendo una aproximación *polling* podemos consultar iterativamente cada *streamer* hasta que alguno de ellos inicie una emisión o bien podemos emplear una estrategia reactiva, basada en eventos, gracias a la cual Twitch nos notificará cuando un *streamer* comience un directo. Empleando la segunda estrategia, aun pudiendo resultar más compleja de configurar e inicializar, obtenemos un sistema más robusto y eficiente ya que no malgastamos recursos computacionales de manera innecesaria.

Esta estrategia es posible gracias al servicio *EventSub*[5] de Twtich, que permite suscribirse a multitud de eventos[6] de la plataforma y recibir notificaciones mediante *webhooks,* es decir, Twitch realizará una petición, normalmente HTTP utilizando SSL, a una dirección web que nosotros hayamos indicado. Al recibir dicha petición, el servidor la procesará, identificando el tipo de notificación y podrá actuar en consecuencia. En nuestro caso la notificación nos avisará que un *streamer* ha iniciado una emisión y actuaremos ejecutando nuestro procedimiento de recogida de datos.

[4] Implementación de JavaScript orientada a servidores
[5] https://dev.twitch.tv/docs/eventsub
[6] https://dev.twitch.tv/docs/eventsub/eventsub-subscription-types

La elección de emplear Twurple no es casual, ya que esta librería provee de una implementación bastante completa de la funcionalidad ofrecida por el servicio *EventSub*. Además, incluye rutinas para integrar la librería como un *middleware* de Express.js[7], lo que resulta muy conveniente y facilita enormemente el proceso de desarrollo.

Una vez decidida de manera general los elementos lógicos a utilizar, debemos decidir cómo alojar dicho servicio, que se debe mantener a la escucha de los eventos de Twitch. La opción más factible y robusta parece aprovechar los créditos de Microsoft Azure que la escuela pone a disposición de los alumnos. Azure es la plataforma servicios y productos de cómputo en la nube de Microsoft. En ella podemos encontrar multitud de soluciones que se ajustan a las necesidades de este proyecto. Una de estas soluciones es App Service[8], un servicio administrado de despliegue de API y aplicaciones web, perfecto para la implementación de nuestro servicio.

Respecto al segundo paso del procedimiento se nos plantean diversas cuestiones. La primera que abordaremos será la referente a la elección de recoger diferentes datos en la primera captura respecto del resto. Esta decisión se toma de forma orgánica, dado que es evidente que no resulta necesario almacenar repetidamente la fecha y hora en la que ha comenzado la emisión, con capturarla al inicio una vez es suficiente. El otro campo que descartamos de las capturas recurrentes es el título de la emisión. Si bien es habitual que los y las *streamers* cambien el título durante el transcurso de la emisión, este no resulta de gran relevancia para el análisis que llevaremos a término en este trabajo, por tanto, lo almacenaremos solamente al inicio.

Además de los campos ya expuestos anteriormente se hace necesario incluir tres más: uno para poder identificar cada emisión de manera unívoca, otro para poder relacionar las emisiones con los *streamers* correspondientes y finalmente una marca de tiempo. Por ello incluiremos en la captura de datos el identificador de emisión asignado por Twitch, el o la *streamer* de dicha emisión y el momento en el que ha sido capturada dicha información (Figura 12 y Figura 13).

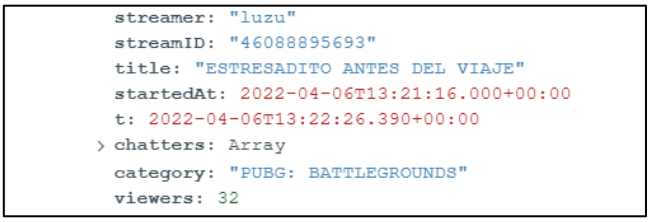

Figura 12. Ejemplo de datos capturados en el momento de recibir el evento de emisión iniciada.

[7] Popular *framework* web para el desarrollo de APIs o aplicaciones web en Node.js. Más información en https://expressjs.com/
[8] https://azure.microsoft.com/es-es/services/app-service/#overview

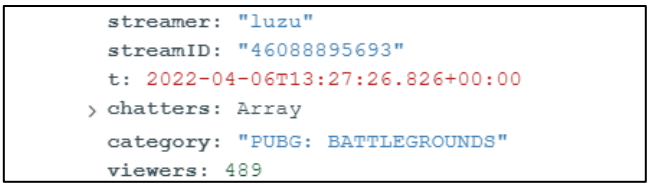

Figura 13. Ejemplo de datos capturados de manera recurrente.

Una segunda duda que es necesaria de resolver es cómo llevar a cabo la ejecución recurrente la rutina de recogida de datos. Dado que hemos decidido alojar nuestro servidor de recogida de datos en la plataforma Azure, podemos considerar el servicio Azure Functions[9], destinado a ejecutar pequeños fragmentos de código de manera reactiva, como por ejemplo, mediante peticiones HTTP o intervalos temporales. Sin embargo, dada la limitada envergadura de este proyecto, no ha sido necesario hacer uso de este servicio, ya que como veremos más adelante, empleando la rutina nativa `setInterval`[10] de Node.js, se han obtenido resultados más que satisfactorios. Esta rutina permite llamar periódicamente a una función tras especificar el intervalo correspondiente en milisegundos.

Por último, es necesario reparar como almacenaremos los datos que sean recolectados. Para tal fin se ha optado por utilizar una base de datos NoSQL como MongoDB debido a su flexibilidad y rendimiento. En el ecosistema de Azure encontramos diversas soluciones de almacenamiento, siendo Cosmos DB la más idónea para este proyecto, ya que se trata de un servicio completamente administrado de bases de datos NoSQL que podremos utilizar fácilmente gracias a la librería de Node.js *mongoose*[11].

En resumidas cuentas, la arquitectura de recogida de datos desarrollada para este trabajo se basa en un servicio en la nube que de manera reactiva, es notificado por Twitch cuando uno de los *streamers* seleccionados inicia una retransmisión. Tras este evento, se realiza una captura de datos inicial y desde ese momento hasta que acabe el *stream* se recogerán datos en un intervalo de 5 minutos. Estos datos serán almacenados en una base de datos MongoDB para su posterior consulta y análisis (Figura 14. Diagrama de la infraestructura planteada).

Una vez definido el sistema y los componentes a utilizar, se procedió a al desarrollo e implementación de una solución. Tras finalizar este proceso se decide llevar a cabo una prueba a pequeña escala, seleccionando un reducido número de *streamers* para evaluar la estabilidad del sistema, corregir posibles fallos y comprobar los requerimientos computacionales, tanto de almacenamiento como de uso de CPU en el plan más básico ofrecido por Azure.

La prueba comenzó el 6 de marzo y finalizó el día 10 del mismo mes. Con un subconjunto de 50 *streamers* escogidos aleatoriamente se pudo comprobar el correcto funcionamiento del sistema y la estabilidad del mismo. Así mismo, esta prueba permitió recoger un pequeño conjunto de datos con el que poder desarrollar utilidades o análisis que posteriormente serian aplicados al conjunto de datos completo.

[9] https://azure.microsoft.com/es-es/services/functions/#overview
[10] https://nodejs.org/api/timers.html#setintervalcallback-delay-args
[11] https://mongoosejs.com/

En el Anexo II se presenta un análisis pormenorizado del rendimiento del sistema y de la estabilidad de emplear la función `setInterval` para orquestar la ejecución repetida de la rutina de captura de datos.

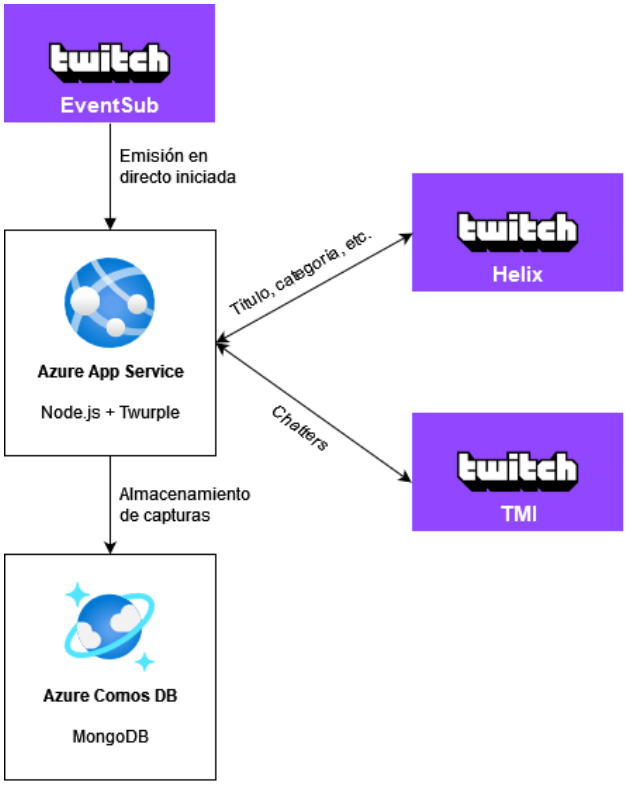

Figura 14. Diagrama de la infraestructura planteada

4. Caracterización de los directos en Twitch

El proceso de recogida de datos fue iniciado el día 18/03/2022 a las 13:37 y finalizó el 21/04/2022 a las 12:26, comprendiendo un total de 33 días, 22 horas y 49 minutos. En dicho intervalo se realizaron 236.535 capturas de datos relativas a 4.370 emisiones en directo de 191 *streamers*. Una vez finalizada la fase de recogida de datos, la base de datos fue volcada en un equipo local para su procesamiento y análisis. Del total de capturas, 50 de ellas tuvieron que ser desechadas por algún error en su contenido o formato, resultando el conjunto de datos en 236.485 capturas.

El conjunto de datos obtenido presenta muchas posibilidades y ángulos desde el cual abordarlo. Estos datos son una instantánea del ecosistema hispanohablante de Twitch, nos proporciona información sobre los contenidos emitidos, los patrones temporales (duración de directos, frecuencia temporal, hora de inicio, ...) y los hábitos de consumo de los espectadores. En esta sección realizamos una breve descripción de alguno de estos aspectos con la finalidad de entender mejor tanto los datos con los que estamos trabajando como la realidad que presentan.

4.1 Patrones temporales de emisión

Tal y como se ha indicado previamente, el conjunto de datos lo forman 236.485 capturas. Debido a la base de datos empleada, estas capturas se almacenan como documentos en formato de diccionario clave-valor (Figura 12, Figura 13). Cada uno de estos documentos, ocupa de media 46,2KB. La distribución temporal de las capturas no presenta grandes diferencias entre los días de la semana. Encontramos que el martes es el día que más capturas se han realizado, seguido del lunes y el sábado. El jueves y el domingo son los días con menos registros (Figura 15).

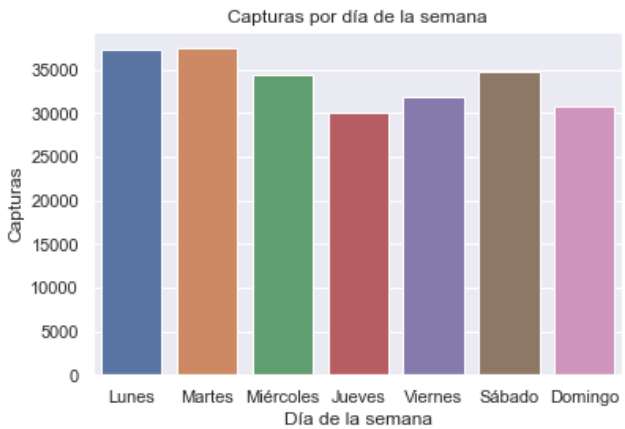

Figura 15. Número de capturas por día de la semana

La distribución diaria de las capturas no arroja ningún resultado especialmente llamativo. Se alcanza el máximo de registros diarios el día 16 de abril, con un total de 10.271 capturas, seguido de los días 8 de abril y 21 de marzo con ~9.000 cada día. En cambio, al agregar el número de capturas por hora del día sí que podemos extraer conclusiones más interesantes (Figura 16).

Para llevar a cabo la agregación horaria denotaremos el conjunto de capturas como $\mathcal{C}$ y una capturada dada como $c_i = \{t, streamID,\ streamer, espectadores\}$, simplificando así el modelo de datos explicado en el capítulo anterior para facilitar el procesamiento y el presente desarrollo. Podemos referirnos a los diferentes atributos de cada captura empleando su nombre y el subíndice de la captura correspondiente. Calcularemos dos métricas: número de capturas registradas y número total de espectadores, ambas relativo a cada hora, y las escalaremos al intervalo [0, 1] para poder situarlas en una misma gráfica y poder compararlas. Definimos las métricas como sigue:

$$f(i,j) = \begin{cases} 1 & si\ la\ hora\ de\ t_i\ es\ igual\ a\ j \\ 0 & en\ caso\ contrario \end{cases}$$

$$capturas_j = \sum_{\substack{c_i:\ f(i,j)=1 \\ \forall i}} 1\,, \qquad j \in \{0,1,2,\dots,23\}$$

$$espectadores_j = \sum_{\substack{c_i:\ f(i,j)=1 \\ \forall i}} espectadores_i\,, \qquad j \in \{0,1,2,\dots,23\}$$

Ecuación 1. Métricas agregadas por hora

Una vez calculadas las métricas, serán escaladas mediante el siguiente procedimiento:

$$espectadores = \frac{espectadores - \min(espectadores)}{\max(espectadores) - \min(espectadores)}$$

Ecuación 2. Escalado de las métricas

La distribución horaria refleja un patrón bastante más significativo (Figura 16). En nuestra muestra de *streamers* observamos cómo durante la mañana se registra la menor cantidad de capturas. Conforme avanza la mañana, se registran poco a poco más capturas. A partir de la 13:00 se produce una rápida subida, alcanzando el máximo a las 18:00, para ir reduciéndose gradualmente durante la noche. Si reparamos en la suma de espectadores encontramos una dinámica parecida, pero con ciertas diferencias. El crecimiento paulatino de capturas durante la mañana no se reproduce de manera equivalente en los espectadores. El número de espectadores no comienza a crecer significativamente hasta las 14:00, mientras que las capturas comienzan varias horas antes. Conforme avanza la tarde, esta diferencia entre capturas y espectadores se reduce, alcanzando ambas métricas sus valores máximos a las 18:00. Algo similar ocurre durante la noche, en la que las capturas descienden progresivamente mientras que los espectadores lo hacen de una manera mucho más aguda.

Figura 16. Capturas por hora. Punto separador decimal. Elaboración propia.

4.2 Categorías

Durante el proceso de captura de datos, se han registrado 369 categorías diferentes, con un reparto muy desigual del tiempo de emisión de cada una de ellas (1291 capturas no presentaban categoría, el ~0.54% del total). En esta distribución encontramos muchas similitudes con los resultados presentados en el capítulo 3.1: unas pocas categorías concentran la mayor parte del contenido emitido en Twitch. Si realizamos una suma cumulativa de cuantas capturas hay registradas para una categoría determinada, encontramos que solamente 61 categorías, lo que correspondería al ~16% del total de categorías, acumulan el 95% de las capturas (Figura 17).

Figura 17. Distribución cumulativa del número de capturas por categoría

Las categorías más emitidas (Tabla 5) corresponden a videojuegos populares entre el público general en el momento de realizar el estudio. La única categoría no

relacionada con los videojuegos, ASMR, la encontramos en 5ª posición. El ASMR, por las iniciales en inglés para *Autonomous Sensory Meridian Response*, es el fenómeno en el que personas experimentan una sensación de hormigueo en la piel, normalmente comenzando en el cráneo y que puede recorrer la parte posterior del cuello (Barratt, 2015). Esta sensación, habitualmente desencadenada por estímulos audiovisuales, es utilizada normalmente como una técnica de relajación.

	Categoría	Número de capturas
1	Just Chatting	46.015
2	Fortnite	43.577
3	VALORANT	16.006
4	League of Legends	13.762
5	ASMR	10.070
6	Grand Theft Auto V	9.718
7	Minecraft	8.649
8	Counter-Strike: Global Offensive	7.091
9	Ark: Survival Evolved	6.500
10	Call of Duty: Warzone	5.948

Tabla 5. Categorías con mayor número de capturas

Si agrupamos las capturas por sus categorías y sumamos el número de espectadores de cada una de ellas, obtenemos el número total de espectadores que en algún momento han visualizado dicha categoría. Si las ordenamos atendiendo a dicho criterio, obtenemos una clasificación ligeramente diferente (Tabla 6). En primera posición encontramos la categoría de *Just Chatting*, que aventaja, y en mucho, a la siguiente, el videojuego Grand Theft Auto V. Deberemos descender hasta la 11ª posición para encontrar el ASMR. En octava y décima posición destacan las categorías de *Sports* (contenidos relacionados con la práctica o el comentario/análisis de deportes tradicionales) y *Special Events* (eventos especiales como conferencias, convenciones y competiciones), claramente situadas fuera del espectro más tradicionalmente asociado al mundo del *streaming* y posibles indicadores de un cambio profundo en la audiencia de este tipo de servicios, fenómeno discutido previamente en el apartado de Estado del arte.

	Categoría	Número total de espectadores
1	Just Chatting	245.940.004
2	Grand Theft Auto V	165.547.310
3	Fortnite	92.680.572
4	League of Legends	64.145.935
5	VALORANT	28.504.701
6	Minecraft	22.743.685
7	Ark: Survival Evolved	22.060.119
8	Sports	19.563.540
9	FIFA 22	19.330.058
10	Special Events	15.213.751

Tabla 6. Categorías según número total de espectadores

4.3 Tiempo emitido por *streamer*

Se ha registrado una cantidad notable de tiempo de emisión en el transcurso de este estudio. Si tenemos en cuenta que cada captura está espaciada temporalmente 5 minutos y que en total tenemos 236.535 capturas, esto resulta en 19.711,5 horas. En el apartado anterior hemos discutido que categorías acumulan la mayor parte de este tiempo de emisión, ahora nos centraremos en la distribución de este tiempo entre los *streamers*.

Para cuantificar la cantidad de tiempo que un *streamer* ha estado en directo simplemente emplearemos el número de capturas registradas para dicho usuario (Tabla 7). La conversión a horas es trivial tal y como se ha indicado más arriba. En primera posición, con 10.253 capturas encontramos a la *streamer* IJenz, la cual emite contenido de ASMR y cuenta con emisiones de más de 40 horas durante el período de captura de datos. Esta *streamer*, al igual que FrancoEscamillaLIVE, *streamer* en segunda posición con 8.065 capturas, suelen realizar emisiones en las que una pequeña fracción al inicio del directo es contenido propiamente en vivo mientras que el resto se trata de una especie de redifusión de contenido ya emitido previamente, lo que les permiten estar en directo ininterrumpidamente durante gran cantidad de horas. En el caso de FrancoEscamillaLIVE, canal a cargo del humorista mexicano que le da nombre, se centra en la emisión de contenido *Just Chatting* y *Always On*.

