Benjamin Wascher, BSc

Identifying Influencers on Twitch with Social Network Analysis

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Computer Science

submitted to

Graz University of Technology

Supervisor

Ass.Prof. Dipl-Ing. Dr.techn. Johanna Pirker

Co-Supervisor

Univ.-Prof. Dipl-Ing. Dr.techn. Frank Kappe

Institute of Interactive Systems and Data Science
Head: Univ.-Prof. Dipl-Ing. Dr. Stefanie Lindstädt

Graz, June 2020

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Date Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Dissertation identisch.

Datum Unterschrift

Abstract

The social component of playing games together, competitive or cooperative, and the potential for every person to do so has increased the numbers of online multiplayer games. Playing video games have become increasingly more popular and with it, the streaming community has also grown. Millions of users have joined different streaming networks. One of these, known as Twitch, presents itself as an interesting field of study. A special characteristic Twitch offers is the social component and interaction between performer and viewer. In this thesis, we want to discover the impact of streamers on their viewership. The focus is set on analyzing streamer behaviors on Twitch and identifying influential streamers in the network. Therefore, a dataset has been collected from Twitch in order to perform social network analysis on it. This analysis provides some network properties which helps to identify key members in the whole streamer network. Several evaluations were performed on the subset of influencers in order to measure the impact on other streamers. Finally, the impact of the influencers is compared with the impact of the randomly picked streamer. The results show that influencers have a greater impact on their follower in regards to follower and viewer count but also gaming behaviors. The following cluster analysis on the influencer data set showed us that playing multiple games addresses more viewer than focusing on a single game.

Kurzfassung

Computerspiele gemeinsam zu spielen, sei es jetzt miteinander oder gegeneinander, aber auch dass die Möglichkeiten dazu für alle Leute offenstehen, haben die Zahlen für Online Computerspiele in die Höhe schießen lassen. Computerspiele wurden in den Jahren immer populärer und somit ist auch die Streaming Community gewachsen. Millionen von Usern sind Streaming Plattformen beigetreten wie zum Beispiel Twitch, was zu einem interessanten Forschungsfeld wurde. Eine spezielle Charakteristik die Twitch bereitstellt, ist die soziale Komponente der Interaktion zwischen Streamer und Zuschauer. In dieser Arbeit wollen wir den Einfluss der Streamer auf deren Zuschauer erforschen. Dafür wurde Daten von Twitch gesammelt, um danach eine Netzwerkanalyse mit den gesammelten Daten zu machen. Mit dieser Netzwerkanalyse können Eigenschaften des sozialen Netzwerkes gefunden werden mit denen man die Schlüssel User in dem Netzwerk identifizieren kann. Einige Analysen wurden an dem Influencer Datenset durchgeführt, um den Einfluss der identifizierten Influencer auf das Netzwerk zu messen. Schlussendlich wurde der Einfluss von Influencern mit dem Einfluss von zufällig gewählten Streamern verglichen. Die Ergebnisse zeigen einen deutlich größeren Einfluss der identifizierten Influencer auf deren Follower in Hinsicht auf Follower, Zuschauer aber auch das Spielverhalten wie der Einfluss der zufällig gewählten Streamer. Die darauffolgende Cluster Analyse an dem Influencer Datenset bestätigt diese Tatsache und zeigt weiters, dass wenn ein Streamer mehrere Spiele spielt, der Stream tendenziell mehr Zuschauer hat als wenn er sich nur auf ein Spiel fokussiert.

Acknowledgements

First of all, I would like to thank my supervisor Ass.Prof. Dipl-Ing. Dr.techn. Johanna Pirker of the Institute of Interactive Systems and Data Scienceat Graz University of Technology. She always supported me with good ideas and guided the work in the right direction whenever it was needed. Also in every meeting, she was encouraging and motivated me to apply the next steps with enthusiasm.

I would also like to thanks Dipl.-Ing. Alexei Scerbakov for his technical support and good advice. Without his help and input, this work would not have been possible. Moreover, great thanks to the company *own3d media GmbH* where Alexei works. *own3d media GmbH* supported us with the needed equipment like cloud computing power and a MongoDB in the cloud.

Finally, I want to express thanks to my wife for providing me with daily support and encouragement. Writing this thesis would not have been possible without your help. Thank you.

And of course to my favourite cousin in the whole world, who made sure the English was comprehensible.