	Streamer	Número de capturas
1	ijenz	10.253
2	francoescamillalive	8.065
3	blanchitooo	7.217
4	axozer	5.186
5	palermo	4.848
6	herocharly	3.521
7	rike	3.310
8	lokonazo1	3.168
9	wolfangkillers	3.150
10	biusito	3.119

Tabla 7. Streamers con más tiempo emitido

Si consideramos las categorías más emitidas por el resto de los *streamers* de la tabla, encontramos que son muy similares a las presentadas en la Tabla 5. La gran mayoría de ellas corresponden a videojuegos exceptuando *Just Chatting* y *I'm Only Sleeping*, categoría designada para aquellas emisiones en las que simplemente se muestra al *streamer* durmiendo. Esta categoría es empleada habitualmente por aquellos usuarios que se encuentran realizando un *stream* extensible, es decir, una emisión en la que conforme nuevos usuarios siguen, se suscriben al canal o aportan donaciones, más dura el directo debido a que estas interacciones van aparejadas de una cierta cantidad de tiempo, que se suma a un contador. Al utilizar esta categoría pueden mantener la emisión en directo sin interrupciones.

Si visualizamos la cantidad de capturas registradas por cada *streamer* de manera cumulativa (Figura 18) observamos que describe una curva mucho más suave que algunas otras presentadas anteriormente para modelizar otros atributos (Figura 4, Figura 5 y Figura 17). De los 191 *streamers* para los cuales hemos recogido datos durante el transcurso de este trabajo, 173 (~90%) cuentan con menos de 2.600 capturas. Solamente 5 usuarios cuentan con más de 3.851 capturas, concentrándose la mayoría de los usuarios en el rango de 10 a 1.290 capturas, con 121 usuarios, lo que representa el ~63% del total.

Figura 18. Número de capturas por streamer cumulativamente

Concluimos este apartado comprobando que nuestro conjunto de datos refleja las conclusiones extraídas mediante la Figura 9, en la que se exponía que no se podía afirmar concluyentemente que una mayor cantidad de tiempo de emisión repercutía en un mayor número medio de espectadores concurrentes. Procediendo de manera idéntica a lo explicado más arriba, ya que nuestra distribución de datos es idéntica, alcanzamos la misma conclusión, ya que todos los coeficientes de correlación (Pearson, Spearman y Kendall) devuelven valores muy cercanos a cero y p-valores muy elevados, lejos de estándares estadísticos aceptables (Tabla 8 y Tabla 9). Por tanto, tampoco podemos concluir mediante los datos recogidos en este trabajo que un mayor tiempo de emisión presente una correlación con un mayor número medio de espectadores concurrentes.

Coeficiente	Valor	p-value
Pearson r	0,0083	0,908
Spearman r	0,0797	0,272
Kendall tau	0,0536	0,27

Tabla 8. Correlación entre tiempo de emisión y número medio de espectadores según datos propios

Coeficiente	Valor	p-value
Pearson r	0,0775	0,286
Spearman r	0,0797	0,272
Kendall tau	0,0536	0,27

Tabla 9. Correlación entre log(tiempo de emisión) y log(número medio de espectadores) según datos propios

4.4 ¿Los grandes, más grandes?

Durante el proceso de revisión bibliográfica encontramos una referencia de especial interés dentro del campo del análisis de redes. Jung y Kin, en su trabajo de 2021, planteaban un modelo dinámico de red para estudiar el efecto de "el rico se hace más rico", más conocido como el efecto Matthew (Hohyun Jung, 2021). Este término, acuñado en 1968 por el sociólogo estadounidense Robert K. Merton, expresa el fenómeno de acumulación de fama, poder o capital en personas que ya experimentaban altos niveles de ellos. En su trabajo de 1968, Merton emplea este término para explorar como científicos con un cierto renombre o estatus en la esfera académica, son proclives a continuar acumulando reconocimiento independientemente de la calidad de sus aportaciones a la literatura, promoviendo así una mayor concentración de los recursos y talento disponible (Merton, 1968).

Salvando las evidentes diferencias, el efecto Matthew guarda algunas similitudes con la conexión preferencial empleada en diferentes modelos de generación de redes aleatorias, ampliamente conocido por el modelo Barabási-Albert (Albert-Laszlo Barabasi, 1999). En este modelo se genera una red a partir de conectar nuevos nodos a otros nodos ya presentes en la red con una probabilidad directamente proporcional al grado de estos últimos. Es decir, aquellos nodos ya existentes en la red y que cuentan con grado alto, serán más proclives a continuar aumentando su grado. Esta dinámica garantiza que la distribución del grado de la red sigue una ley de potencias, similar al que exhiben las redes formadas por los seres humanos.

Aun siendo estos dos fenómenos fundamentalmente diferentes y aplicables a campos muy diferentes del conocimiento, este efecto plantea cuestiones interesantes dentro del análisis aplicado a redes sociales. En 2014, Kondor et al. analizaron todas las transacciones hasta el momento en Bitcoin y concluyeron que la conexión preferencial gobierna el crecimiento de la red a la vez que se muestra una alta correlación entre el grado entrante de los nodos y su capital (Kondor D, 2014).

Resulta de interés comprobar si nuestro subconjunto de usuarios de Twitch, los *streamers* más grandes de la comunidad hispanohablante, en efecto reproducen este efecto. Aunque nuestra muestra sea una selección de los 200 *streamers* con más seguidores, ya encontramos grandes diferencias entre ellos (consultar Figura 2) y por tanto puede ser un buen punto de partida para evaluar si efectivamente se reproduce este efecto.

Para evaluar el crecimiento del número de seguidores de nuestra muestra, compararemos sus números de seguidores al inicio del proceso y al finalizarlo, obteniendo los datos del mismo portal que al comienzo (Boyd, 2022). Para cada uno de nuestros *streamers* seleccionados contaremos con el valor inicial de seguidores, el actual y calcularemos su diferencia. En la Tabla 10 podremos observar, mediante un resumen de cinco números (valores mínimos, máximos y 3 cuartiles), la notable diferencia en magnitud de los valores de inicio y fin respecto a la diferencia. Deberemos, por tanto, tener en cuenta esta diferencia de magnitudes a la hora de realizar un análisis de correlaciones, centrado en este caso entre la cantidad inicial de seguidores y la diferencia.

	Inicio	*Fin*	*Diferencia*
Mínimo	$1{,}59 * 10^5$	$1{,}6 * 10^5$	−9.000
25%	$2{,}21 * 10^5$	$2{,}36 * 10^5$	3.000
50%	$3{,}47 * 10^5$	$3{,}51 * 10^5$	12.000
75%	$7{,}53 * 10^5$	$7{,}74 * 10^5$	29.500
Máximo	$1{,}19 * 10^7$	$1{,}25 * 10^7$	680.000

Tabla 10. Resumen de cinco números (mínimo, tres quartiles y máximo) de las tres variables a estudiar. Elaboración propia.

Debido a esta diferencia de magnitudes es necesario ajustar el número inicial de seguidores empleando un logaritmo neperiano. Al calcular diferentes coeficientes de correlación, se hace evidente que existe una correlación positiva entre las variables de número inicial de seguidores y la diferencia. Esta correlación positiva oscila entre el 0,41 y el 0,68 según el coeficiente empleado. Estos resultados confirman la existencia del fenómeno por el cual aquellos creadores que más crecen son aquellos que ya son más grandes, encontrando una cola de *streamers* con pocos seguidores y poco crecimiento seguido de un reducido número de usuarios con muchos seguidores y mucho crecimiento (Figura 19 y Tabla 11).

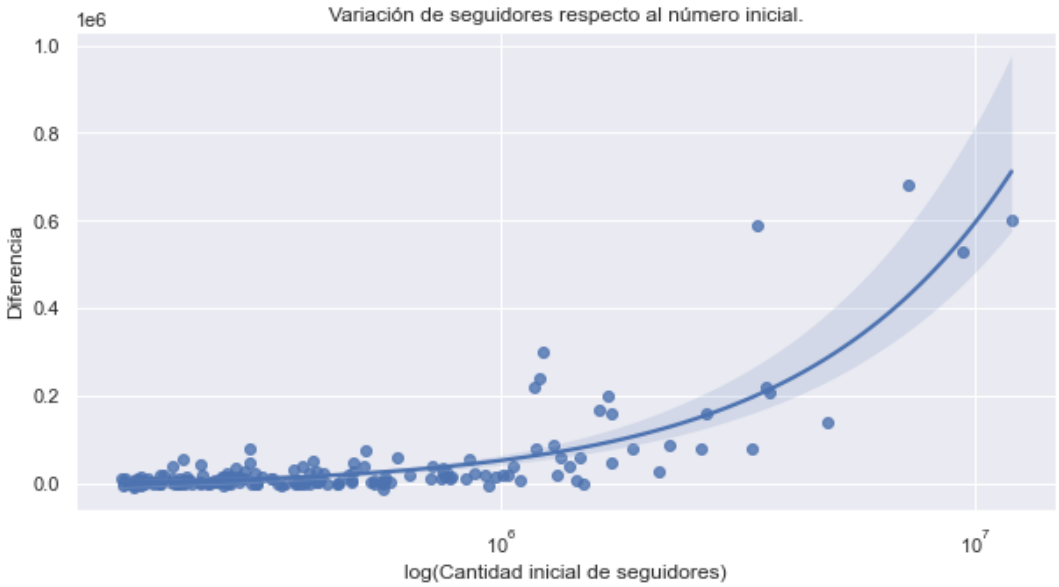

Figura 19. Correlación entre la variación del número de seguidores y la cantidad inicial. Interpolación lineal con el eje X en escala logarítmica. Punto separador decimal. Elaboración propia.

Coeficiente	**Valor**	**p-value**
Pearson r	0,685	$1{,}32 * 10^{-25}$
Spearman r	0,562	$5{,}25 * 10^{-16}$
Kendall tau	0,41	$1{,}61 * 10^{-15}$

Tabla 11. Coeficientes de correlación y sus p-valores

4.5 Conclusiones

Esta sección nos ha permitido comprobar diferentes aspectos de la plataforma. De igual forma que las grandes cifras de audiencias y seguidores se acumulan en pocos *streamers,* unas pocas categorías acumulan la mayoría del tiempo emitido. Estas categorías corresponden tanto a videojuegos populares como a categorías no relacionadas directamente con los videojuegos, como el Just Chatting. Esta categoría cuenta con 1.48 veces más espectadores totales que la segunda, el videojuego de 2013 de Rockstar Games, Grand Theft Auto 5. Es interesante comentar que en los últimos meses ha ganado gran popularidad utilizar este videojuego en su versión multijugador como un marco en el que desarrollar *roleplay:* modalidad de juego en la que los jugadores interpretan un personaje con una personalidad e historia concretos que interactúa con personajes de otros jugadores. De esta forma, el contenido emitido por estos *streamers* de GTA V podría asimilarse a una especie de historia, serie o telenovela que se desarrolla en el contexto de un videojuego, siendo los personajes de esta los avatares virtuales interpretados por los *streamers*.

A diferencia de muchas otras características, el tiempo que cada *streamer* ha emitido durante el transcurso de este estudio sigue una distribución mucho más suave que otros aspectos de la plataforma analizados hasta ahora. Aunque encontramos unas pocas excepciones, esta distribución nos indica que la mayoría de *streamers* emiten una cantidad similar de horas. En alguna ocasión hemos encontrado *streamers* que han realizado emisiones extensibles o de larga duración a modo de reto que han podido alterar mínimamente el análisis general. Sin embargo, este tipo de emisiones son relativamente comunes y no deben desestimarse como *valores atípicos,* sino que merecen su propia exploración para entender como responde la audiencia a estos directos tan largos y como interactúa el *streamer* para lograr mantener la atención y el interés de sus espectadores.

Respecto al factor temporal de las emisiones, analizándolo de manera agregada según los días de la semana, no se han extraído conclusiones especialmente relevantes. En cambio, al agregar estos datos según hora del día emerge un patrón más significativo que nos permite observar claramente las franjas horarias en las que más contenido es emitido y consumido. Resultaría de interés ampliar este análisis realizando una agregación tanto horaria como por día de la semana para explorar si aparecen otros patrones, por ejemplo, una clara diferencia entre consumo y emisión horaria según si son días laborables o de fin de semana. Paralelamente, no se ha alcanzado ninguna conclusión relevante al analizar el número de emisiones de manera diaria durante todo el periodo de análisis. De realizarse un futuro estudio de mayor dimensión y que pudiera extenderse durante la duración de varios meses o un año completo, sería muy constructivo poder analizar las variaciones a lo largo del tiempo según periodos festivos, estacionales o en función eventos significativos, tanto internos como externos a la plataforma.

Por último, podemos concluir que durante el tiempo de estudio aquellos *streamers* que contaban con un mayor número de seguidores han sido precisamente

los que más han crecido. Esto nos puede indicar que Twitch es una plataforma en la que se reproduce el efecto Mathew y que merece la pena indagar en mayor profundidad la evolución del número de seguidores de un *streamer* dado: ¿cómo consiguen crecer los usuarios de la plataforma? ¿Un incremento de seguidores se traduce en una mayor audiencia? ¿Influye las horas de emisión o el tipo de categorías?

5. Agrupación de *streamers* por contenido

Gracias a la resolución del conjunto de datos, podemos caracterizar de manera precisa cada uno de los *streamers* de nuestra muestra según el número de capturas de datos que se han registrado emitiendo una determinada categoría. Las categorías son los tipos de contenido que se pueden emitir en Twitch. Los *streamers* fijan una categoría para una emisión o fragmento de ella para que los posibles espectadores sepan que contenido pueden consumir. Estas categorías pueden incluir videojuegos (*Counter-Strike: Global Offensive, League of Legends, Valorant*...) pero también una gran diversidad de actividades no relacionadas con los videojuegos (Música, Aire libre y viajes, Política, ...).

Esta caracterización nos permitirá llevar a cabo una clusterización en la que exploraremos la existencia de comunidades de *streamers* según el contenido que suelen emitir. Más adelante podremos comparar estas agrupaciones con las que puedan emerger al construir y analizar la red de *streamers* según audiencia compartida. Para llevar a cabo este proceso planteamos dos posibles alternativas resumidas en la Figura 20.

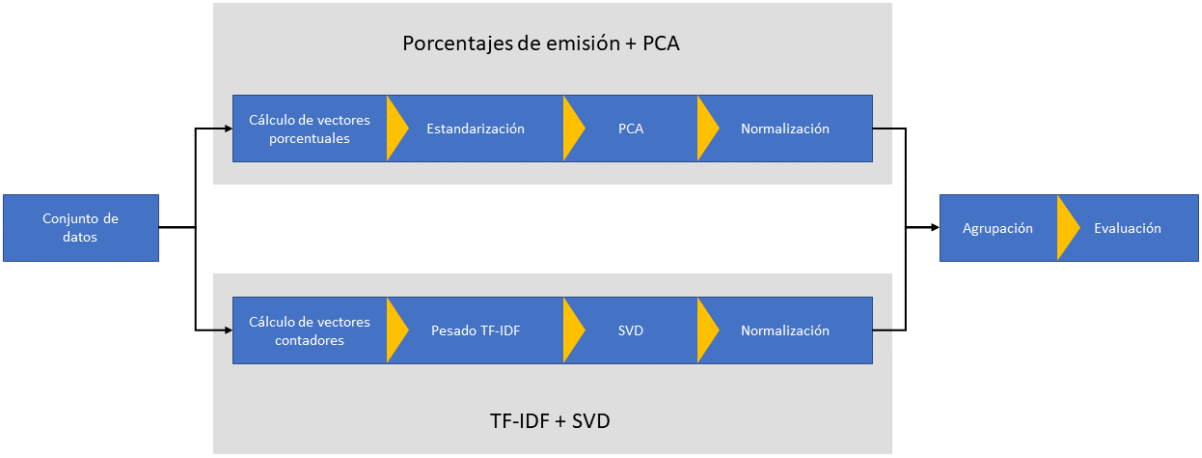

Figura 20. Diagrama de alternativas para el proceso de agrupación

5.1 Vectorización

Planteamos dos estrategias diferentes para llevar a cabo la vectorización de cada *streamer*, es decir, la obtención de vectores que representen las características de cada uno de ellos en un espacio vectorial determinado. La primera aproximación que seguiremos será calcular un vector v que recogerá la proporción del tiempo que ha dedicado a cada streamer a una determinada categoría durante el tiempo que ha emitido en directo. Definiremos $\mathcal{C}$ como el conjunto de categorías detectadas en nuestro

conjunto de datos: $\mathcal{C} = \{Sports,\ Genshin\ Impact,\ ASMR, \dots\}$ ($|\mathcal{C}| = 370$) y $\mathcal{S}$ como el conjunto de *streamers* seleccionados. Seguidamente definiremos los vectores para cada uno de los *streamers*:

$$f(s,c) = |capturas\ del\ streamer\ s\ emitiendo\ la\ categoría\ c|$$
$$h(s) = |capturas\ del\ streamer\ s|$$
$$v_i = \frac{1}{h(s)}\ \{f(s,c) \mid \forall\ c \in \mathcal{C}\} \qquad s \in \mathcal{S}, \qquad v_i \in \mathbb{R}^{|\mathcal{C}|}$$

Ecuación 3. Vectorización por contenido

Nos encontramos por tanto con una representación en un espacio vectorial de una relativa alta dimensionalidad, superando el número de dimensiones al número de muestras. Así mismo, debemos destacar que es una representación muy dispersa, siendo de media 6 las componentes del vector diferentes a cero para cada *streamer*. Por tanto, nos encontramos en una situación en la que, previsiblemente, será beneficioso aplicar una reducción de dimensionalidad a estos vectores para mejorar los resultados de las tareas desarrolladas más posteriormente.

El segundo método de vectorización que decidimos emplear viene prestado por el campo del procesamiento de lenguaje natural. El pesado TF-IDF, de las siglas en inglés para *term-frequency times inverse document-frequency,* es una técnica empleada para la obtención de vectores que representen documentos de un corpus. Al aplicar este pesado, se tiene en cuenta cómo de común es cada termino. Así se pueden ajustar los mejor vectores para tener en cuenta *tokens* como las palabras vacías o *stopwords*, palabras muy frecuentes en un idioma y que realmente no aportan mucha información, como para el castellano podrían ser las preposiciones o los artículos determinantes.

La motivación para emplear esta técnica de pesado viene dada por la gran presencia de unas pocas categorías de manera transversal a todo el conjunto de *streamers,* como podría ser la categoría de Just Chatting. Siguiendo la analogía con el campo del PLN cada *streamer* puede ser considerado como un documento compuesto por la sucesión de las categorías que ha emitido y así podemos considerar las categorías muy frecuentes para todos los *streamers* como palabras vacías. Aplicando este tipo de pesado, podemos dar un valor adecuado para categorías que podrían aportar información clave a la hora de realizar agrupaciones que quizás de otro modo podrían quedar ocultas por un incorrecto pesado que no considerara la frecuencia de las categorías para el conjunto completo de *streamers*. Definimos el pesado TF-IDF de la siguiente forma[12]:

$$tfidf(c,s) = tf(c,s) * idf(c) = f(s,c) * \log\frac{1+n}{1+df(c)} + 1$$
$$n = |\mathcal{S}|$$
$$df(c) = |\{\ 1 \mid f(s,c) \neq 0, \forall s \in \mathcal{S}\}|$$

Ecuación 4. Definición del pesado TF-IDF

[12] Cabe destacar que hay múltiples definiciones y que aquí describimos la implementada en *scikit-learn* (Pedregosa, 2011), la librería utilizada para llevar a cabo los cálculos.

Al aplicar este tipo de pesado también obtenemos una representación muy dispersa, por lo que al igual que en el esquema anterior, será necesario aplicar una reducción de dimensionalidad para obtener buenos resultados a la hora de realizar las agrupaciones.

5.2 Reducción de dimensionalidad

5.2.1 PCA

Para reducir el número de dimensiones de los espacios vectoriales de representación se opta por aplicar dos técnicas reducción de dimensionalidad: PCA para el pesado por porcentaje de tiempo emitido y SVD para el pesado TFIDF. PCA, por sus siglas en inglés para *Principal Component Analyisis*, fue inventada en 1901 por Karl Pearson (F.R.S., 1901) aunque de manera paralela e independiente también fue desarrollada por Harold Hotelling en la década de los 30 (Hotelling, 1936). PCA se basa en la proyección de los puntos en el espacio vectorial de origen a otro espacio de menor dimensión en el que su base está formada por los autovectores de la matriz de covarianza. Ordenando los autovectores según su autovalor, obtendremos los componentes principales de nuestro conjunto de datos.

Los componentes principales con mayor autovalor son aquellos que recogen una mayor cantidad de varianza de los datos originales. Es importante destacar que antes de aplicar PCA es necesario llevar a cabo un proceso de estandarización, es decir, transformar el conjunto de vectores de forma que la media es 0 y la varianza es unitaria. Seleccionando un número reducido de componentes principales respecto de la cantidad total podemos proyectar los vectores a un espacio de dimensionalidad mucho menor y conservando gran parte de la varianza, es decir, sin perder mucha información. En el caso que nos concierne, seleccionando las primeras 92 componentes conservamos más del 95% de la varianza (Figura 21). Con un ~24% de las dimensiones originales podemos capturar gran parte de la información, sin embargo, un espacio vectorial de 92 dimensiones sigue siendo bastante grande.

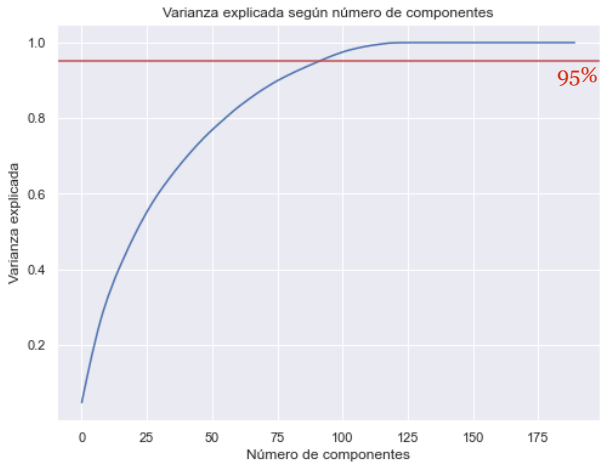

Figura 21. Varianza explicada según número de componentes (PCA). Elaboración propia.

5.2.2 DVS

En cuanto al segundo pesado, empleando TF-IDF, optaremos por aplicar una reducción de dimensionalidad basada en DVS o descomposición en valores singulares. (*Singular Value Decomposition*). Esta es una técnica con muchas aplicaciones en el campo de la estadística en la que una matriz dada se descompone empleando otras tres matrices, es decir, se expresa una matriz como una factorización de otras tres, de la forma: $A = U\Sigma V^t$ de forma que $U \in \mathbb{R}^{m\,x\,m}, V \in \mathbb{R}^{n\,x\,n}$ $y\ \Sigma \in \mathbb{R}^{m\,x\,n}$, si $A \in \mathbb{R}^{m\,x\,n}$ se garantiza que U y V son matrices ortonormales, es decir, sus vectores fila y columna son ortonormales entre sí.

La matriz Σ recoge en su diagonal los valores singulares de la matriz A, obtenidos mediante el cálculo de los autovalores y autovectores de la matriz. Si reducimos la matriz Σ de forma que solamente conservemos los k valores singulares mayores y sustituyamos el resto por ceros, obtendremos una matriz reconstruida $\tilde{A}$ cuyo rango viene determinado por el parámetro k, de manera que habremos obtenido una aproximación a la matriz original de menor dimensionalidad. Este procedimiento también se conoce como el teorema de Eckart-Young-Mirsky (Eckart, 1936; Mirsky, 1960).

Al aplicar esta técnica encontramos que bastan 46 componentes para explicar un 95% de la varianza original, exactamente la mitad de los requeridos en la aproximación de pesado porcentual seguido de PCA. A diferencia de PCA, al aplicar DVS no es necesario estandarizar las muestras ya que DVS no necesita que estas estén centradas respecto al origen (Figura 22).

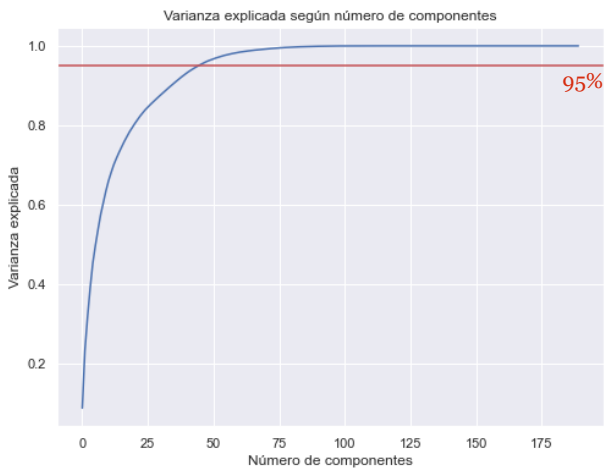

Figura 22. Varianza explicada según número de componentes (DVS). Punto separador decimal. Elaboración propia.

5.3 Métodos de agrupación y evaluación

Para realizar la clusterización, hemos seleccionado tres técnicas de las más utilizadas. Evaluaremos estas técnicas mediante 3 métricas que nos permitirán escoger qué

método y parámetros son los más adecuados para nuestro caso particular. A continuación, se realizará una breve explicación de las tres técnicas de clusterización:

- K-means: también denominado k-medias por su traducción al castellano, es un método de clusterización que realiza una partición de las muestras en k conjuntos. Una muestra pertenece al conjunto cuya media le es más cercana, de aquí su nombre. El término k-means fue empleado por primera vez en 1967 por James MacQueen (MacQueen, 1967) aunque la idea original es atribuida a Hugo Steinhaus (Steinhaus, 1957). K-means lleva a cabo la partición de las observaciones $\{x_1, x_2, \dots, x_n\} \subset \mathbb{R}^D$ en k conjuntos con el objetivo de minimizar la suma de cuadrados dentro de cada conjunto C_i:

$$argmin_S \sum_{i=1}^{k} \sum_{x_j \in C_i} ||x_j - \mu_i||^2$$

$$\mu_i = \frac{\sum_{x_j \in C_i} x_j}{|C_i|}$$

Ecuación 5. Minimización de la suma de cuadrados de las agrupaciones

- DBScan: método de agrupación basado en densidad. Propuesto por M. Ester, H. Kriegel et al. en 1996 se basa en la agrupación de los puntos según cómo de compactamente se encuentran distribuidos en el espacio de representación, agrupando aquellos que cuentan con muchos puntos cercanos y marcando como atípicos o *outliers* aquellos que se encuentran aislados en zonas de baja densidad (Martin Ester, 1996). Una gran ventaja que plantea esta aproximación en comparación con *k-means* es que el número de clusters se detecta automáticamente durante el proceso de partición y, por tanto, no necesitamos conocer previamente el número de grupos existentes. Es importante destacar que este método cuenta con dos parámetros: ε, distancia máxima para la cual un punto se considera o no vecino de otro; y $minPts$, número mínimo de vecinos que debe tener un punto para poder considerarlo como núcleo de un posible grupo.

- OPTICS (*Ordering points to identify the clustering structure*): es otro algoritmo basado en densidad que identifica automáticamente los grupos presentes en un conjunto de datos. Desarrollado por Ankerst, Breunig et al. en 1999, sigue un planteamiento similar al propuesto por DBScan, pero se centra en atajar el mayor problema planteado por este método, y es encontrar agrupaciones significativas en conjuntos de datos con densidades variables (Mihael Ankerst, 1999). Al igual que DBScan, OPTICS requiere de los parámetros ε y $minPts$. La gran diferencia es que OPTICS relaja ε, pasando de un solo valor a un rango de distancias, siendo ε el límite de este rango.

A la hora de poder evaluar el resultado de estos métodos, se nos plantea un problema típico de las tareas de aprendizaje automático no supervisado. Desconocemos las verdaderas agrupaciones subyacentes en el conjunto de datos, si es que las hay, y es por ello que no tenemos una referencia fidedigna que nos pueda indicar si una muestra esta correctamente clasificada en una u otra agrupación. Es por ello que, por ejemplo, no podemos utilizar técnicas como las basadas en

métricas de Información Mutua y debemos recurrir a métricas específicamente pensadas para este tipo de tareas.

La primera métrica que consideraremos es el coeficiente de Silhouette (Rousseeuw, 1987). Este coeficiente toma valores en el rango $[-1, 1]$, relacionándose valores altos de este coeficiente con agrupaciones bien definidas, mientras que valores cercanos al 0 indican agrupaciones con solapes. Esta métrica desarrollada por Rousseeuw en 1987 se calcula muestra a muestra y posteriormente se calcula la media para obtener el coeficiente a nivel general. Asumiendo que hemos llevado a cabo un proceso de agrupación y que hemos obtenido los siguientes conjuntos $\{C_1, C_2, \dots, C_k\}$ definimos el coeficiente de Silhouette como:

$$a(i) = \frac{1}{|C_I| - 1} \sum_{j \in C_I, i \neq j} d(i,j)$$

$$i \in C_I, \qquad d(i,j) \equiv distancia\ entre\ los\ puntos\ i, j$$

Ecuación 6. Coeficiente de Silhouette: distancia media de un punto contra todos los otros puntos del mismo conjunto.

$$b(i) = min_{J \neq I} \frac{1}{|C_I|} \sum_{j \in C_J} d(i,j)$$

Ecuación 7. Coeficiente de Silhouette: distancia media mínima de un punto a todos los puntos de otro conjunto

$$s(i) = \begin{cases} \dfrac{b(i) - a(i)}{\max(\mathrm{a(i)}, \mathrm{b(i)})} & si\ |C_I| > 1 \\ 0 & si\ |C_I| = 1 \end{cases}$$

Ecuación 8. Coeficiente de Silhouette para una muestra

La segunda métrica que utilizaremos será el índice de Calinski-Harabasz (Harabasz, 1974). Este índice se calcula como la ratio de la dispersión entre conjuntos sobre la dispersión media de los conjuntos, definiendo la dispersión como la suma de distancias al cuadrado. Al igual que en el coeficiente de Silhouette, esta es una medida de cohesión-separación (dispersión media de los conjuntos y la dispersión entre conjuntos respectivamente). Valores elevados de esta métrica indican agrupaciones bien definidas.

La última métrica que emplearemos es el índice de Davies-Bouldin (Davies, 1979), la cual indica, para valores cercanos al 0, una correcta separación entre grupos. Intuitivamente, el índice representa la *similitud* media entre conjuntos, siendo la similitud una medida que compara la distancia entre conjuntos con sus propios tamaños. Habiendo agrupado un conjunto de muestras en k grupos $\{C_1, C_2, \dots, C_k\}$, definimos este índice como:

$$s_i = distancia\ media\ entre\ los\ puntos\ y\ el\ centroide\ de\ un\ grupo$$
$$d_{i,j} = distancia\ entre\ los\ centroides\ de\ los\ grupos\ i\ y\ j$$
$$DB = \frac{1}{k}\sum_{i=1}^{k} max_{i \neq j} \frac{s_i + s_j}{d_{i,j}}$$

Ecuación 9. Índice de Davies-Bouldin

5.4 Resultados

Para obtener los mejores resultados, aplicaremos los tres métodos vistos anteriormente y exploraremos exhaustivamente sus parámetros, probando diferentes combinaciones. Posteriormente calcularemos el coeficiente de Silhouette y los índices de Calinski-Harabasz y Davies-Bouldin. Evaluaremos los resultados en base a estas métricas y decidiremos cual es la mejor agrupación para, posteriormente, analizar las conclusiones que puedan emerger.

En el caso de k-medias el modelo solamente necesita el parámetro k, el número de grupos, y decidimos explorar los siguientes valores: $[2, 3, \dots, 10]$. Para DBScan y OPTICS deberemos decidir los valores de ε y $minPts$, para ello exploraremos exhaustivamente las combinaciones posibles entre $\varepsilon \in [0{,}1, 0{,}2, \dots, 0{,}9, 1]$ y $minPts \in [2, 4, 6, 8, 10, 15, 20]$. Además, dado que para DBScan y OPTICS podemos especificar la función utilizada para mediar la distancia entre los puntos, exploraremos también los resultados entre emplear las distancias L2 y la distancia coseno (Ecuación 10).

$$distancia\ coseno(v, w) = 1 - \frac{v * w^t}{||v|| * ||w||}$$

Ecuación 10. Distancia coseno

Para la primera aproximación se han realizado 289 experimentos, 33 de ellos dando como resultado una sola agrupación, siendo por tanto descartados. Cada experimento, combinación de un método y sus parámetros correspondientes, se ha repetido 10 veces, con el fin de poder promediar los resultados de las métricas y calcular sus desviaciones estándar. Respecto a la segunda aproximación se han realizado el mismo número de experimentos, siendo 9 de ellos descartados por dar lugar a una sola agrupación.

5.4.1 Resultados: Porcentajes + PCA

La Tabla 12 muestra los 5 mejores experimentos para cada una de las métricas (celdas sombreadas). Los valores de dichas métricas han sido redondeados a 3 decimales para mejorar la legibilidad. Se han omitido los valores para las desviaciones

estándar ya que estos son muy bajas, todos los métodos han resultado muy robustos, devolviendo resultados casi idénticos para todas las ejecuciones.

Método	*minPts* / k	ε	Distancia	Grupos	Silhouette	Calinski-Harabasz	Davies-Bouldin
OPTICS	2	0,9	cos	34	0,349	5.,307	1,426
OPTICS	2	1	cos	33	0,347	5,37	1,404
OPTICS	2	0,6	cos	31	0,345	5,5	1,36
OPTICS	2	0,7	cos	31	0,345	5,5	1,36
OPTICS	2	0,8	cos	31	0,345	5,5	1,36
DBSCAN	20	0,8	L2	2	0,072	16,862	1,721
DBSCAN	20	0,3	cos	2	0,075	16,845	1,509
DBSCAN	15	0,2	cos	2	0,07	16,707	1,444
OPTICS	15	0,2	cos	2	0,07	16,707	1,444
DBSCAN	15	0,6	L2	2	0,073	15,895	1,145
OPTICS	8	0,1	L2	2	0,056	13,21	1,023
DBSCAN	10	0,1	L2	2	0,056	13,21	1,023
OPTICS	10	0,1	L2	2	0,056	13,21	1,023
DBSCAN	8	0,1	L2	2	0,056	13,21	1,023
OPTICS	6	0,1	L2	3	0,045	10,238	1,052

Tabla 12. Mejores resultados para cada métrica utilizando Porcentajes + PCA

Los resultados obtenidos son muy dispares y no podemos extraer conclusiones de manera inmediata. Aquellos experimentos que han obtenido buenos resultados para una métrica, han obtenido resultados mediocres para el resto de ellas. Por ejemplo, los cinco experimentos que obtienen los mejores resultados para el índice de Calinski-Harabasz presentan valores para Silhouette muy cercanos a cero, es decir, según Silhouette esas agrupaciones presentan *clusters* con mucho solape.

Destaca que no aparezca k-means como buena alternativa a OPTICS o DBSCAN, esto puede indicarnos que los *clusters* no siguen formas hiperesféricas, las cuales suelen producir muy buenos resultados con k-means. OPTICS obtiene los mejores resultados en Silhouette, con un valor bajo para el parámetro *minPts* y valores elevados para ε, produciendo agrupaciones con un número muy elevado de grupos, entre 31 y 34. En cambio, para el índice de Calinski-Harabasz predomina DBSCAN, con valores elevados para el parámetro *minPts,* produciendo agrupaciones con solamente dos grupos. En cuanto al índice de Davies-Bouldin encontramos una presencia más repartida entre OPTICS y DBSCAN, produciendo estos experimentos, de nuevo, agrupaciones con muy pocos grupos.

Para poder determinar de manera más robusta qué experimento genera los mejores resultados, decidimos buscar alguna forma de harmonizar las tres medidas y obtener un solo índice. Decidimos calcular, para cada uno de los experimentos, los percentiles de sus métricas de evaluación respecto al resto de todos los resultados de esa métrica. Posteriormente, calculamos la media de estos tres percentiles por cada experimento, obteniendo así un indicador de como de superior es, de media, un experimento dado respecto del resto, realizando un compromiso entre las tres métricas de evaluación (Tabla 13). Para poder comparar los percentiles de las tres métricas correctamente se debe realizar un paso intermedio, calculando la inversa del índice de Davies-Bouldin, ya que este se valora de manera descendente, contrariamente al resto de métricas.

Método	$minPts$ / k	ε	Dist.	Grupos	Silhouette		Calinski-Harabasz		Davies-Bouldin		$\overline{Per}$
					V.	Per.	V.	Per.	V.	Per.	
OPTICS	6	0,6	L2	4	0,13	**69,55**	14,31	**77,85**	1,25	**65,05**	**70,82**
OPTICS	6	0,7	L2	4	0,12	**65,4**	14,05	**73,88**	1,22	**69,72**	**69,67**
OPTICS	10	0,7	L2	4	0,13	**70,93**	14,56	**84,26**	1,33	**52,42**	**69,2**
OPTICS	10	0,9	L2	4	0,13	**70,24**	14,39	**80,62**	1,31	**55,36**	**68,74**
OPTICS	10	0,8	L2	4	0,13	**70,24**	14,39	**80,62**	1,31	**55,36**	**68,74**

Abreviaturas: V – valor de la métrica; Per. – percentil de la métrica; $\overline{Per}$ – media de los percentiles.

Tabla 13. Mejores agrupaciones Porcentajes + PCA según el percentil medio de las métricas de evaluación

Procediendo de esta forma, obtenemos resultados mucho más coherentes: los cinco experimentos con una media de percentiles más alta corresponden todos al método OPTICS y combinaciones de parámetros que generan cuatro grupos. Podemos observar como cada uno de estos experimentos no obtiene resultados extraordinarios en ninguna métrica. Sin embargo, al promediar los percentiles, obtenemos una representación de cómo de superior es de media dicho experimentos respecto del resto. Pasamos de métricas que nos indican cómo de buena es en sí misma una agrupación para valorar los experimentos mediante una métrica comparativa o relativa, es decir, nos indica cómo de buena es una agrupación en relación al resto.