Author

Benjamin Wascher

Contents

1 Introduction

In the past years, video games have become increasingly more popular, and with it the streaming community has also grown. By now, millions of users have joined different streaming networks such as Twitch, which presents itself as an interesting field of study. This thesis focuses on analyzing streamer behaviors on Twitch and identifying influencers in the network. Moreover, the impact of these key members in the network on other streamers is analyzed.

1.1 Motivation

Streaming platforms such as Twitch are becoming ever more popular and more widely used. Streamers want to share their gaming experiences with other users. Since Twitch is a social media platform, we can identify interesting social structures within this platform. One tool to analyze the social structure of a network is social network analysis (SNA). However, looking at literature such as (Harpstead et al., 2019), we can see that only little work has been done with SNA on Twitch. Harpstead et al. (2019) gives an overview of previous analyses which have been done on the platform Twitch. In this listing of different work on Twitch, only two papers appear in which a SNA was used (Churchill & Xu, 2016; Dux, 2018). In this work we want to explore in greater depth the social structure and streaming community on Twitch with a strong focus on influencers. Every community includes key members who are keeping the community alive. These key members are called influencers and are assumed to have a greater impact on the network than other users. Furthermore, we assume that the information flow around these key members is stronger than anywhere else in the network and that they allow for information to be distributed more efficiently in the network

(Sharma & Cosley, 2016). SNA offers the potential to identify influencers in a network, detect network trends, and determine changes in the streaming behaviors. Moreover, the impact on other streamers can be measured and how decisions of influencers change the network. Understanding influential streamer behaviors can give as additional benefits, such as learning more about the network structure, how people communicate, and also if people have an impact on each other. Several social networks have already been analyzed by Ediger et al. (2010), Teutle (2010), Akhtar et al. (2013), Nazir et al. (2008), Ugander et al. (2011) and Leibzon (2016) in order to obtain a better understanding of the network structure. No analysis has been done on the Twitch network with regards to identifying influential streamers.

1.2 Goals and Objectives

In this thesis we are going to analyze the streaming platform Twitch. In order to be able to perform evaluations on the streaming behavior, we need a suitable and current data set. Furthermore, we would like to understand the streamer behavior on Twitch and find influential people. One major research goal is to crawl, clean, and provide a dataset from Twitch for future researchers. The second part of the main contribution is the analysis of the dataset with the focus on answering the following research questions.

RQ1: How can we identify influential streamers on Twitch?
RQ2: How can the fastest growing influencers be determined?
RQ3: What impact do the influential streamers have on other streamers?

1.3 Methodology and Structure

To answer these questions, we first collected a large data set. In the next step we applied a Social Network Analysis (SNA) on the network structure to find out how streamers are related to each other. From these results, we identified influential streamers in the whole network. Furthermore, we defined some metrics which separates influencers from all other streamers.

The work is structured as follows. In Chapter 2 we give the basic background information about the techniques and methods we applied in this thesis. This includes SNA and its background in graph theory. Next, we show the application of SNA and previous research on different social networks. After introducing SNA, we give a detailed overview of the streaming platform Twitch including definitions of important terms that are used in this work. Next, we have a deeper look into Esport and try to understand the behavior of streamers. At the end of this chapter we would like to introduce a method to identify influential people in a network applying SNA. In Chapter 3 we describe in detail how we collected our data set. We give inside into the databases, namely *MongoDb*[1] and *PostgreSql*[2], and the application programming interface (API)[3] of Twitch we used. To finish this chapter, we describe the database structure and collected features. Chapter 4 deals with the processing of the large data set. Problems occurred during the processing of the large data set. Therefore, several steps of optimization were necessary to increase the performance of the processing steps. Besides describing the problems which occurred and solutions to it, we characterize important metrics of the data set. This leads us to Chapter 5 which includes our result of the network analysis. Finally, we list the limitations of our work. Concluding with Chapter 6 we summarize our work and results. Additionally, we give examples of further research.

[1]https://www.mongodb.com/cloud/atlas
[2]https://www.postgresql.org/
[3]https://dev.twitch.tv/docs/api/

2 Background and Related Work

Nowadays gaming behavior includes increasingly more social aspects. While a few years ago only a few games offered a multiplayer mode, many major modern games include multiplayer features and let players interact with other players. The social component of gaming is important. Players are eager to achieve challenges together and play in teams as well as against each other. However, the gaming industry is expanding fast these days and is growing every year. Statistic shows, that by the end of 2019 more than 2.5 billion people played video games across the world (Andre, 2020). The game marked is still growing and therefore its communities as well. Due to the expansion of the internet, new social networks were formed for different purposes, including for gaming. Social networks contain much information and by analyzing its structure, key members of the social network can be identified. Twitch is a social network of streamers, driven by the gaming community. People like to socialize on this online platform and meet their virtual friends there. By now, Twitch is the most popular streaming platform (Iqbal, 2020). In our work, the focus is set on this platform to identify key members, influential streamers, of this social network. The method used in this work to analyze the network is social network analysis (SNA). In the following sections, Twitch and the SNA method are explained in more detail.

2.1 Twitch - The Streaming Platform

Nowadays virtual communities like Twitch, *Facebook*, *Instagram*[1], *Twitter*[2] are growing fast and have several Millions of users every day which is

[1]https://www.instagram.com
[2]https://www.twitter.com

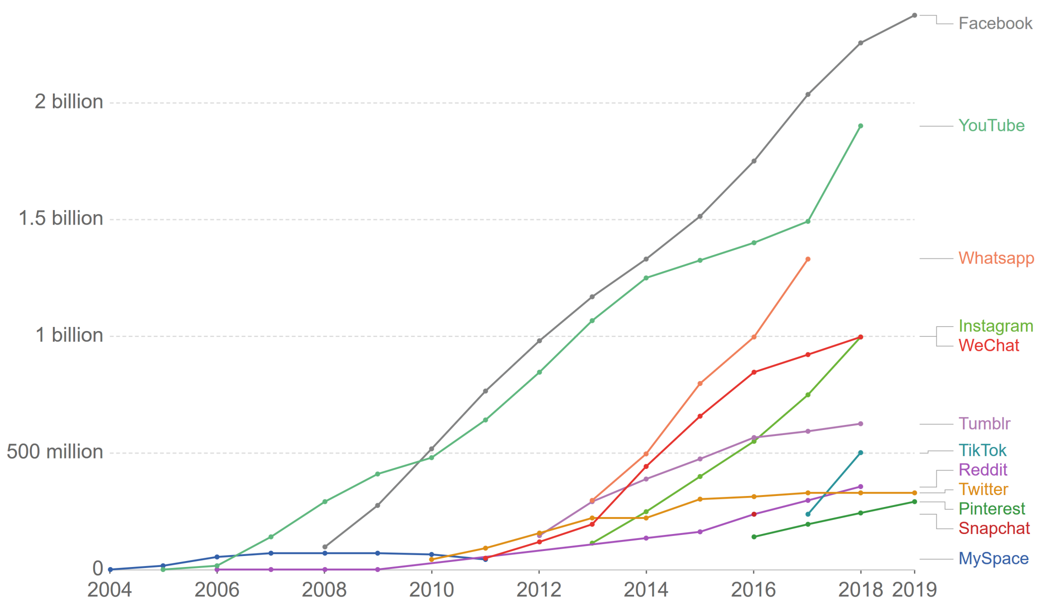


Figure 2.1: This diagram shows the growth of social network from 2004 to 2019. The diagram is adapted from Ortiz-Ospina (2019).

illustrated in Figure 2.1 and 2.2. Social media plays an important role in everyone's life. These online communities are used to get in touch with each other, share information, or play games together. A social network is organized around the user. In social networks, users try to find people with the same interests, discuss topics, and build up relationships. A variety of social networks exist nowadays and every network is aimed at a specific user group, e.g. Twitch[3] for streamer and the gaming community. Live-streaming has become very popular and is a fast-growing phenomenon. Twitch is by now one of the most widely used streaming platforms (Gandolfi, 2016). Twitch is oriented on gaming content; however, live-streams are not limited to gaming and can show performers dancing, teaching, talking, or eating food. Furthermore, viewers can interact with the performers in real-time. This creates a social connection between the live stream performer and the viewership where viewers can influence the streamer (Sjblom & Hamari, 2017).