Estos cinco mejores experimentos producen las mismas agrupaciones, por lo que nos limitaremos a analizar el primero de ellos. Para poder caracterizar cada uno de los grupos producidos, emplearemos diferentes variables: número de *streamers* que lo componen, categorías con un tiempo de emisión superior a un valor de referencia (5% en este caso) y las tres categorías con mayor peso. Obtenemos estas dos últimas variables considerando el centroide del grupo, es decir, promediando todas las muestras que componen el grupo, para obtener una muestra ficticia ideal o representativa del resto.

En la Tabla 14 encontraremos la agrupación producida por el mejor experimento según la media de percentiles. Nos encontramos con cuatro grupos, uno de ellos, el etiquetado como -1, corresponde a lo que los algoritmos de agrupación por densidad denominan como ruido, el grupo de aquellas muestras que no forman parte de alguno de los otros grupos.

Grupo	Tamaño	Categorías con más de un 5% de media	Categorías con más peso	Peso
0	17	1	Just Chatting	0,980
			FIFA 22	0,008
			Slots	0,007
1	13	2	Fortnite	0,856
			Just Chatting	0,090
			VALORANT	0,034
2	12	2	League of Legends	0,863
			Just Chatting	0,073
			Teamfight Tactics	0,025
-1	149	4	Just Chatting	0,243
			Fortnite	0,121
			VALORANT	0,072

Tabla 14. Resultados de la mejor agrupación utilizando Porcentajes + PCA

En esta agrupación encontramos tres grupos de tamaño similar y en los que una sola categoría acumula la mayor parte del peso, en este caso el porcentaje de tiempo de emisión. El último grupo, en cambio, destaca por su tamaño (acumula el ~78% de los *streamers*) y no hay una categoría que acumule la mayor parte del peso. Los tres primeros grupos los denominaremos grupos especializados, son grupos con una cantidad reducida de *streamers* en los que una única categoría acumula la mayor parte del tiempo de emisión. Al último grupo lo denominaremos de variedad, ya que se presenta como una amalgama mucho más heterogénea de *streamers* y de contenidos.

De esta agrupación destacan diferentes cuestiones, la primera de ellas siendo el reducido número de grupos generados, dado lo amplia que es la muestra tanto en número de *streamers* como en el período temporal de captura, es sorprendente que solamente haya detectado 3 grupos estrictamente hablando. Por otro lado, quizás por este reducido número de grupos, la mayoría de categorías predominantes corresponden a videojuegos bastante populares actualmente. Sin embargo, en todos los grupos encontramos la categoría Just Chatting, ya bien como categoría predominante (grupos 0 y -1) o bien como categoría secundaria (grupos 1 y 2).

La diversidad de contenidos dentro de cada grupo parece bastante limitada, en los grupos especializados encontramos a lo sumo 2 categorías con un peso mayor al 5%, mientras que para el grupo de variedad encontramos 4. En los primeros, como ya hemos comentado anteriormente, una sola categoría acumula el grueso del tiempo de emisión. Sin embargo, en el grupo de variedad esta situación se invierte, teniendo que competir muchas categorías por el tiempo de emisión. Nos encontramos de nuevo con un reparto desigual, las dos categorías principales acumulan el 24% y el 12% del tiempo de emisión mientras que una larga cola de categorías se reparten el tiempo restante y por tanto a cada una de ellas les corresponde una fracción pequeña del total.

El escaso número de grupos y el gran tamaño del grupo de variedad nos indica que probablemente existan otros grupos que esta aproximación no ha sido capaz de detectar.

5.4.2 Resultados: TFIDF + DVS

De manera idéntica a la sección anterior, comenzaremos explorando los experimentos que obtienen los mejores resultados para cada una de las métricas en la Tabla 15. Nuevamente se han omitido los valores para las desviaciones estándar de las métricas, ya se han obtenido variaciones mínimas al producir todos los métodos resultados muy robustos.

Método	$minPts$ / k	ε	**Distancia**	**Grupos**	**Silhouette**	**Calinski-Harabasz**	**Davies-Bouldin**
OPTICS	4	1	cos	13	0,445	19,823	1,542
OPTICS	4	0,9	cos	13	0,445	19,823	1,542
OPTICS	4	0,6	cos	13	0,445	19,823	1,542
OPTICS	4	0,8	cos	13	0,445	19,823	1,542
OPTICS	4	0,7	cos	13	0,445	19,823	1,542
KMEANS	4	-	-	4	0,24	27,966	1,629
KMEANS	3	-	-	3	0,181	27,395	1,898
KMEANS	5	-	-	5	0,269	26,43	1,71
KMEANS	2	-	-	2	0,121	26,016	1,823
OPTICS	15	0,2	cos	4	0,236	25,814	1,485
OPTICS	2	0,1	L2	14	-0,0	6,14	0,96
DBSCAN	2	0,1	L2	13	0,07	7,71	0,96
OPTICS	2	0,2	L2	17	0,08	6,79	0,98
DBSCAN	2	0,2	L2	16	0,17	8,6	0,99
DBSCAN	4	0,1	L2	7	0,06	10,87	1,03

Tabla 15. Resultados de los experimentos de agrupación

Los resultados obtenidos son bastante dispares en cuanto al número de grupos y no se puede extraer una conclusión evidente a simple vista. OPTICS obtiene los mejores resultados para el coeficiente Silhouette, para el índice Calinski-Harabasz lo hace K-means mientras que para el Davies-Bouldin encontramos tanto OPTICS como DBSCAN. OPTICS obtiene buenos resultados en Silhouette fijando el número mínimo de puntos a 4 y valores altos de épsilon. K-means solo aparece como una buena opción si consideramos el índice de Calinski-Harabasz y considerando valores pequeños para el parámetro k. Estos experimentos presentan los mejores resultados para cada una de las métricas. Sin embargo, en las otras dos restantes rinden de manera mediocre.

Para salvar estas disparidades y ofrecer resultados más coherentes procederemos de igual manera a lo expuesto en la sección anterior. Para cada experimento calcularemos el percentil medio de sus métricas para obtener una medida unificada que nos indique cómo de buena es una agrupación en relación al resto (Tabla 16). Posteriormente, analizaremos los resultados producidos por la mejor combinación de modelo y parámetros.

Método	$minPts$ / k	ε	**Dist.**	**Grupos**	**Silhouette**		**Calinski-Harabasz**		**Davies-Bouldin**		$\overline{Per}$
					V.	**Per.**	V.	**Per.**	V.	**Per.**	
OPTICS	8	0,2	cos	6	0,31	**78,89**	23,67	**82,7**	1,45	**58,82**	**73,47**
OPTICS	4	0,3	cos	12	0,43	**94,46**	20,26	**59,17**	1,43	**61,59**	**71,74**
OPTICS	8	0,6	L2	6	0,27	**71,63**	23,75	**83,04**	1,46	**57,79**	**70,82**
OPTICS	4	0,4	cos	12	0,43	**94,12**	20,26	**58,82**	1,45	**59,17**	**70,7**
DBSCAN	2	0,4	L2	23	0,37	**90,66**	12,51	**34,95**	1,14	**86,16**	**70,59**

Abreviaturas: V – valor de la métrica; Per. – percentil de la métrica; $\overline{Per}$ – media de los percentiles.

Tabla 16. Mejores agrupaciones TFIDF+SVD según el percentil medio de las métricas de evaluación

El método que mejor resultado ha producido ha sido OPTICS con $minPts = 8$ y $\varepsilon = 0{,}2$ empleando la distancia coseno. Ha obtenido un percentil medio para las métricas de evaluación de 82,7. Para caracterizar cada uno de los grupos detectados podemos tomar sus centroides, es decir, la media de todos los puntos que los forman. Así podemos comprender mejor que tipo de contenido emiten los *streamers* de cada grupo (Tabla 17).

Grupo	Tamaño	Categorías con más de un 5% de media	Categorías con más peso	Valor TF-IDF
0	31	2	Just Chatting	0,911
			FIFA 22	0,056
			Fall Guys: Ultimate Knockout	0,039
1	26	4	Fortnite	0,913
			Just Chatting	0,124
			VALORANT	0,07
2	11	5	VALORANT	0,906
			Fortnite	0,128
			Just Chatting	0,105
3	12	6	Minecraft	0,855
			Just Chatting	0,185
			Fortnite	0,112
4	15	2	League of Legends	0,943
			Just Chatting	0,107
			Teamfight Tactics	0,038
5	96	9	Just Chatting	0,177
			Ark: Survival Evolved	0,101
			Fortnite	0,083

Tabla 17. Resultados de la mejor agrupación utilizando TFIDF+DVS

Al igual que en el caso de la aproximación de porcentajes + PCA, destacan dos tipologías de grupo: los de variedad, compuestos por *streamers* que emiten gran variedad de contenido, y los especializados, aquellos en los que una categoría en

concreto acapara la mayoría del peso. Del primer tipo solamente encontramos el grupo 5, con un alto número de *streamers*: 96. Los *streamers* de este grupo emiten gran variedad de categorías, 9 de ellas con un valor considerable (> 0,05). La categoría con mayor peso es Just Chatting, con un peso relativamente bajo del 0,177. En el resto de los grupos podemos observar como la categoría predominante obtiene valores en torno al 0.9. En este caso la categoría predominante no alcanza valores tan elevados porque comparte mucho tiempo de emisión con otras categorías. La diferencia con la segunda categoría es la más pequeña de todos los grupos: 0,076.

En los grupos especializados encontramos 95 *streamers,* el ~50% del total. En estos grupos una sola categoría tiene el protagonismo de las emisiones, obteniendo pesos entre el 0,85 y el 0,94. El tamaño de estos grupos (todos excepto el -1) varía notablemente, con 11 *streamers* para el grupo 2 y con 31 para el grupo 0. Merece la pena destacar algunas curiosidades:

- La categoría Just Chatting, de traducción al castellano por "simplemente hablando", es transversal a casi todos los grupos como ya prevenimos al inicio de este proceso. Esta categoría aparece en todos los grupos como una de las categorías predominante. Una explicación a este fenómeno es que los *streamers* de videojuegos habitualmente suelen destinar un tiempo al inicio de las emisiones a charlar, interactuando normalmente con el chat, para que los espectadores y especialmente aquellos que son seguidores del canal vayan conectándose al directo (los *streamers* pueden enviar una notificación a todos sus seguidores, avisándoles de que han iniciado una emisión). También es habitual que dediquen un tiempo al final de cada emisión, entre partidas y entre diferentes videojuegos a charlar.
- Los grupos 1 y 2 presentan las mismas categorías en el listado de las tres con más peso, cada uno de ellos con categorías predominantes diferentes: Fortnite y VALORANT respectivamente. Esta similitud entre los grupos refleja el interés compartido por un conjunto de *streamers* por estos videojuegos de acción. En el grupo 3 Fortnite y VALORANT ocupan la 3ª y 4ª posición, con pesos del 0,11 y el 0,10 respectivamente. Aunque en este grupo ninguno de los dos videojuegos represente la categoría predominante la similitud en sus pesos de nuevo nos indica un interés común por estos dos videojuegos.
- Todos los grupos, tanto especializados como de variedad, cuentan con muy pocas categorías con un peso superior al 0,05. En el caso de los grupos especializados encontramos que al acumular una sola categoría gran parte del tiempo de emisión, el resto de las categorías han de repartirse una fracción muy pequeña del tiempo restante y por tanto es muy difícil que alguna de ellas logre pesos mayores. Sin embargo, en el grupo de variedad esta situación se invierte, resultando difícil llegar al 0,05 debido a la gran cantidad de categorías, no al tiempo que están acumulan.

Por último, resulta interesante destacar los resultados obtenidos con la segunda combinación de modelo y parámetros que mejor media de percentiles obtiene (Tabla 18). En esta ocasión obtenemos 12 grupos con tamaños mucho más heterogéneos, resultando en una agrupación mucho más granular. Así mismo, la diferenciación entre grupos especializados y los de variedad resulta más compleja: grupos como el 3, 7 y 10 presentan una categoría predominante con un peso relativamente elevado. Sin

embargo, el número de categorías con una media superior al valor umbral es igual o superior al grupo -1 (grupo definido por los algoritmos de densidad como aquel grupo en el que se sitúan las muestras ruido, aquellas que no han podido ser incluidas en ningún otro grupo).

Grupo	Tamaño	Categorías con más de un 0.05 de media	Categorías con más peso	Valor TF-IDF
0	27	1	Just Chatting	0,949
			FIFA 22	0,041
			The Stanley Parable	0,021
1	23	3	Fortnite	0,948
			Just Chatting	0,082
			VALORANT	0,079
2	10	5	VALORANT	0,918
			Fortnite	0,141
			Ark: Survival Evolved	0,086
3	7	8	Grand Theft Auto V	0,839
			Just Chatting	0,237
			VALORANT	0,137
4	16	3	League of Legends	0,926
			Just Chatting	0,12
			VALORANT	0,051
5	7	4	Counter-Strike: Global Offensive	0,925
			Just Chatting	0,11
			War Thunder	0,088
6	5	2	Ark: Survival Evolved	0,91
			Just Chatting	0,217
			IRONSIGHT	0,042
7	12	6	Minecraft	0,855
			Just Chatting	0,185
			Fortnite	0,112
8	5	3	Call of Duty: Warzone	0,924
			Call Of Duty: Modern Warfare	0,195
			Gran Turismo 7	0,075
9	6	3	Sports	0,754
			FIFA 22	0,321
			F1 2021	0,061
10	7	12	Elden Ring	0,722
			Just Chatting	0,222
			Kirby and the Forgotten Land	0,108
-1	66	6	Just Chatting	0,235
			Fortnite	0,131
			Ark: Survival Evolved	0,071

Tabla 18. Resultados de la segunda mejor agrupación mediante TFIDF+DVS

Además, es notable la aparición de grupos como el 8 o el 9. En el caso del grupo 8 encontramos un grupo en el que las dos categorías con mayor peso pertenecen a la saga de videojuegos de acción Call Of Duty. El grupo 9 aparece como un grupo para los deportes tradicionales, la categoría predominante es Sports, y las adaptaciones de estos deportes al mundo de los videojuegos con FIFA 22 y F1 2021 ocupando las siguientes

posiciones (ambos videojuegos son simuladores, con mayor o menor nivel de realismo, del fútbol y la Fórmula 1 respectivamente).

Ambas combinaciones de modelo y parámetros han producido resultados mucho más exhaustivos e informativos que los obtenidos con la primera aproximación. Empleando pesado TF-IDF seguido de una reducción DVS obtenemos resultados mejores en número de grupos, cohesión de tamaños y contenidos. En la primera aproximación empleando vectores de porcentajes de tiempo de emisión y una reducción de dimensionalidad PCA la gran mayoría de *streamers* simplemente eran reconocidos como ruido y se detectaban pocos grupos especializados. Por tanto, podemos concluir que en el caso que nos ocupa la segunda aproximación ha generado resultados de mayor calidad y que nos ayudan a comprender mejor las comunidades existentes de *streamers*.

6. Twitch: una red de *streamers*

6.1 Construcción

Planteamos la construcción de la red en base a una medida de similitud entre los *streamers* que nos permita establecer como de relacionados están unos con otros. Los nodos de la red representarán *streamers* mientras que las aristas que los conecten representarán el grado de similitud entre ellos. Calcularemos esta medida de similitud mediante sus audiencias. Consideraremos la audiencia de un *streamer* como todos aquellos usuarios que hemos registrado en algún momento como *chatters* de su directo. Para calcular la audiencia de un *streamer* durante todo el periodo de recogida de datos simplemente deberemos obtener el conjunto de todos los usuarios presentes en el campo de *chatters* para las capturas de dicho *streamer*.

Una vez hemos establecido las audiencias para todos los *streamers,* podemos proceder a calcular la similitud entre ellos. Para esta tarea hemos decidido calcular la similitud como la ratio de audiencia de un nodo que también forma parte de la audiencia de otro (Ecuación 11). De esta forma construiremos una red dirigida, ya que esta medida de similitud no es simétrica dados dos streamers cualesquiera.

$$w(a,b) = \frac{|\, audiencia(a) \cap audiencia(b) \,|}{|audiencia(a)|}$$

Ecuación 11. Peso de la arista entre dos nodos dados

Estableciendo de esta forma las relaciones entre nodos obtenemos una red dirigida completa: todos los *streamers* comparten audiencia entre sí. La arista $(lvpes, bobicraftmc)$ cuenta con el peso mínimo de $3{,}797 * 10^{-5}$. La arista con el peso máximo es $(tvander, ibai)$ con un peso de 0,91, es decir, el 91% de la audiencia de *tvander* también consume contenido de *ibai*.

Si analizamos el peso mediano de las aristas, tanto entrantes como salientes, según el número de seguidores de los *streamers* al comienzo del proceso de recogida de datos, encontramos que para los *streamers* con un número reducido de seguidores la mayoría de las aristas entrantes tienen un peso menor a 0,1, muy cercano a 0, mientras que los streamers más grandes presentan valores superiores (Figura 23). En cambio, si analizamos las aristas salientes la situación es muy diferente (Figura 24). En esta ocasión no encontramos una correlación evidente entre el número de seguidores y el peso de las aristas salientes. Independientemente del número de seguidores del *streamer* observamos como el peso mediano de las aristas salientes se sitúa en torno al 0,01, es decir, normalmente un *streamer* comparte con otro una audiencia del 1%. Es interesante destacar que los valores más altos de esta estadística los encontramos en los *streamers* con menor número de seguidores, mientras que los que cuentan con cifras de seguidores millonarias acorde a al conjunto.

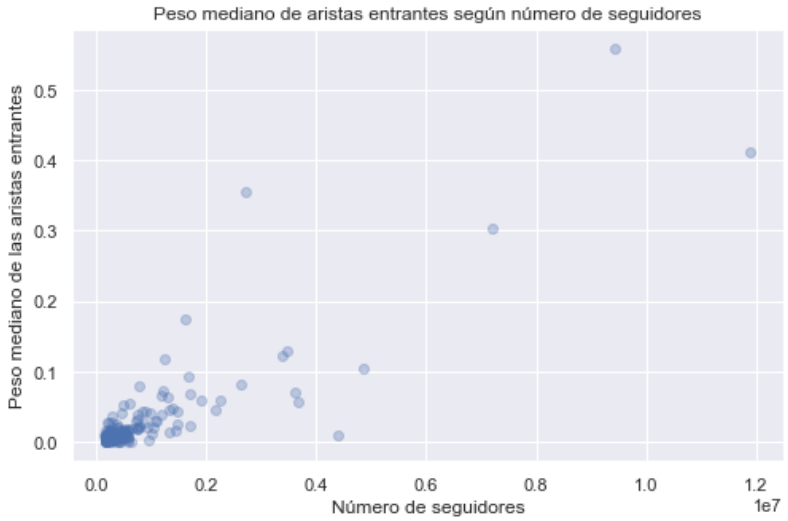

Figura 23. Peso mediano de las aristas entrantes según el número de seguidores del streamer. Punto separador decimal. Elaboración propia.