[3]https://www.twitch.tv

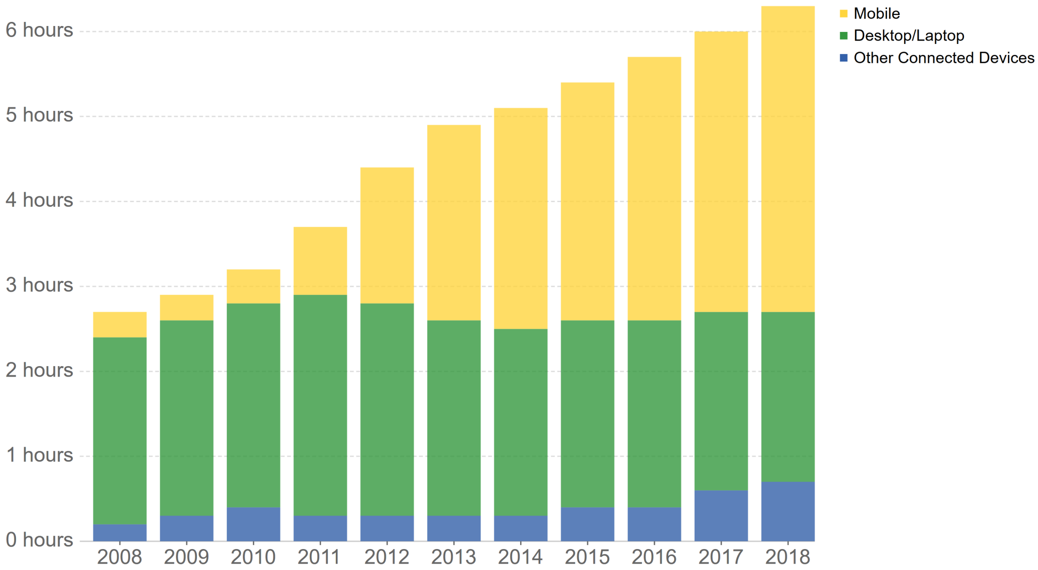


Figure 2.2: Showing the hours per day spend using digital media in the United States. The diagram is adapted from Ortiz-Ospina (2019).

In the following sections an overview of the streaming platform Twitch is given. Furthermore, Twitch-specific definitions and terms are explained in more detail. Another increasingly important media which has a big role on streaming platforms is Esports. This is an important topic in our work due to the great impact of Esport events on both streamer and viewer behavior of Twitch users. Finally, a brief introduction into streamer behavior is given which deals with the question *Why people stream on Twitch*.

2.1.1 Twitch

In gaming communities, Twitch is already well known as a live streaming platform focused on video games. The idea behind Twitch is to broadcast and share games and make them publicly accessible. The web site Twitch was launched in 2011 as a streaming platform and already in October 2013 it had 45 million visitors. The platform caught *Amazon*'s attention and

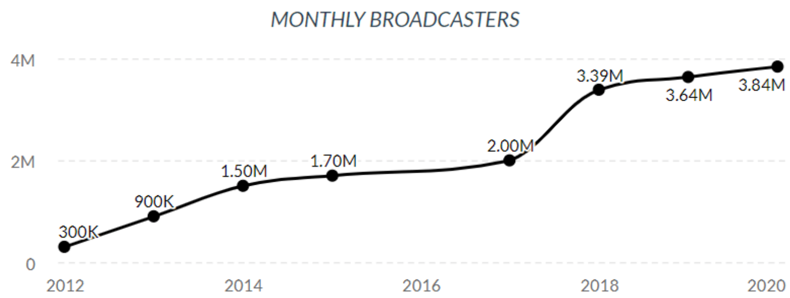


Figure 2.3: The diagram shows the monthly broadcaster count and the growing popularity from 2012 to 2020. The diagram is adapted from TwitchTracker (2020c).

Amazon[4] purchased Twitch with $970 Million in 2014 (Zhang & Liu, 2015). More and more players started to stream their games and within a short time Twitch became popular. As it can be seen in Figure 2.3 Twitch has grown continuously since 2013. In other words we could say the current count of the monthly broadcaster on average is nearly twice as much as in 2017 and four times as much in the year 2013.

Not only has the broadcaster count increased during the years but also the viewer count. While on average 208,000 viewers watched on Twitch in 2013, those numbers increased over the years to 1.73 Millions (Figure 2.4). Several peaks of the viewer count could be recorded in this time. In April 2020, the viewer count reached more than 4 million viewers, for the first time in Twitch history, which is an increase by more than 50% compared to the numbers recorded for March 2020 (TwitchTracker, 2020c).

Compared with other streaming portals such as *Mixer*[5] or *YouTube*[6], Twitch has by far the highest viewer counts over a long period (Pires & Simon, 2015). Even when streamers leave from Twitch to another platform, the viewer counts remain steady on Twitch. Figure 2.5 shows the number of viewers of different streaming platforms.

A reason why Twitch has been so successful over the years and still is,

[4] https://www.amazon.com/
[5] https://www.mixer.com/
[6] https://www.youtube.com/live

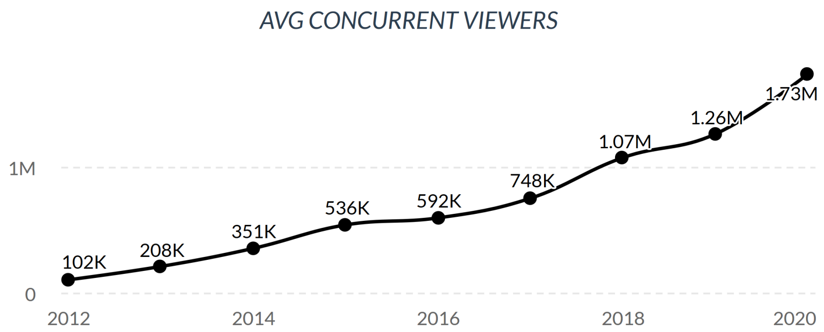


Figure 2.4: The diagram shows the monthly viewer count watching streams on Twitch. The numbers are recorded from 2012 to 2020. The diagram is adapted from TwitchTracker (2020c).

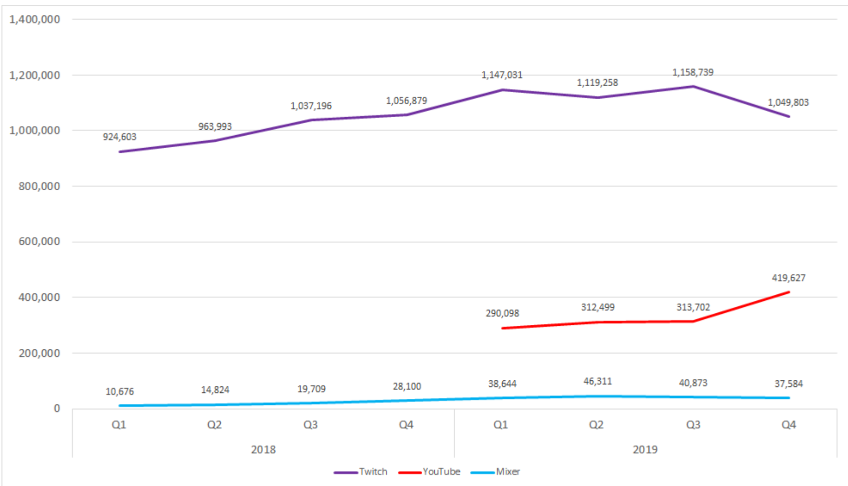


Figure 2.5: This diagram compares the viewer count of *Mixer*, *YouTube* and Twitch. The diagram is adapted from TwitchTracker (2020c).

lies in its social component. On Twitch, not only can videos be watched, there is also an interaction between the viewer and producer of the content. Bründl et al. (2017) pointed out that it is exactly this interaction which attracts the viewers. In other words, it could be said that it is like a virtual gathering. Viewers normally watch a variety of different streams not focused on only one streamer or game (Dux, 2018). Several surveys of Twitch's social structure were made and the results show that people are interested in content they to which they can relate. Most people who started to watch streams on Twitch, were interested in a game they played by themselves. As mentioned before, the social aspect is an important factor on Twitch. Streamers interact with their audience through a chat or by directly speaking to them. A *channel* of a streamer also gives the viewers the opportunity to communicate with each other and discuss topics. A huge impact on the viewer count of Twitch and also its gaining popularity throughout the last years is caused by electronic sport (Esport) events. These are competitive tournaments at a professional level which attract many people. This topic is addressed in Section 2.1.3.

2.1.2 Definitions

In this section we collected all the important Twitch terms relevant for our thesis and explained their meaning. Furthermore, the user interface of the streaming platform is described.