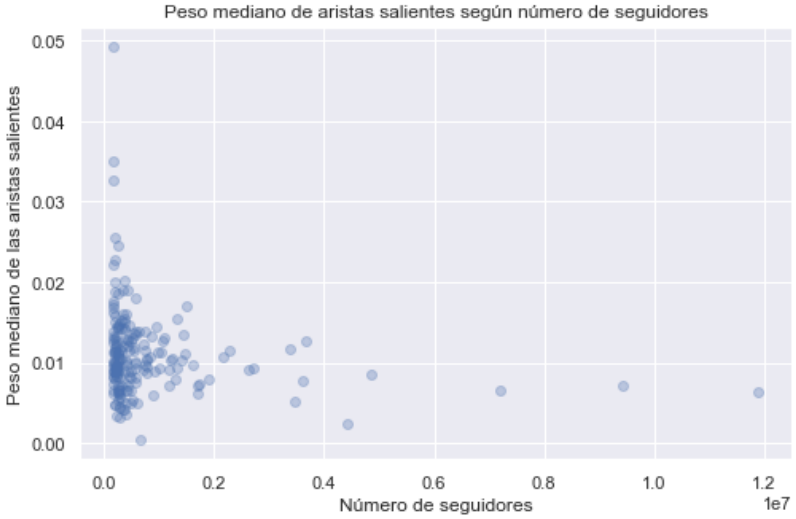

Figura 24. Peso mediano de las aristas salientes según el número de seguidores del streamer. Punto separador decimal. Elaboración propia.

6.2 Filtrado de aristas

Construyendo la red de esta forma obtenemos un grafo completo, es decir, todos los nodos están conectados entre sí y la densidad del grafo es 1. Para poder analizar la red y extraer conclusiones valiosas debemos modificarla para obtener una nueva con una densidad menor. Para ello eliminaremos aquellas aristas que representen relaciones superfluas, intentando mantener las que aporten más información sobre la verdadera estructura subyacente.

Para llevar a cabo esta tarea, existen diferentes alternativas. Mediante una revisión de las metodologías existentes, Darren Edge et al. presentan cuatro técnicas

para llevar a cabo el filtrado de aristas (Edge, 2018). Estas técnicas se basan en la eliminación de aristas según los criterios de:

1. Centralidad de intermediación decreciente (*decreasing edge betweenness centrality*): eliminar aquellas aristas en orden decreciente de centralidad de intermediación. Esta técnica es muy utilizada en algoritmos de detección de comunidades.
2. Peso creciente: eliminar aquellas aristas según su peso u otras métricas agregadas.
3. Información mutua creciente: la presencia de nodos con un alto grado de entrada o salida que son conectados con otros nodos con alto grado pero que realmente representan relaciones sin gran interés. Para determinar este grado de interés entre las conexiones aplican el concepto de información mutua[13] prestado del campo de la teoría de la información.
4. Aleatoriamente: eliminación de aristas aleatoriamente.

Alternativamente, existen métodos estadísticos para realizar este filtrado. Por ejemplo, Navid Dianati propone la utilización de modelos basados en distribuciones binomiales para llevar a cabo esta tarea (Dianati, 2016). De esta forma, se puede caracterizar la significancia de una arista a partir de las propiedades de sus extremos: cuanto mayor sea el grado de los nodos que conecta, mayor deberá ser su peso para poder ser considerada como significativa.

Tras valorar las diferentes alternativas, decidimos realizar un filtrado basado en peso creciente. La gran mayoría de las aristas de la red presentan un peso muy bajo: el 89% de las aristas tiene un peso inferior al 0.1 (Figura 25), es decir, representan relaciones entre *streamers* que comparten una fracción muy pequeña de su audiencia y que probablemente no sea significativa.

Figura 25. Histograma del peso de las aristas. Punto separador decimal. Elaboración propia.

Estas pequeñas relaciones de audiencia pueden surgir por el hecho que unos pocos espectadores compartan intereses con dos *streamers* por muy diferentes que sean los tipos de contenido que ofrezcan o bien pueden emerger por la audiencia flotante, aquellos espectadores que van conectándose de un *stream* a otro como un televidente

[13] La información mutua mide la dependencia estadística entre dos variables aleatorias. Más exactamente, esta medida mide la reducción de entropía de una variable aleatoria al conocer otra.

que navega por los canales de televisión zapeando hasta encontrar un contenido de su agrado.

Una vez hemos decidido qué técnica vamos a emplear para modificar nuestro grafo, surge la necesidad de establecer *cuánto* o *hasta dónde* vamos a modificarlo: ¿filtramos todas las aristas menores de 0.1 o filtramos todas aquellas menores de 0.4? Debemos establecer un criterio que nos oriente durante el proceso de podado de aristas para detectar cuándo hemos alcanzado un punto lo suficientemente bueno como para poder comenzar a trabajar con la red resultante.

Para establecer este criterio, vamos a apoyarnos en dos propiedades bien conocidas de este tipo de redes que representan sistemas humanos, en este caso una red social. Este tipo de grafos presentan distribuciones de grado que siguen leyes de potencias y son redes de mundo pequeño, es decir, la distancia entre dos nodos cualesquiera crece proporcionalmente con respecto al logaritmo del número de nodos de la red. Encontramos ejemplos de estas dos propiedades en muchísimos casos: al construir una red basándonos en las relaciones de seguimiento en Facebook (Ugander, 2011) esta presenta las características habituales de un grafo de mundo pequeño; analizando de manera similar la red social Twitter los resultados arrojan que la red subyacente presenta distribuciones de grado similares a una ley de potencias (Myers, 2014).

Llevaremos a cabo el podado de aristas explorando que peso mínimo deben tener las aristas para que el grafo en conjunto presente las dos propiedades comentadas anteriormente: una distribución de grado que siga una ley de potencias y que el grafo exhiba las características típicas de una red de mundo pequeño. Los umbrales que consideraremos son $\{0, 0{,}05, \dots, 0{,}95, 1\}$.

Para evaluar si un grafo es o no de mundo pequeño no existe una media o definición exacta que nos permita afirmar o negar si un grafo es de mundo pequeño. Normalmente se emplean medidas que comparan las propiedades de una red con otra aleatoria equivalente y proporcionan un valor orientativo. En nuestro caso emplearemos la métrica ω propuesta por Qawi K. et al. (2011) y definida como:

$$\omega = \frac{L_{rand}}{L} - \frac{C}{C_{latt}}$$

Ecuación 12. Métrica omega de mundo pequeño

De esta forma comparamos el camino más corto medio, L, de nuestra red con la misma propiedad de una red aleatoria equivalente, L_{rand}, y los coeficientes de clusterización de nuestra red, C, con el coeficiente de una red de retículo aleatoria equivalente. Los valores cercanos a cero indican que la red presenta características de mundo pequeño. Debe destacarse que esta medida se define para grafos no dirigidos y conexos por lo que el procedimiento para evaluarla será el siguiente:

```
para umbral en umbrales hacer
      F = eliminar aristas de G con peso menor a umbral
      transformar F en no dirigido
      F = componente gigante de F
```

```
calcular ω de F
```

Pseudocódigo 1. Cálculo de omega

Debido al coste computacional de la implementación de esta medida[14], hemos optado por analizar un subconjunto de los valores mínimos ($\{0{,}1, \dots, 0{,}75\}$). Los resultados obtenidos nos indican que para valores menores a 0,5 la red presenta características típicas de un grafo de mundo pequeño (Figura 26).

Figura 26. Variación de la métrica omega según el peso mínimo de las aristas. Punto separador decimal. Elaboración propia.

Una vez establecido qué umbrales mínimos dan lugar a un grafo de tipo mundo pequeño, debemos centrarnos en obtener una distribución de grado que se asemeje lo máximo posible a una distribución de ley de potencias[15]. Para evaluar si una distribución se asemeja a una ley de potencias o no emplearemos la librería *powerlaw* para Python. Esta librería desarrollada por Alstott et al. ofrece una sencilla interfaz mediante la cual analizar distribuciones y evaluar como de similares son a otra serie de distribuciones de referencia (Alstott & Bullmore, 2014).

Para poder decidir si una distribución sigue una ley de potencias, los autores proponen comparar el ajuste de la distribución de datos en estudio con respecto a otras distribuciones de referencia. De esta forma podremos afirmar o no si una ley de potencias es la mejor descripción disponible para dicha distribución o si en cambio encontramos alguna otra que se ajuste mejor. Esta comparación se lleva a cabo empleando el logaritmo de la ratio de verosimilitud entre dos distribuciones dadas. Si la ratio es positiva, la primera distribución se ajusta mejor, si es negativa lo hace la segunda. Esta ratio además viene acompañada de un valor de significancia estadística o p-valor que nos indicará si el signo de esta ratio es significativo o no. En caso de no serlo no podremos afirmar cuál de las dos distribuciones es mejor.

[14] El cálculo de la media de los caminos más cortos y la construcción de la red de retículo aleatoria equivalente requieren de un tiempo considerable de cómputo. Se ha empleado la implementación disponible en la librería de Python NetworkX (Swart, 2008).
[15] Las leyes de potencias se definen de forma: $y = ax^k$.

Procederemos de la siguiente forma: dado un valor umbral eliminaremos todas las aristas del grafo con un peso menor a él, obtendremos la distribución de grado de la red resultante y comprobaremos cual es la distribución que mejor se le ajusta: una ley de potencias, una exponencial o una log-normal. Si exigimos que las ratios vayan acompañadas de un p-valor menor que 0,01, las únicas ratios lo suficientemente significativas son aquellas que comparan una ley de potencias con una exponencial. Obtenemos el valor máximo de la ratio para un umbral de 0,35, es decir, al eliminar las aristas con un peso menor a 0,35 el grafo resultante es el que presenta la distribución que mejor se explica mediante una ley de potencias.

Valor umbral	Comparando	log(ratio verosimilitud)	p-valor
0,20	P vs. E	3,111654	0,001860
0,25	P vs. E	3,061801	0,002200
0,30	P vs. E	2,858977	0,004250
0,35	**P vs. E**	**3,405490**	**0,000660**
0,40	P vs. E	3,213466	0,001311

Abreviaturas: P – ley de potencias; E - exponencial
Tabla 19. Comparaciones que mejor explican la distribución de grado según valor mínimo de las aristas

Atendiendo a los resultados del cálculo de ω podemos observar como el umbral 0,35 también presenta buenos resultados para dicha métrica, por tanto, podemos concluir que al eliminar las aristas con un peso menor a 0,35 obtenemos un grafo que exhibe las propiedades básicas esperables para una red de este tipo. Este grafo cuenta con 2.05% de las aristas originales y 17 nodos quedan desconectados de la componente gigante. En los siguientes capítulos se presentan los resultados obtenidos al trabajar sobre la red podada empleando dicho valor.

6.3 Análisis

La red obtenida presenta una densidad del 0,025, un valor muy inferior al punto de partida. Tal y como buscábamos en la sección anterior, la distribución de grado se asemeja mucho a una ley de potencias (Figura 27). El ~83% de los nodos de los nodos tienen un grado menor o igual a 8 mientras que 4 nodos alargan la cola de la distribución hasta alcanzar un grado de 165 que solamente presenta un nodo, el que corresponde a Ibai Llanos.

Este nodo tiene un grado entrante de 164, esto nos indica que el 94% del resto de *streamers* comparte una porción significativa de su audiencia con este *streamer*. Llamativamente Ibai solamente tiene una arista saliente, que le conecta con el segundo nodo con mayor grado, el correspondiente a Auronplay, alias para Raúl Álvarez. Estos dos *streamers* cuentan, de largo, con la mayor cantidad de seguidores para todos los *streamers* hispanohablantes: 9.420.000 y 11.900.000 respectivamente en el momento en el que iniciamos la captura de datos.

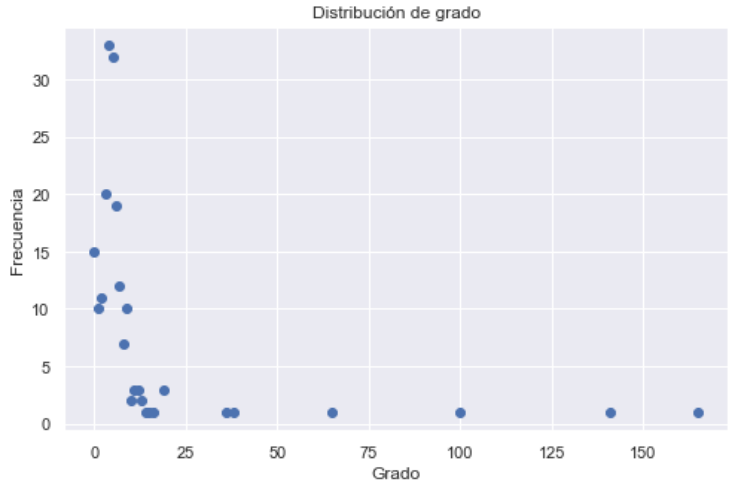

Figura 27. Distribución de grado de la red

Si visualizamos la distribución del grado en función del número de seguidores de cada *streamer* al comienzo del estudio (Figura 28), nos encontramos nuevamente con una situación en la que una larga cola de *streamers* con "pocos" seguidores cuentan con un grado pequeño mientras que unos pocos *streamers* presentan un grado muy elevado. Esta correlación no es tan inmediata, ya que destacan unos pocos nodos que tiene un grado bastante más elevado que otros nodos con mayor número de seguidores. Esto nos puede indicar que algunos *streamers* juegan un papel estructural importante dentro de la red que no podemos juzgar simplemente por el número de seguidores.

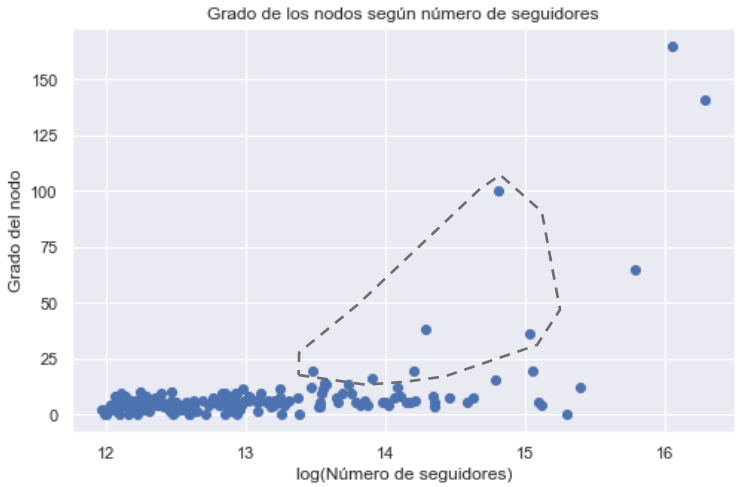

Figura 28. Grado de los nodos según el número de seguidores. Elaboración propia.

Una manera más conveniente de analizar la importancia de un nodo en la red es calculando diferentes medidas de centralidad. Existen multitud de medidas de centralidad, el objetivo de todas ellas es proporcionar una medida cuantitativa de la importancia de un nodo en la estructura de una red. Hemos escogido tres de las más utilizadas:

- Centralidad de cercanía (*closeness centrality*): inversa de la suma de la distancia de los caminos más cortos desde un nodo dado a todos los demás (Bavelas, 1950). Esta métrica valora como más centrales aquellos nodos más cercanos al resto.

- Centralidad de intermediación (*betweenness centrality*): número de veces que un nodo forma parte del camino más corto entre dos nodos cualesquiera (Freeman, 1977). Esta medida nos puede indicar aquellos nodos que sirven de nexo entre diferentes comunidades y por los cuales fluye mucha información.
- Centralidad de vector propio (*eigenvector centrality*): un nodo se considera influyente si este a su vez está conectado a otros nodos influyentes, un valor elevado de esta métrica indica que un nodo está conectado a otros nodos con valores elevados de centralidad de vector propio. Para su cálculo, y tal como indica su nombre, se emplean los valores y vectores propios de la matriz de adyacencia del grafo. Existen múltiples aproximaciones para su cómputo, en este caso emplearemos una basada en los trabajos de Phillip Bonacich (1987) y Mark E. J. Newman (2010) implementada en el paquete NetworkX (Swart, 2008).

Para el cálculo de estas medidas utilizaremos los pesos de las aristas para ponderar la longitud de los caminos en relación a las audiencias compartidas entre los *streamers*. La centralidad de intermediación devuelve valores muy cercanos a cero para todos los nodos de la red. La centralidad de vector propio también devuelve mayoritariamente valores muy bajos excepto para tres nodos. En el caso de centralidad por cercanía obtenemos unos resultados menos homogéneos, acumulándose la mayoría de los nodos en valores inferiores a 0,25, mientras que 6 nodos alcanzan valores mayores (Figura 29).

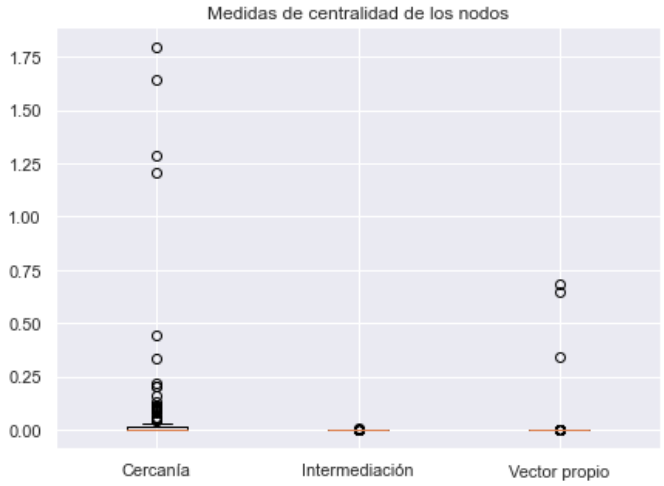

Figura 29. Diagrama de caja para las tres medidas de centralidad. Punto separador decimal. Elaboración propia.

Los nodos que alcanzan valores mayores para la centralidad de cercanía y de vector propio corresponden a *streamers* con las mayores cantidades de seguidores. Sin embargo, no todos los usuarios con mayor número de seguidores obtienen buenos resultados en estas métricas de centralidad, es más, la mitad de los *streamers* que obtienen medidas superiores al 0,25 no ocupan las primeras posiciones en número de seguidores. Este resultado viene a reforzar la hipótesis que la relevancia o importancia de los nodos en la red no viene simplemente dada por el número de seguidores, sino que se deben tener en cuenta otros factores adicionales.

En la Tabla 20 podemos observar pormenorizadamente estos resultados. Las métricas han sido redondeadas a 4 decimales por cuestiones de legibilidad. Encontramos que hay presencia tanto de algunos de los cinco primeros *streamers* por número de seguidores, así como de algunos otros con un número bastante menor.

Streamer	Seguidores	Posición ranking	C. Cercanía	C. Vector propio
auronplay	**11.900.000**	**1**	**1,7968**	0,6436
ibai	9.420.000	4	1,6423	**0,6841**
juansguarnizo	7.190.000	5	1,2861	0,3432
coscu	3.390.000	12	0,3326	0,0
elxokas	2.720.000	18	1,2051	0,0001
illojuan	1.610.000	37	0,4421	0,0

Tabla 20. Nodos con mayores medidas de centralidad. El ranking hace referencia a la ordenación de los streamers según su número de seguidores.