User vs Streamer on Twitch

In this work, we distinguish between common users and streamers on Twitch. A common user is defined as a registered Twitch user who has an account on Twitch but none or almost no streaming activities. In general, these users are only viewers and watch other streamers. In our thesis, a registered user is defined as a streamer when the state *affiliate* is achieved. According to Twitch (2020a), several requirements are necessary to reach this goal:

- Reach 50 Followers

- Stream for at least 8 hours
- Stream on 7 different days
- Have an average of 3 concurrent viewers

These requirements must be fullfilled within a 30-day period. The group of *affiliate* users will be the most important for us. Beside the *affiliate* state the highest state to achieve as Twitch user is *partner*. Currently around 41000 streamers have joined the *partner* program of Twitch (Iqbal, 2020). Besides these two main groups we discovered during our work another subsection of the *partner* program, called *oCPM*. Cost per mille or short *CPM*, is a commonly used measurement in advertisement and defines a rate an advertiser must pay for each 1,000 views (Twitch, 2019). By examining this user group it turned out that game studios like *EA*[7] or *Ubisoft*[8] own these channels and use them for example game releases. Furthermore, game leagues such as *ESL*[9] or *Overwatchleague*[10] perform their tournaments on these channels. In general, the viewer count of these channels is at several million views, and the follower crowd is also relatively large. In this work, the focus is set on analyzing streamer behaviors. Therefore, this category is excluded from our analysis due to the fact that these channels do not correspond to typical streamer behavior and with their high viewer counts and followership they would influence the results.

Follower vs. Follows

On Twitch, users who broadcast live streams and share their content are referenced as *streamers*. A streamer owns a channel which is accessible to users. All users, registered at Twitch or not, who watch live streams on Twitch are called *Viewers*. Registered users can *follow* their favorite streamers. On the other hand, *followers* are users who follow a streamer. In this paper, the *follower* count is always referenced as the incoming connections and the *follows* count as the outgoing connections in the social network graph.

[7]https://www.ea.com
[8]http://ubisoft.com/
[9]https://www.eslgaming.com/category/counter-strike
[10]https://overwatchleague.com/en-us/

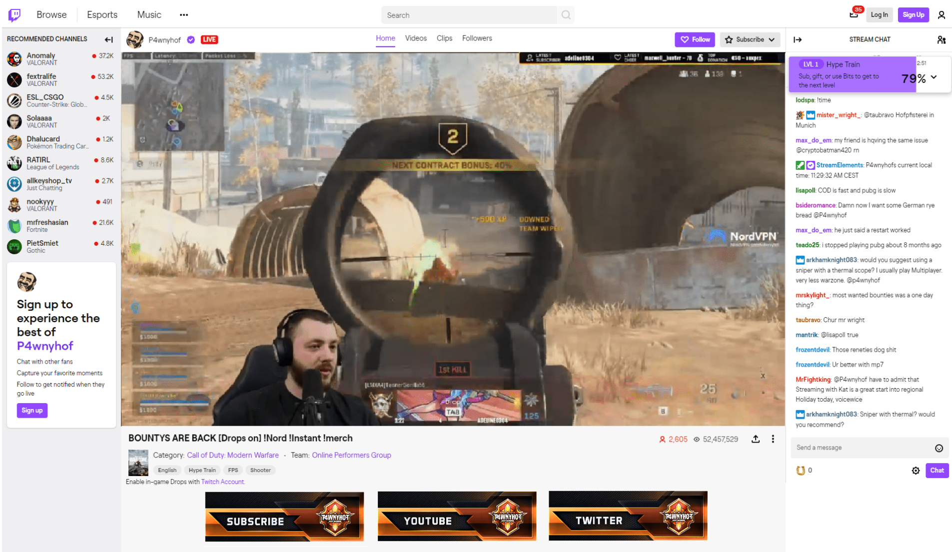


Figure 2.6: A screenshot of a live-stream from Twitch partner *p4wnyhof*.

Channel

A user's channel is the platform where the live streams are viewed. Moreover it contains general information about the user's profile, videos, clips, and also a list of followers. In addition, total viewer count and current viewer count are also publicly visible. *Affiliate* or *partner* streamer include advertisements of their sponsors in their channels.

User Interface

Figure 2.6 show the user interface of Twitch currently showing *p4wnyhof*[11] playing *Call of Duty: Modern Warfare*[12]. The streamer *p4wnyhof* is part of Twitch partner program. The user interface of Twitch is structured according to the following parts: the main screen shows the current live stream of the game which is an identical copy of the streamer's play screen. Additionally, streamers can use a webcam and which is shown on the Twitch live stream.

[11]https://www.twitch.tv/p4wnyhof
[12] Infinity Ward , 2019. https://www.callofduty.com/modernwarfare.

This is part of the social component of streamers where viewers can see the performer. Furthermore, at *affiliate* streamer advertising or recognition to a user who supported the streamer sometimes pops up and is visible to the audience. On the right side of the screen is the user chat where registered users can comment on the streamers game-play or discuss topics. Very often, the performer respond to these messages to make the live stream more attractive to the viewership. On the left-hand side of the screen is a list of recommended and popular channels. Below the mainstream window, some channel data is listed like game category, team, different tags to find the stream, total views of the channel, which is currently about 52.46 Million, and the current viewer count with 2,605. Below there is a space in the channel to place advertisements and list sponsors.

2.1.3 Esport Events

Besides individual broadcast streams, Twitch also offers live streams of the world's most popular Esport events. Esport is a form of sport in which professional gamers play competitive tournaments. These tournaments in the video game culture have existed for almost 20 years but through live streaming, these events became much more popular and the community has grown fast. Nowadays, Esport events are very popular and play an important role in the streaming world. As an example, in 2018 Twitch hosted the finals of *Fortnite*[13] with a total prize money of $10 Million (Iqbal, 2020). In the following year, the game *League of Legends* achieved the highest concurrent viewer count during one stage. The highest price money was offered by the popular game *Defence of the Ancients 2*[14] (Dota 2) with $25.5 Million in 2018 and $34 Million in 2019 (Earnings, 2020a). Esport events are widely acclaimed by audiences. The prize money is comparable to traditional sport events but also the size of the audience. Figure 2.7 illustrates an estimation of how big the viewership in the US could be in 2021.

Supported by the great popularity of Esport events and the growing live-streaming communities, the Esport audience is growing every year. Due to the high viewership size, companies and organizations are interested

[13] Epic Games , 2017. https://www.epicgames.com/fortnite.
[14] Valve , 2013. https://www.dota2.com.

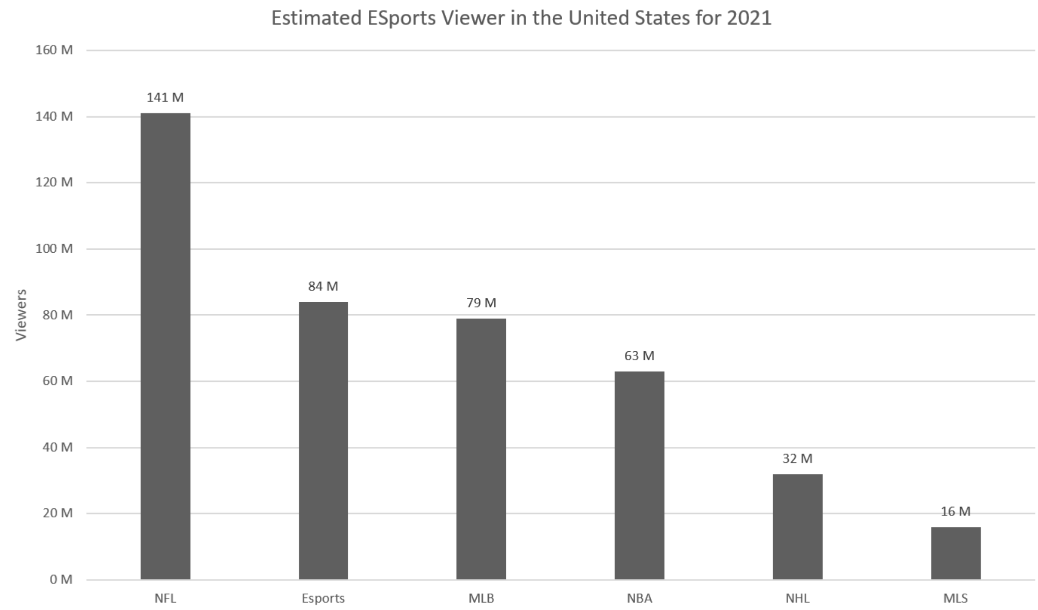


Figure 2.7: This chart illustrates an estimated viewer count (in Millions) for big sport events in the United States in 2021. This chart is adapted from Staff (2019).

Rank	Game	Number of Tournaments	Total Prize Money	Player Count
1.	Dota 2	1,348	$223.5 Mio	3,600
2.	Counter-Strike: Global Offence[15]	4,861	$95.1 Mio	12,685
3.	Fortnite	537	$85.2 Mio	3,283
4.	League of Legends	2,426	$74.5 Mio	6,883
5.	StarCraft II[16]	5,776	$32.8 Mio	2,033

Table 2.1: This listing shows the best payed tournaments (Earnings, 2020b).

in hosting and sponsor Esport events. The most popular video games in tournaments are listed in Table 2.1. According to hosting Esport events with their large audience, Twitch has gained in popularity. However, not only Esport events and tournaments are streamed on Twitch but also some traditional sport events are starting to live-stream their games on Twitch.