Este resultado aporta solidez a la conveniencia de analizar Twitch empleando redes que no utilicen la relación de seguidores y seguidos. Al considerar la audiencia real de cada uno de estos usuarios obtenemos una representación más fidedigna de las relaciones y similitudes existentes dentro de la comunidad. La audiencia, seguidores o no, que consumen los contenidos ofrecidos por estos *streamers* nos ayuda a entender mejor la realidad de la plataforma que una simple *follower:* el hecho de conectarse y ver a un *streamer* conlleva una inversión de tiempo del espectador y muestra que realmente le interesa y le entretiene lo que está viendo. En cambio, apretar el botón de seguir nos puede animar a entender que un espectador determinado siente interés por un *streamer,* pero si en el día a día no consume su contenido esta relación de seguimiento pierde la posible representatividad que pudiera tener de la realidad de la plataforma.

Al representar gráficamente la red aplicando el algoritmo de ForceAtlas2 para la distribución de los nodos y al escalar el tamaño de cada uno de ellos en función de la centralidad de cercanía, obtenemos un resultado interesante. El algoritmo ForceAtlas2, desarrollado por Mathieu Jacomy et al., es un algoritmo de distribución dirigido por fuerzas: simula un sistema físico en el que los nodos se repelen entre sí, pero las aristas que los conectan los atraen (Jacomy, 2014). Este tipo de algoritmos producen disposiciones espaciales agradables visualmente y que resultan intuitivas a la hora de analizar un grafo y sus posibles comunidades.

Aplicando dicho algoritmo hasta alcanzar una disposición aparentemente estática y escalando cada nodo según su medida de centralidad por cercanía obtenemos la Figura 30. En ella encontraremos resaltados mediante óvalos azules los nodos presentes en la Tabla 20. Inmediatamente saltan a la vista diversas agrupaciones de nodos en las esquinas superiores izquierda y derecha. En el centro encontramos un grupo compacto de nodos, mientras que en la parte inferior de la figura encontramos pequeñas agrupaciones de dos o tres nodos.

En la zona central encontramos tres nodos de gran tamaño, estos corresponden a tres de los seis nodos con mayor valor de centralidad por cercanía. Los tres restantes los encontramos más diseminados: *coscu* aparece como nodo relativamente central de una agrupación en la esquina superior derecha, *juansguarnizo* se sitúa en el centro de

diferentes grupos en la esquina superior izquierda, mientras que a *illojuan* lo encontramos formando parte del grupo central de nodos.

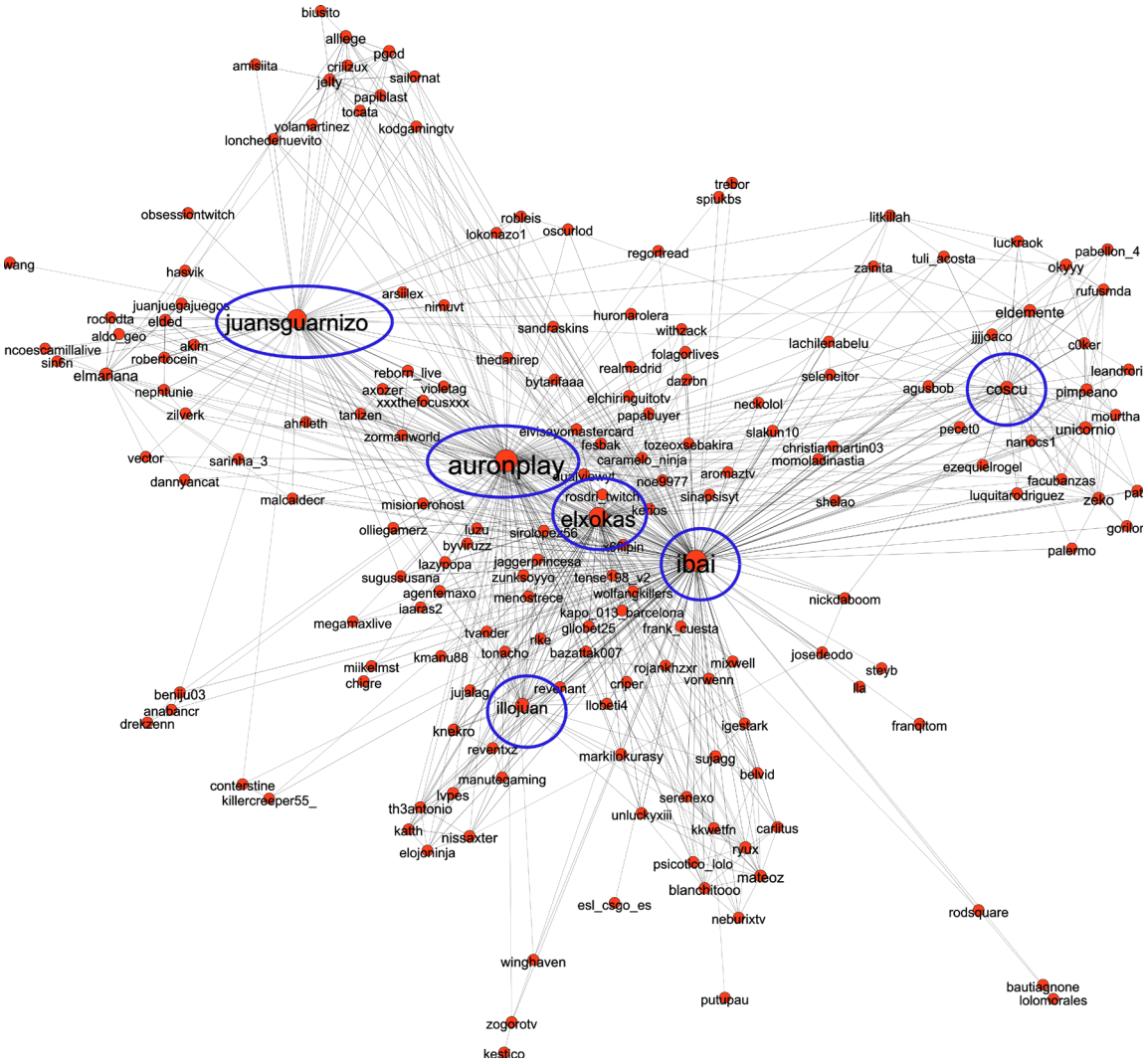

Figura 30. Representación de la red. Tamaño de los nodos según centralidad de cercanía. Disposición aplicando ForceAtlas2. Los nodos señalados son los presenten en la Tabla 20. Elaboración propia.

6.4 Detección de comunidades

La detección de comunidades y su caracterización es una de las principales tareas dentro del análisis de redes. El concepto de comunidad en este contexto no tiene una definición formal sólida. Habitualmente se entiende por comunidad un grupo de nodos que se encuentran más densamente conectados entre ellos que con el resto de nodos de la red. Los nodos que forman parte de una comunidad se espera que compartan ciertos atributos o características. En nuestro caso se trataría del contenido que emiten, la forma en que interactúan con sus espectadores o las relaciones de amistad que pueda haber entre diferentes *streamers* y que haya propiciado compartir una audiencia significativa entre ellos dando lugar a una comunidad. Se puede encontrar una extensa discusión sobre esta cuestión en el artículo de Santo Fortunato (2010).

Algunos de los métodos más empleados en la actualidad para la detección automática de comunidades se basan en un problema de optimización: tienen como objetivo maximizar la modularidad. Según Fortunato, esta medida fue inicialmente introducida por M. E. J. Newman y M. Girvan como una métrica para decidir cuando parar el algoritmo de detección de comunidades que desarrollaron y que lleva sus nombres, el algoritmo Newman-Girvan. La modularidad es una medida que cuantifica la fuerza o la calidad de una partición dada para una red. Basándose en la asunción que los grafos aleatorios no presentan comunidades, la modularidad compara la densidad de una comunidad o sub-grafo con la densidad esperada en el caso que las conexiones entre los nodos de dicha partición fueran aleatorias. Existen diversas maneras de formular dicha medida, pero el factor más importante es determinar que modelo se va a utilizar para generar el grafo equivalente aleatorio.

Para realizar la detección de comunidades hemos utilizado el algoritmo de Leiden. Este algoritmo desarrollado por Traag et al. (2019) viene a solventar los diversos problemas que presentaba el algoritmo de Louvain (D. Blondel, 2008). El algoritmo de Louvain se divide en dos fases, en la primera de ellas asigna a cada nodo su propia comunidad y busca maximizar la modularidad moviendo cada nodo a sus comunidades vecinas. Una vez que no podemos mejorar la modularidad pasamos a la segunda fase, en la que se crea un nuevo grafo donde los nodos representan las comunidades detectadas y las aristas entre ellas agregan las aristas presentes entre los nodos de diferentes comunidades en el grafo original. Estas dos fases se repiten de manera iterativa hasta que se alcanza el máximo de modularidad.

Este algoritmo presenta una debilidad considerable y es que muchas veces encontramos nodos que juegan un papel fundamental actuando como nexo entre dos comunidades. Estos nodos corren el riesgo de ser movidos y agregados a una comunidad, dejando a otra incomunicada. El algoritmo de Leiden amplia Louvain introduciendo una fase en la que se pueden particionar las comunidades, no solo agregarlas. De esta forma se consiguen resultados de mayor calidad, detectando comunidades que de otra forma podrían pasar desapercibidas.

Llevaremos a cabo los experimentos de detección de comunidades utilizando Gephi, una plataforma de código abierto para la visualización y análisis de grafos (Bastian, 2009). Gephi nos permite instalar componentes adicionales desarrollados por la comunidad y que extienden la funcionalidad original. Uno de estos complementos es el *Leiden Algorithm,* que como su propio nombre indica implemente el algoritmo Leiden y ha sido desarrollado por el autor principal del articulo original, Vincent Traag.

Tanto Leiden como Louvain cuentan con un parámetro llamado resolución, su correcto ajuste es vital para obtener buenos resultados. La resolución gobierna el cálculo de la modularidad, valores altos de este parámetro favorecen la detección de un mayor número de comunidades de menor tamaño, mientras que cuanto menor sea su valor se detectan menos comunidades, pero de mayor tamaño. El caso de uso en cuestión y las peculiaridades de cada red determinan el valor que debería tomar la resolución para que ofrezca un buen equilibrio entre número y tamaño de las comunidades.

Para los ejemplos presentados por Traag et al. (2019) utilizan una resolución igual a 1. En el caso que nos ocupa se han explorado los resultados devueltos con

diferentes resoluciones y finalmente también se ha optado por utilizar el valor 1, ya que presenta un buen equilibrio entre número de comunidades y la modularidad (Tabla 21).

Resolución	Modularidad	# comunidades	Resolución	Modularidad	# comunidades
2,0	0,179	19	1,0	0,342	9
1,8	0,205	16	0,8	0,412	6
1,6	0,231	14	0,6	0,504	4
1,4	0,261	10	0,4	0,558	3
1,2	0,296	7	0,2	0,619	2

Tabla 21. Número de comunidades y modularidad asociada según resolución

Encontramos 9 comunidades de tamaños bastante heterogéneos: la comunidad 0 contiene 67 nodos (~38,51%) mientras que la comunidad 8 cuenta con solamente 2 nodos. La comunidad 0 agrupa a la mayoría de los nodos dispersos en la zona central y los dos nodos más relevantes en esta comunidad, según sus centralidades de cercanía, son *ibai* y *elxokas*. La segunda comunidad en número de nodos es la 1, con 40 *streamers* (~22.99%), en ella encontramos los nodos situados en la mitad izquierda de la figura, con dos figuras prominentes dentro de ellas *auronplay* y *juansguarnizo*. La comunidad 3 destaca por presentar una agrupación bastante bien definida de *streamers* latinoamericanos, principalmente de Argentina; en el centro encontramos a *coscu*. En la Figura 31 encontraremos una representación gráfica de la red con la misma distribución que la empleada en la Figura 30. Se han destacado las comunidades más pequeñas mediante un círculo para mejorar su visibilidad.

Podemos observar como la distribución de los nodos empleando el algoritmo ForceAtlas2 ha resultado muy acorde a los resultados obtenidos tras detectar las comunidades existentes. De esta red podemos extraer diferentes conclusiones. La primera de ellas es que los *streamers estrella,* aquellos como *ibai*, *auronplay* o *juansguarnizo,* que cuentan con millones de seguidores y que reúnen diariamente a miles de espectadores, juegan un papel central en la red, ya que la mayoría de espectadores que puedan consumir contenido de otros *streamers* más pequeños, también lo hacen sistemáticamente de los más grandes.

La red, aún representando exclusivamente información sobre las audiencias compartidas entre *streamers,* consigue capturar las similitudes de contenido entre ellos. En la zona inferior izquierda de la clase 0 encontramos un pequeño cúmulo de nodos (*reventxz*, *manutegaming*, *lvpes*, *th3antonio*, *katth*, *nissaxter* y *elojoninja*), los cuales representan *streamers* que suelen emitir contenido centrado en dos videojuegos de Riot Games, League of Legends y Teamfight Tactics, conocidos popularmente como LOL y TFT.

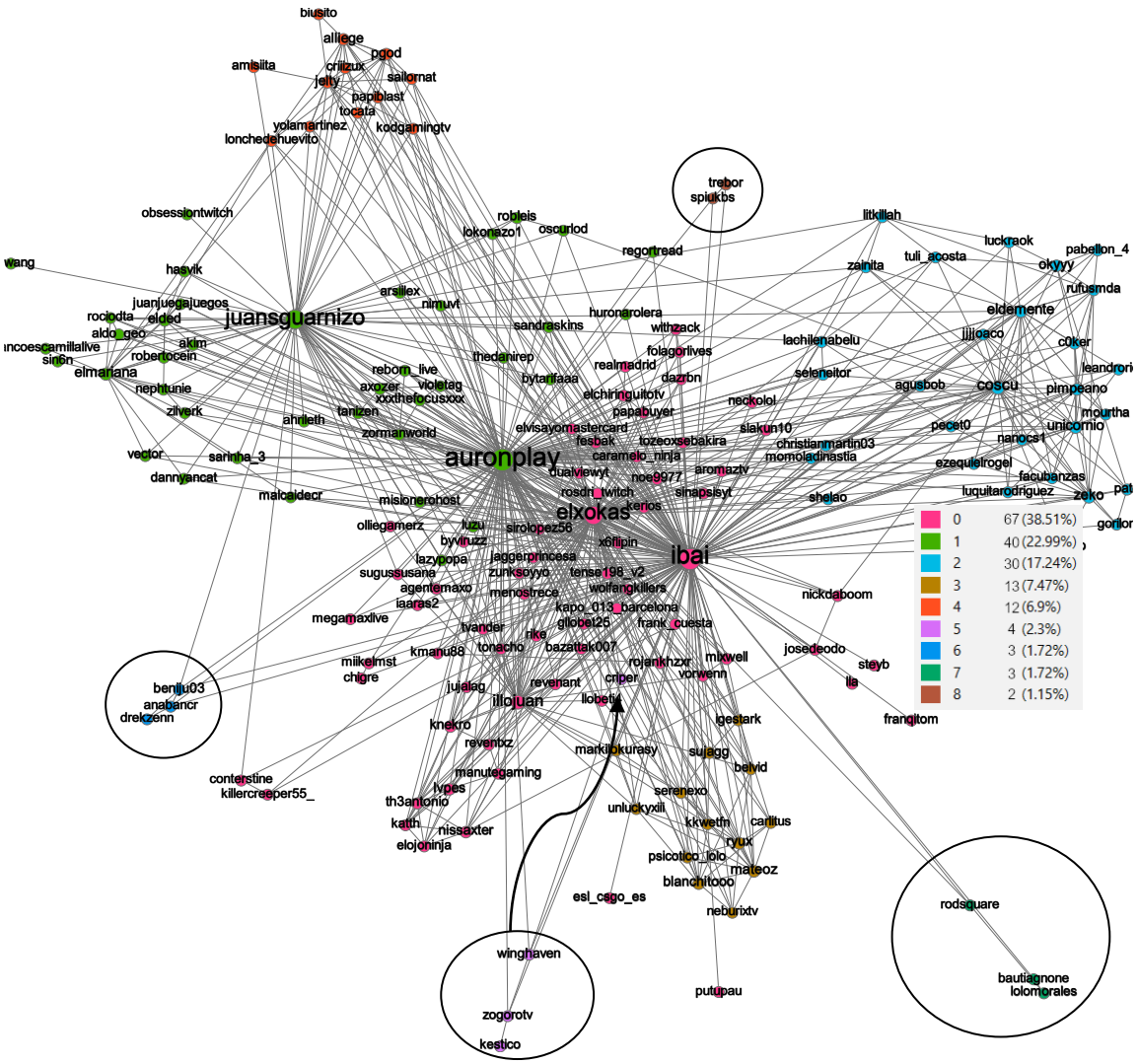

Figura 31. Representación gráfica de la red. Nodos coloreados según a la comunidad a la que pertenezcan. Elaboración propia.

Así mismo, la clase 3, representada en la figura con un color marrón en la zona central inferior, corresponde a *streamers* que emiten principalmente Fortnite. La clase 4, situada en la esquina superior izquierda con nodos naranja, agrupa *streamers* que también emiten mayoritariamente Fortnite. Sin embargo, la mayoría de estos *streamers* son de nacionalidad mexicana. Estas agrupaciones nos indican que la relación de audiencia compartida nos permite capturar las similitudes entre *streamers* tanto a nivel de contenido como también aparentemente a nivel de nacionalidad. Este fenómeno puede explicarse gracias a que es asumible que espectadores que consumen un tipo contenido sean proclives a hacerlo también en otros canales, por ejemplo, un determinado videojuego. Respecto a las agrupaciones por nacionalidad podemos valorar dos factores diferentes, el cultural y la mera disposición geográfica. Culturalmente es comprensible que los espectadores elijan consumir contenidos de creadores cercanos a ellos, en cuanto a discurso, aspectos lingüísticos o demás idiosincrasias habituales de una cultura dada. Geográficamente debemos tener en cuenta la naturaleza del contenido ofrecido en Twitch, se trata de emisiones en directo, por lo que los horarios a los que estas se desarrollen son un factor importante a la hora de entender la audiencia de un *streamer* determinado: un espectador, de origen latinoamericano o no, que pueda residir que Europa, difícilmente se convertirá en parte

de la audiencia de un *streamer* que se encuentre en Latinoamérica y que emita en ciertos momentos del día en los que sea de madrugada en Europa.

6.5 Comunidades vs. *Clustering* por contenido

En el capítulo 5 analizamos la existencia de *clusters* o agrupaciones de *streamers* según dos representaciones vectoriales diferentes basadas en las categorías emitidas por cada uno de ellos. Los resultados demostraban la existencia de diversas agrupaciones según la categoría que predominantemente emitían estos usuarios. Según la combinación de modelo y parámetros podemos obtener unas agrupaciones más o menos granulares, descubriendo un número diferente de *clusters* centrados en la emisión de una u otra categoría en concreto.

En la sección 6.4 hemos discutido como la red propuesta presenta comunidades que también reflejan qué tipo de contenidos emiten los *streamers* que forman parte de ellas, aún no contando con ningún tipo de información sobre ese aspecto. Por ejemplo, encontramos dos comunidades bien delimitadas en las que la mayoría de sus integrantes emiten principalmente el videojuego Fortnite. Este fenómeno nos motiva a realizar una comparación entre las agrupaciones de *streamers* producidas por la detección de comunidades de la red y las agrupaciones del capítulo 5. ¿En qué grado las agrupaciones obtenidas meramente por contenido corresponden las obtenidas en la red?

Para contestar a esta pregunta consideraremos las agrupaciones correspondientes a las comunidades detectadas en la red, la mejor agrupación producida por la aproximación de vectores de porcentajes y PCA y las dos mejores agrupaciones obtenidas con la segunda aproximación, TFIDF y DVS. Realizaremos la comparativa entre estas diferentes particiones del conjunto de *streamers* mediante el índice de Rand (1971). Esta métrica (Ecuación 13) es una medida de similitud entre dos particiones. Se define a como el número de pares de elementos que se encuentran en un mismo grupo en ambas agrupaciones y b como el número de pares de elementos que se encuentran en grupos diferentes tanto en la primera como en la segunda agrupación.

$$R = \frac{a + b}{\binom{n}{2}}$$

Ecuación 13. Índice de Rand

El divisor corresponde a un coeficiente binomial y representa el número total de parejas no ordenadas de elementos en un conjunto de cardinalidad n. En otras palabras, podemos expresar el índice de Rand como la frecuencia de coincidencia sobre el conjunto total de pares de elementos. Desde un punto de vista estadístico se puede entender como la probabilidad de que las dos agrupaciones concuerden para un par aleatorio de elementos.

El índice de Rand es ampliamente utilizado en su versión ajustada, de esta forma, el Índice Ajustado de Rand (IAR) toma el valor 0 como valor base al considerar particiones aleatorias y en las que todos los grupos de una partición tienen la misma cardinalidad (Hubert, 1985). El IAR se encuentra acotado superiormente a 1 cuando las agrupaciones son idénticas. A diferencia de la versión no ajustada, este puede devolver valores negativos. Valores cercanos al 0 indican que las agrupaciones corresponden a una partición aleatoria independientemente del número *clusters* o muestras.

Comparando las agrupaciones obtenidas mediante la detección de comunidades con aquellas obtenidas en los procesos de agrupación según contenido mediante el IAR, los resultados obtenidos son valores muy cercanos a cero para cualquiera de las comparaciones. El mejor valor de IAR, 0,0881, se obtiene con la mejor agrupación utilizando el pesado TFIDF y DVA. El peor resultado lo obtiene la mejor agrupación de la aproximación porcentajes y PCA, 0,0176 (Tabla 22).

Comparación con ...	Índice Ajustado de Rand
1ª agrupación de porcentajes + PCA	0,0176
1ª agrupación de TFIDF + DVS	0,0881
2ª agrupación de TFIDF + DVS	0,0656

Tabla 22. Comparación entre agrupaciones por contenido y comunidades de la red

Estos resultados tan pobres nos indican que no existe una correspondencia directa entre las agrupaciones basadas meramente en contenido con aquellas obtenidas al construir la red. La coincidencia o similitud de los contenidos emitidos por los *streamers* no es uno de los factores principales que rigen la creación de las diferentes comunidades encontradas en la red.

7. Conclusiones

Mediante el presente trabajo hemos presentado un sistema de captura de datos que, junto con metodologías innovadoras, nos ha permitido llevar a cabo un análisis de la comunidad hispanohablante en la plataforma Twitch. Este servicio de emisión de contenido audiovisual en vivo ha demostrado ser una importante alternativa al consumo tradicional de televisión y radio, así como un revulsivo para el consumo en línea, ya que hasta hace unos pocos años este tipo de servicios aún debían superar multitud de limitaciones técnicas para poder convertirse en accesible para millones de personas en todo el mundo.

Partiendo de una revisión bibliográfica que contextualiza la plataforma, las líneas de investigación existentes en torno a ella y la situación en la que se encuentra, se propone un sistema de recogida y almacenamiento de datos que nos ha permitido capturar una ingente cantidad de información sobre los hábitos de emisión del segmento de *streamers* con más peso en la escena hispanohablante. Este sistema se ha demostrado robusto y capaz de soportar el volumen de datos generado. La utilización de tecnologías basadas en eventos y de computación en la nube ha permitido desarrollar un sistema escalable y fiable.

Mediante los datos recogidos y aquellos obtenidos mediante terceros, ha sido posible realizar una instantánea de un ecosistema dominado por unos pocos usuarios con grandes audiencias diarias y millonarias cifras de seguidores. Así mismo, se ha encontrado que aquellos *streamers* que más han crecido en número de seguidores durante el periodo de estudio, han sido precisamente los que inicialmente ya contaban con una mayor cantidad, reproduciendo así el efecto del rico se hace más rico, o efecto Mathew.

Estos grandes *streamers* consiguen capturar gran parte de la audiencia de la plataforma: el 94% de los *streamers* de la red comparten una porción significativa con un solo nodo, el correspondiente a Ibai Llanos. Nos ha sido posible alcanzar esos resultados gracias a la creación de una red dirigida de *streamers* en la cual hemos podido capturar la relación entre ellos mediante la similitud de sus audiencias. Tras aplicar una metodología para el filtrado del grafo, hemos obtenido una versión reducida que reproduce las propiedades esperadas para un grafo como este, en el que se representa una red humana.

Aplicando un algoritmo de detección de comunidades hemos demostrado que esta red refleja tanto la similitud en el contenido emitido por los *streamers* como otras características como el país de origen u otros aspectos culturales. Comparando estas comunidades con las agrupaciones generadas considerando exclusivamente la similitud del contenido emitido por cada *streamer* podemos concluir que estas particiones del conjunto de *streamers* son del todo diferentes, reafirmando así la conclusión que la relación de similitud de audiencias entre *streamers* nos permite construir una red que captura tanto parecidos en el contenido como en otros aspectos más contextuales.

7.1 Trabajos futuros

Existen diversas cuestiones que nos hubiera gustado abordar y que finalmente se han quedado en el tintero. La aproximación empleada en este trabajo para construir la red es muy fácilmente aplicable a la inversa, es decir, podemos poner el foco en los espectadores y analizar cómo se agrupan y relacionan entre ellos según la similitud del contenido que consumen o qué *streamers* suelen ver habitualmente. Esta línea sería verdaderamente innovadora, ya que en la revisión bibliográfica realizada no se ha encontrado ninguna referencia que aborde esta cuestión.

Así mismo, puede resultar interesante aplicar técnicas más modernas de aprendizaje automático o inteligencia artificial para generar *embeddings* de los *streamers* según el tipo de contenido emitido, el patrón temporal de emisión, número de espectadores u otros factores. Podría explorarse la utilización de *transformers* para tal fin, ya que son modelos de aprendizaje profundo que presentan muy buenos resultados en este tipo de tareas.

Por otro lado, sobre el grafo de *streamers,* podrían llevarse a cabo *node embeddings* para explorar el rendimiento de diferentes técnicas en la captura de las características estructurales de cada nodo y su transformación objetos de espacios vectoriales, fácilmente utilizables en otras tareas de aprendizaje automático como la agrupación, llevada a cabo en este trabajo. De nuevo, estas ideas finales se pueden llevar a cabo situando al espectador en el centro del estudio, analizando sus similitudes en cuanto a gustos o patrones de consumo según categoría y momento del día. Sin embargo, antes de avanzar en esa dirección, sería necesario realizar una revisión exhaustiva de los términos y condiciones de Twitch, así como de los posibles dilemas éticos que pudieran surgir.

Finalmente, este tipo de metodologías aquí propuestas pueden y deben ser empleadas para analizar la salud de la plataforma desde una perspectiva de género y de respeto a los derechos y libertades de los usuarios. Se pueden analizar las diferencias en los patrones de emisión y de consumo de los espectadores según la identidad de género de los y las *streamers* o de cualquier otra característica como el país de nacimiento o residencia.

8. Referencias

Albert-Laszlo Barabasi, R. A. (1999). Emergence of Scaling in Random Networks. *Science, 286*, 509*512. doi:10.1126/science.286.5439.509

Alstott, J., & Bullmore, E. a. (2014). powerlaw: A Python Package for Analysis of Heavy-Tailed Distributions. *PLoS ONE*. doi:10.1371/journal.pone.0085777

Ariza, L. M. (2015). La terminología "gamer" en el contexto. *Revista Electrónica del Lenguaje, 2*. Obtenido de https://www.revistaelectronicalenguaje.com/wp-content/uploads/2015/10/vol-02-05.pdf

Barratt, E. a. (2015). Autonomous Sensory Meridian Response (ASMR): A flow-like mental state. *PeerJ, 3*, e851. doi:10.7717/peerj.851

Bastian, M. a. (2009). Gephi: An Open Source Software for Exploring and Manipulating Networks. *International AAAI Conference on Weblogs and Social Media*. Obtenido de http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154

Bavelas, A. (1950). Communication Patterns in Task-Oriented Groups. *The Journal of the Acoustical Society of America*, 725-730. doi:10.1121/1.1906679

Bonacich, P. (1987). Power and Centrality: A Family of Measures. *American Journal of Sociology*, 1170 - 1182. Obtenido de http://www.jstor.org/stable/2780000

Boyd, D. (2022). *SullyGnome.com*. Recuperado el 08 de 04 de 2022, de SullyGnome.com: https://sullygnome.com/games/2018april/streamed?language=es

Claudia Flores-Saviaga, J. H. (2019). Audience and Streamer Participation at Scale on Twitch. *30th ACM Conference on Hypertext and Social Media* (págs. 277–278). New York: Association for Computing Machinery. doi:https://doi.org/10.1145/3342220.3344926

Colin Ford, D. G. (2017). Chat Speed OP PogChamp: Practices of Coherence in Massive Twitch Chat. *CHI Conference Extended Abstracts on Human Factors in Computing Systems* (págs. 858–871). New York: Association for Computing Machinery. doi:https://doi.org/10.1145/3027063.3052765

D. Blondel, V. a.-L. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*. doi:10.1088/1742-5468/2008/10/p10008

Davies, D. L. (1979). A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 224-227. doi:10.1109/TPAMI.1979.4766909

Dianati, N. (2016). Unwinding the hairball graph: Pruning algorithms for weighted complex networks. *Physical Review E, 93*. doi:10.1103/physreve.93.012304

Dux, J. (2018). Social Live-Streaming : Twitch.TV and Uses and Gratification Theory Social Network Analysis. *8th International Conference on Computer Science, Engineering and Applications*, *8*. doi:https://doi.org/10.5121/csit.2018.80305

Dux, J. (2018). Social Live-Streaming : Twitch.TV and Uses and Gratification Theory Social Network Analysis. *8th International Conference on Computer Science, Engineering and Applications*, (págs. 47-61). doi:http://dx.doi.org/10.5121/csit.2018.80305

Eckart, C. Y. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, 211-218. doi:https://doi.org/10.1007/BF02288367

Edge, D. a. (2018). Trimming the Hairball: Edge Cutting Strategies for Making Dense Graphs Usable. *2018 IEEE International Conference on Big Data (Big Data)*, (págs. 3951-3958). doi:10.1109/BigData.2018.8622521

Erik Harpstead, J. S. (2019). Toward a Twitch Research Toolkit: A Systematic Review of Approaches to Research on Game Streaming. *Annual Symposium on Computer-Human Interaction in Play* (págs. 111–119). New York: Association for Computing Machinery. doi:https://doi.org/10.1145/3311350.3347149

F.R.S., K. P. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 559-572. doi:10.1080/14786440109462720

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 75--174. doi:10.1016/j.physrep.2009.11.002

Freeman, L. C. (1977). A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 35 - 41. Obtenido de http://www.jstor.org/stable/3033543

García, R. C. (2021). El lenguaje de los videojuegosanglicismo y creatividad léxica en la plataforma Twitch. En R. P. Salud Adelaida Flores Borjabad, *Nuevos retos y perspectivas de la investigación en Literatura, Lingüística y Traducción* (págs. 1062-1082).

Gutiérrez Lozano, J. F. (2020). El auge de Twitch: nuevas ofertas audiovisuales y cambios del consumo televisivo entre la audiencia juvenil. *Revista Internacional De Comunicación, 50*, págs. 159-175. doi:https://doi.org/10.12795/Ambitos.2020.i50.11

Harabasz, T. C. (1974). A dendrite method for cluster analysis. *Communications in Statistics, 3*(1), 1-27. doi:10.1080/03610927408827101

Hohyun Jung, F. K. (2021). On the effects of capability and popularity on network dynamics with applications to YouTube and Twitch networks. *Physica A: Statistical Mechanics and its Applications, 571*. doi:https://doi.org/10.1016/j.physa.2020.125663.

Hotelling, H. (1936). Relations Between Two Sets of Variates. *Biometrika*, 321-377. doi:https://doi.org/10.2307/2333955

Hubert, L. a. (1985). Comparing partitions. *Journal of Classification*, 193–218. doi:10.1007/BF01908075

Jacomy, M. A. (2014). ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software. *PLOS ONE*, 1-12. doi:10.1371/journal.pone.0098679

Kondor D, P. M. (2014). Do the rich get richer? An empirical analysis of the Bitcoin transaction network. *PloS one, 9*(2). doi:https://doi.org/10.1371/journal.pone.0086197

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. *Berkeley Symposium on Mathematical Statistics and Probability*, (págs. 281-297).

Martin Ester, H.-P. K. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Second International Conference on Knowledge Discovery and Data Mining* (págs. 226-231). AAAI Press. doi:https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.121.9220

Merton, R. K. (1968). The Matthew Effect in Science. *Science, 159*, 56-63. doi:10.1126/science.159.3810.56

Mihael Ankerst, M. M.-p. (1999). OPTICS: Ordering Points To Identify the Clustering Structure. *Int. Conf. on Management of Data* (págs. 49-60). Philadelphia : ACM Press.

Mirsky, L. (1960). SYMMETRIC GAUGE FUNCTIONS AND UNITARILY INVARIANT NORMS. *The Quarterly Journal of Mathematics, 11*(1), 50-59. doi:https://doi.org/10.1093/qmath/11.1.50

Myers, S. A. (2014). Information Network or Social Network? The Structure of the Twitter Follow Graph. *23rd International Conference on World Wide Web* (págs. 493–498). New York, NY, USA: Association for Computing Machinery. doi:10.1145/2567948.2576939

Newman, M. (2010). *Networks: An Introduction*. OUP Oxford. doi:10.1093/acprof:oso/9780199206650.001.0001

Pavel Dolin, L. d. (2021). FeelsGoodMan: Inferring Semantics of Twitch Neologisms. doi:https://doi.org/10.48550/arXiv.2108.08411

Pedregosa, F. a. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research, 12*, 2825--2830}.

Rachel Kowert, E. D. (2021). The one-and-a-half sided parasocial relationship: The curious case of live streaming. *Computers in Human Behavior Reports, 4*. doi:https://doi.org/10.1016/j.chbr.2021.100150.

Rand, W. M. (1971). Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 846-850. doi:10.1080/01621459.1971.10482356

Rosenblatt, M. (1956). Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics, 27*, 832-837. doi:https://doi.org/10.1214%2Faoms%2F1177728190

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics, 20*, 53-65. doi:https://doi.org/10.1016/0377-0427(87)90125-7

Rozemberczki, B. a. (2021). Twitch Gamers: a Dataset for Evaluating Proximity Preserving and Structural Role-based Node Embeddings. doi:https://doi.org/10.48550/arxiv.2101.03091

Steinhaus, H. (1957). Sur la division des corps matériels en parties. *Bull. Acad. Polon. Sci. Cl.*, 801-804.

Swart, A. A. (2008). Exploring Network Structure, Dynamics, and Function using NetworkX. *Proceedings of the 7th Python in Science Conference* (págs. 11 - 15). Pasadena, CA USA: Gäel Varoquaux, Travis Vaught, and Jarrod Millman.

Telesford, Q. K. (2011). The Ubiquity of Small-World Networks. *Brain Connectivity*, 367-375. doi:10.1089/brain.2011.0038

Traag, V. A. (2019). From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports*. doi:10.1038/s41598-019-41695-z

Twitch Interactive, Inc. (2022). *Understanding Viewer Count vs. Users in Chat*. Recuperado el 11 de 05 de 2022, de Help: https://help.twitch.tv/s/article/understanding-viewer-count-vs-users-in-chat?language=en_US

Twitch Interactive, Inc. (2019). *Chatters / Viewers Helix API Endpoint*. Obtenido de Twitch UserVoice Developers: https://twitch.uservoice.com/forums/310213-developers/suggestions/39145294-chatters-viewers-helix-api-endpoint

Twitch Interactive, Inc. (2020). *Chatters API - Not seeing VIPs and some other chatters/viewers*. Obtenido de Twitch Developer Forums: https://discuss.dev.twitch.tv/t/chatters-api-not-seeing-vips-and-some-other-chatters-viewers/27279

Twitch Interactive, Inc. (2020). *Huge gap chatters vs. viewers*. Obtenido de Twitch Developer Forums: https://discuss.dev.twitch.tv/t/huge-gap-chatters-vs-viewers/25940/3

Twitch Interactive, Inc. (2022). *Community Coding Resources*. Obtenido de Twitch Developers: https://dev.twitch.tv/code

Twitch Interactive, Inc. (2022). *Press Center*. Obtenido de Twitch TV: https://www.twitch.tv/p/press-center/

TwitchTracker. (2022). Obtenido de https://twitchtracker.com/

Twurple. (2022). *Twurple Github Repository*. Recuperado el 11 de 05 de 2022, de Github: https://github.com/twurple/twurple

Ugander, J. a. (2011). The anatomy of the facebook social graph. *arXiv preprint arXiv:1111.4503*.

Waskom, M. (2012). *Seaborn API Reference KDEplot*. Recuperado el 11 de 05 de 2022, de Seaborn: https://seaborn.pydata.org/generated/seaborn.kdeplot.html

William A. Hamilton, O. G. (2014). Streaming on twitch: fostering participatory communities of play within live mixed media. *Conference on Human Factors in Computing Systems* (págs. 1315–1324). New York: Association for Computing Machinery. doi:https://doi.org/10.1145/2556288.2557048

Anexo I

Los streamers para los cuales se han recogido datos, con estadísticas[16] referentes al 23/03/2022, son los siguientes:

Streamer	Seguidores	Número medio de espectadores concurrentes	Horas emitidas	Horas consumidas
auronplay	11.900.000	106.038	97,50	10.300.000
ibai	9.420.000	90.423	111,70	10.100.000
juansguarnizo	7.190.000	53.017	203,30	10.800.000
elded	4.870.000	9.231	177,50	1.640.000
quackity	4.410.000	56.670	5,80	327.000
slakun10	3.670.000	15.513	15,60	242.000
robleis	3.610.000	11.605	180,80	2.100.000
elmariana	3.470.000	42.369	93,60	3.970.000
coscu	3.390.000	5.829	85,80	500.000
elxokas	2.720.000	48.385	268,50	13.000.000
jelty	2.640.000	7.960	84,80	675.000
litkillah	2.270.000	8.810	15,80	139.000
luzu	2.160.000	9.085	78,80	716.000
reborn_live	1.900.000	4.530	134,40	609.000
nimuvt	1.720.000	8.575	70,80	607.000
robertocein	1.710.000	2.266	155,90	353.000
axozer	1.690.000	8.111	168	1.360.000
illojuan	1.610.000	36.249	147,60	5.350.000
markilokurasy	1.490.000	3.963	31,10	123.000
eldemente	1.470.000	4.127	51,30	212.000
llobeti4	1.450.000	1.211	89,60	109.000
folagorlives	1.400.000	4.764	252,90	1.200.000
lachilenabelu	1.340.000	2.894	11,50	33.200
pimpeano	1.320.000	2.553	39,40	101.000
xxxthefocusxxx	1.290.000	2.874	132,80	382.000
thedanirep	1.230.000	3.352	22,30	74.800
benijuo3	1.210.000	12.697	25,40	323.000
jaggerprincesa	1.190.000	5.578	28,90	161.000
zilverk	1.180.000	13.530	86	1.160.000
elrichmc	1.150.000	2.947	66,20	195.000
zeko	1.100.000	1.520	53,70	81.600
bytarifaaa	1.060.000	1.167	113,70	133.000
sujagg	1.040.000	1.468	224,60	330.000
menostrece	1.010.000	2.849	156,80	447.000
yolo	994.000	4.614	21,70	100.000
agentemaxo	976.000	2.535	113,70	288.000
jjjjoaco	944.000	351	57,70	20.200
alliege	928.000	2.042	125	255.000
lvpes	886.000	17.840	152,10	2.710.000

[16] Todas las estadísticas except el número de seguidores se calculan en el period de 30 días anteriores a la consulta de los datos.

momoladinastia	862.000	3.055	6	18.400
reventxz	847.000	2.649	177,10	469.000
pgod	788.000	2.100	64,10	135.000
mateoz	782.000	499	158,30	79.000
knekro	773.000	16.077	265,90	4.270.000
okyyy	764.000	3.261	39,90	130.000
mixwell	757.000	10.882	149,20	1.620.000
conterstine	755.000	2.383	159,80	381.000
iaaras2	752.000	1.157	63,20	73.100
unicornio	718.000	4.508	107,90	486.000
ryux	712.000	541	150,90	81.600
bethesda	650.000	21.951	6,50	142.000
zainita	646.000	1.094	96,80	106.000
elchiringuitotv	608.000	4.922	180,70	890.000
roier	594.000	2.855	59,70	170.000
withzack	585.000	793	13	10.300
jashlem	571.000	634	50,80	32.200
tuli_acosta	570.000	1.106	13,70	15.100
nephtunie	570.000	712	152,70	109.000
blanchitooo	570.000	873	198,10	173.000
olliegamerz	568.000	3.697	122,20	452.000
neburixtv	566.000	199	55,90	11.100
violetag	566.000	478	87,40	41.700
bazattak007	545.000	150	54,60	8.200
tocata	542.000	401	117,10	46.900
hasvik	534.000	629	85	53.400
aldo_geo	522.000	4.122	107,30	442.000
oscurlod	518.000	729	65,20	47.500
nissaxter	492.000	4.395	87,90	386.000
rociodta	489.000	1.565	61,70	96.600
akim	486.000	325	84,40	27.400
lla	486.000	6.591	33,40	220.000
tvander	477.000	5.181	112	580.000
elojoninja	458.000	1.382	138,10	191.000
igestark	455.000	475	118,50	56.300
nanocs1	454.000	521	196,40	102.000
pato	440.000	10.147	15,10	153.000
rufusmda	437.000	716	9,60	6.900
tense198_v2	432.000	230	336,80	77.400
huronarolera	428.000	219	19,40	4.300
pabellon_4	423.000	2.819	37,70	106.000
bobicraftmc	418.000	1.172	14,90	17.500
th3antonio	418.000	3.968	69,90	277.000
revenant	415.000	4.802	217,70	1.050.000
aquinoby2002	412.000	3.587	28,60	103.000
facubanzas	412.000	427	42,80	18.300
zormanworld	407.000	768	115,60	88.700
alewang	402.000	892	72,20	64.400
vector	398.000	488	31,90	15.500
carlitus	395.000	117	156	18.300
rodsquare	389.000	2.266	135,20	306.000
x6flipin	385.000	910	60,10	54.700

westcol	382.000	5.492	55,30	304.000
belvid	381.000	243	120,40	29.200
peceto	375.000	123	106,10	13.000
vorwenn	370.000	660	27,30	18.000
tanizen	368.000	1.606	208,10	334.000
luckraok	368.000	862	10,80	9.300
winghaven	366.000	1.071	140,30	150.000
papiblast	351.000	127	14,70	1.900
rike	347.000	193	95	18.400
gllobet25	340.000	359	128,40	46.100
neutrooyt	332.000	2.158	179	386.000
agusbob	328.000	985	34,30	33.800
libardoisaza	323.000	302	39,60	12.000
franqitom	314.000	523	73,60	38.500
neckolol	307.000	1.583	32,20	50.900
psicotico_lolo	307.000	117	179,20	20.900
eddy_skabeche	305.000	370	16,10	6.000
rosdri_twitch	305.000	361	78,80	28.400
kestico	304.000	1.162	84,50	98.100
lolomorales	299.000	268	12,10	3.200
ijenz	298.000	927	718,60	666.000
yanpolgg	297.000	269	31,60	8.500
memounstro	296.000	69	120,40	8.300
francoescamillalive	290.000	35	567,70	19.900
juanjuegajuegos	289.000	298	126,90	37.900
chigre	282.000	3.187	70,80	226.000
sandraskins	282.000	856	84,50	72.300
noe9977	281.000	546	69,20	37.800
jujalag	278.000	3.249	123,20	400.000
josedeodo	272.000	7.503	12,50	93.800
nickdaboom	270.000	529	64,80	34.300
putupau	270.000	2.778	261,60	727.000
miikelmst	269.000	1.466	116	170.000
seleneitor	267.000	542	5,20	2.800
killercreeper55_	267.000	321	32,10	10.300
luquitarodriguez	266.000	3.198	14,80	47.400
angievelasco08	265.000	5.253	29,10	153.000
blendfreshon	263.000	460	49,10	22.500
realmadrid	263.000	1.436	34,80	49.900
serenexo	261.000	75	114,20	8.600
manutegaming	261.000	1.815	65,80	120.000
kerios	259.000	1.441	90,50	130.000
esl_csgo_es	259.000	2.786	48,80	136.000
bautiagnone	258.000	697	58,80	41.000
arsiilex	253.000	388	57,10	22.100
dannyancat	251.000	888	74	65.800
jucaviapri	248.000	1.096	33,30	36.500
dualviewyt	247.000	516	59,10	30.500
zunksoyyo	246.000	383	205,20	78.600
kapo_013_barcelona	244.000	1.407	29,40	41.300
katth	243.000	330	90,70	29.900
shelao	237.000	587	75,90	44.600

criiizux	234.000	276	81,20	22.400
papabuyer	234.000	750	97,70	73.300
elvisayomastercard	234.000	1.062	32	34.000
yolamartinez	233.000	398	104,60	41.700
lazypopa	224.000	241	104,80	25.300
steyb	223.000	315	195,40	61.500
lokonazo1	219.000	515	182	93.700
amisiita	219.000	529	19,70	10.400
unluckyxiii	217.000	110	160,10	17.600
kmanu88	217.000	707	24,30	17.200
spiukbs	215.000	491	64,90	31.900
sin6n	215.000	798	64,90	51.800
martibenza	214.000	5.261	10,80	56.800
rojankhzxr	214.000	644	68,10	43.800
tozeoxsebakira	213.000	294	97,50	28.700
zogorotv	213.000	186	102,80	19.100
frank_cuesta	212.000	842	88,30	74.300
sugussusana	210.000	681	63,90	43.500
christianmartin03	210.000	1.865	18,20	33.900
fesbak	209.000	551	92,30	50.900
kkwetfn	209.000	82	133,70	10.900
kodgamingtv	208.000	272	17,50	4.800
dazrbn	206.000	512	356,90	183.000
llocochon	205.000	1.748	171,10	299.000
lulignzalez	205.000	4.919	7	34.100
caramelo_ninja	203.000	647	112,30	72.700
mourtha	203.000	110	7,90	868
sirolopez56	196.000	1.378	134,10	185.000
sarinha_3	193.000	368	91,60	33.700
megamaxlive	192.000	784	123,10	96.600
darkinvasion	192.000	112	42,60	4.800
malcaidecr	191.000	133	80,70	10.700
lonchedehuevito	187.000	1.763	50,10	88.300
leandroriccio	186.000	157	9,50	1.500
criper	184.000	653	108,20	70.600
coker	182.000	1.862	36,60	68.200
byviruzz	182.000	183	12	2.200
misionerohost	181.000	88	20	1.800
vicioonemoretime	180.000	172	97,30	16.800
sinapsisyt	180.000	142	109,30	15.600
sailornat	178.000	319	35,70	11.400
obsessiontwitch	177.000	687	11,40	7.800
tonacho	176.000	294	77,90	22.900
anabancr	175.000	3.035	65,70	199.000
ahrileth	175.000	71	153,80	10.900
ezequielrogel	174.000	48	136,20	6.600
trebor	172.000	1.405	54,30	76.400
palermo	171.000	82	135,20	11.000
aromaztv	170.000	219	60,50	13.200
wolfangkillers	169.000	167	181,20	30.200
gorilon	169.000	1.484	70,10	104.000
herocharly	164.000	27	271	7.200

kathk7	163.000	64	23,90	1.500
biusito	162.000	198	189,40	37.500
sharonwinner	161.000	284	76	21.600
regortread	161.000	42	125,80	5.300
drekzenn	159.000	1.143	142,70	163.000

Tabla 23. Listado completo de streamers objeto de estudio

Anexo II. Rendimiento del sistema de recogida de datos.

Uno de los aspectos críticos del sistema de recogida de datos era que fuera capaz de responder al alto volumen de datos entrante y poder mantener los intervalos de captura dentro de un intervalo de tiempo aceptable. Temíamos que la concurrencia de diversas emisiones con un alto número de espectadores conllevará un alto coste computacional de procesamiento, tanto en operaciones de CPU como de IO, y que esto pudiera afectar a la deseada regularidad de la captura de datos. Cabe decir que una desviación de unos pocos segundos en esta frecuencia de captura no es de vital importancia, ya que no tendría por qué suponer una gran diferencia a efectos prácticos sabiendo de la alta tasa de *caching* en los *endpoints* de Twitch, sin embargo, resultaba de especial interés evaluar cómo de robusto había sido el sistema para poder valorar su capacidad de escalado.

Es importante señalar que Node.js, la plataforma elegida para el desarrollo del sistema de captura, se basa en JavaScript y, por tanto, se trata de un lenguaje que no soporta múltiples hilos de ejecución. La concurrencia en Node.js se entiende como la capacidad del bucle de eventos interno de planificar la ejecución de diferentes rutinas en tiempo de ejecución[17]. En caso de que muchos de los intervalos de recogida de datos comenzarán a desviarse excesivamente de su tiempo previsto de ejecución, esta desviación comenzaría a acumularse y podría causar que la recogida de datos se desviara excesivamente.

Para evaluar el comportamiento del sistema durante todo el proceso, calcularemos la desviación entre las sucesivas capturas para todas las retrasmisiones. Seleccionamos todas las capturas para una retransmisión dada, seleccionamos el campo referente al instante temporal de recogida y los ordenamos de manera ascendente. Siendo t el conjunto de marcas temporales ordenadas, calculamos las desviaciones en segundos como: $\Delta = \{(t_{i+1} - t_i) - 300 \mid i \in [0, |t| - 1]\}$, siendo 300 la frecuencia esperada en segundos. Al considerar todas las desviaciones en conjunto y tras eliminar posibles valores atípicos aplicando el método del rango intercuartil, obtenemos el histograma de la Figura 32:

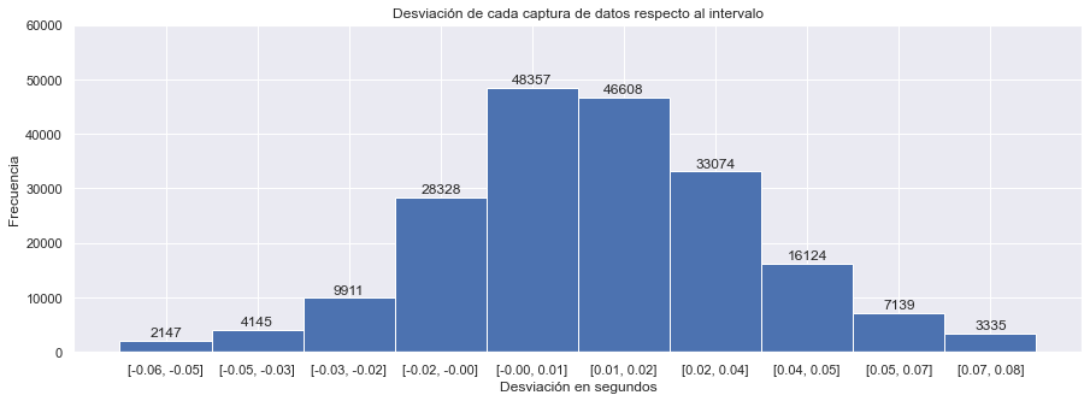

Figura 32. Histograma de la desviación temporal entre capturas respecto al intervalo omitiendo valores atípicos. Punto separador decimal. Elaboración propia.

[17] https://nodejs.org/en/docs/guides/blocking-vs-non-blocking/

Podemos comprobar como los valores no considerados atípicos se distribuyen en el rango [-0,06, 0,08] segundos de desviación, concentrándose la mayoría en el intervalo [-0,02, 0,04]. Sin embargo, llama la atención que aproximadamente el 15% (~37.000) de las muestras son consideradas atípicas. Si nos fijamos en estas desviaciones atípicas (Figura 33), podemos observar como aproximadamente la mitad de ellas se encuentran en el intervalo [-50,74, 11,58] segundos, variaciones inferiores al minuto esperables en algunos momentos de alta carga del servidor. Sin embargo, una cantidad considerable de desviaciones en el rango [-300, -247,68]. Aun habiendo reiniciado dos veces el servidor durante el proceso de recogida de datos y pudiendo esto haber impactado los resultados obtenidos, resulta preocupante el alto número de desviaciones con tal magnitud y nos induce a pensar en la posibilidad de algún error en el proceso de captura que debe atenderse en caso de querer escalar el sistema a un mayor número de *streamers* o de reducir el intervalo.

Figura 33. Histograma de las desviaciones temporales atípicas entre capturas respecto al intervalo. Punto separador decimal. Elaboración propia.

Al agregar las desviaciones por cada emisión y calcular una medida de centralidad como la mediana obtenemos que el 93% de las emisiones tienen una desviación mediana de sus capturas en el intervalo [-0,006, 0,034] segundos (Figura 34). Estos resultados nos llevan a concluir que el rendimiento del sistema es adecuado pero que es necesario revisar el procedimiento utilizado para identificar el origen de esas esporádicas desviaciones con magnitudes fuera de lo aceptable, siendo quizás recomendable desacoplar el sistema en dos componentes: uno encargado de la orquestación de las capturas y otro encargado de realizar las peticiones y las rutinas más intensivas en cuanto a recursos computacionales, para asegurar que la carga de un componente no influye en la planificación del muestreado.

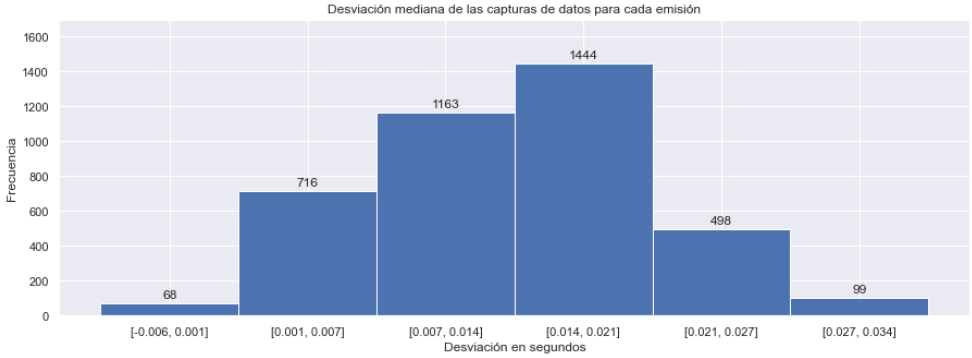

Figura 34. Histograma de la desviación temporal mediana por emisión respecto del intervalo. Punto separador decimal. Elaboración propia.

Anexo III

OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**				X
ODS 4. **Educación de calidad.**				X
ODS 5. **Igualdad de género.**		X		
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**		X		
ODS 9. **Industria, innovación e infraestructuras.**	X			
ODS 10. **Reducción de las desigualdades**.				X
ODS 11. **Ciudades y comunidades sostenibles**.				X
ODS 12. **Producción y consumo responsables.**				X
ODS 13. **Acción por el clima.**				X
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**				X
ODS 17. **Alianzas para lograr objetivos.**				X

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

Twitch es una plataforma en línea de emisión de contenido audiovisual en vivo utilizada por millones de personas diariamente. La presencia de las tecnologías de la información y las redes sociales en nuestro día a día ha ido en constante aumento en los últimos años. Uno de los factores decisivos en esta adopción generalizada de nuevas tecnologías han sido las innovaciones técnicas y científicas. Estas han convertido herramientas que antes estaban al alcance de un segmento de la población muy acotado

en bienes y servicios de consumo diarios para miles de millones de personas en todo el mundo.

Entender la manera en que los seres humanos interactuamos unos con otros en estos nuevos espacios virtuales es vital para comprender una gran parte de nuestras sociedades actuales. Asegurarse de que estos espacios innovadores sean abiertos, tolerantes y contribuyan a la consecución real de la igualdad de género resulta imperante en el contexto actual en el que nos encontramos.

Este tipo de plataformas representan una oportunidad única para desarrollar nuevas maneras de crear y consumir contenido audiovisual de una manera más democrática y abierta. Ayudar a comprender como evoluciona Twitch, que clase de contenido se crea y se consume, como se organizan e interactúan los usuarios, nos ayuda a comprender mejor el mundo en el que vivimos.

El Objetivo de Desarrollo Sostenible más cercano al contenido de este trabajo es el número 9: Industria, innovación e infraestructuras. La evolución constante en la manera de crear y consumir contenido en Internet no para. Las innovaciones tecnológicas nos descubren maneras de relacionarnos que antes eran imposibles de concebir. Estos espacios virtuales entre iguales suponen oportunidades únicas para la consecución de otros muchos ODS. Entender cómo funcionan y estructuran estos espacios puede ayudarnos a alcanzarlos.

En menor medida, este trabajo puede estar relacionado con los Objetivos 5 y 8, ya que tanto la igualdad de género y respeto a los y las *streamers* sin importar su identidad de género, etnia, país de nacimiento o identidad sexual, como el derecho a unas condiciones de trabajo decente y sostenible son debates abiertos en una plataforma en rápido crecimiento con una comunidad de espectadores y *streamers* muy diversa.