Professional players and teams who perform in tournaments often receive a partnership offered by Twitch and broadcast regularly. Professional players are highly attractive to viewers and have a high number of viewers during their stream time. Thus Twitch ensures to provide an entertaining community as well as a professional level to its streams. Juho and Max (2017) are focusing on the question of why people watch Esport. The results show that some people want to escape everyday life while others want to acquire knowledge from Esports. Furthermore, novelty and enjoyment of watching competitive video games are also a strong motivator to enjoy Esport events. Esport games are usually complex and some previous knowledge and concentration are necessary to follow the game flow. Therefore, some users focus on the aesthetic part of games while other focus more on the technical and rule-based proceedings. In addition, the study shows that the enjoyment of Esports is related to an unexpected and dramatic turn of events. People enjoy it if a team can manage to win a game against the odds.

In summary, people enjoy watching Esports and the viewer crowd is still growing. After better understanding the viewer's behavior and motivation of viewers we will now focus on the streamer behavior in the next section.

2.1.4 Understanding Streamer Behaviour

Often the question arises *Why do people broadcast live streams*? Live-streaming is a new form of entertainment where viewers and streamers can interact with each other. But why do people prefer to watch other people rather than doing it on their own? Hilvert-Bruce et al. (2018) answer this question in their work. An important aspect is the social component and the streamer/viewer relationship. People want to share their experiences with friends, talk about games, and be accepted by a community. Live streams have become a virtual place to meet friends. In chat rooms, users talk and joke about the content they are watching. Therefore, viewers are attracted to channels in which they are noticed and can influence the streamer. Another important aspect is entertaining the audience. Many professional streamers on Twitch attend Esport events or broadcast professional game streams where viewers can learn from professional players. Besides professional gamers, streamers try to play games with different focuses to attract viewers. The different gameplays attract different viewers. Therefore, some streamer provide game previews to present a new game and respond to viewers questions who are interested in buying the game. Other live streamers try to finish games connected with special challenges and show the viewer how to be successful. These challenges could be to finish the game as fast as possible or only with limited resources in a short time. Another favored category are gameplay walk-throughs in which streamer show, in general, how to play the game. Most of the time special secrets or secret missions are presented, not known to all gamers.

Zhao et al. (2018) also worked on the streaming behavior of Twitch users and achieved similar results with regards to the social component. In this work, the question of why streamers *continue* broadcasting is answered. The authors found out that a good performance of live streams is related to the motivation of streamer which is true in the case of Twitch. Generally speaking, Twitch is a well-designed platform and even streams with millions of viewers, offering good performance without technical issues (Pires & Simon, 2015). This makes Twitch more attractive to streamers and viewers which also has a big impact on the streaming behavior. Not only is the performance important, but also what Twitch is doing for their broadcaster. In order to increase the satisfaction of the live-performers needs, Twitch hosts tournaments and launches challenges with special prices (Zhao et al., 2018). Moreover, Twitch also offers interaction between the performer and the audience such as likes, virtual gifts, or donations for encouragement. The streamers on the other hand, try to obtain these gifts from the viewers and reward them (Bründl & Hess, 2016). Besides users encouraging streamers with gifts, Twitch offers marketing opportunities through advertising and sponsoring. Popular streamers with a large follower crowd are often sponsored by game studios with gaming equipment. In return, the streamers show the name of the sponsor during the broadcasts (Zhao et al., 2018). Through the high popularity of Twitch and the entire Esport scene, advertising has become a major part of live steaming. Therefore, streamers fight for their popularity and viewership to obtain sponsors. Many streamers try to increase their number of viewers by attending Esport events. All in all, these dynamics present a well working system. Furthermore, it is a strong motivation for streamers to continue broadcasting and keep their streams attractive to their audience.

As we heard before, streamers try to keep and increase their viewership. The social component and interaction with the audience is also an important component for a successful streamer. Generally spoken, streamers are broadcasting different contents from casual games to educational channels. As we can see, and according to Smith et al. (2013), the reason to broadcast is driven by different motivations. In order to get a better understanding of the streamers' behavior and network structure, we discuss SNA as a method to analyze a social network.

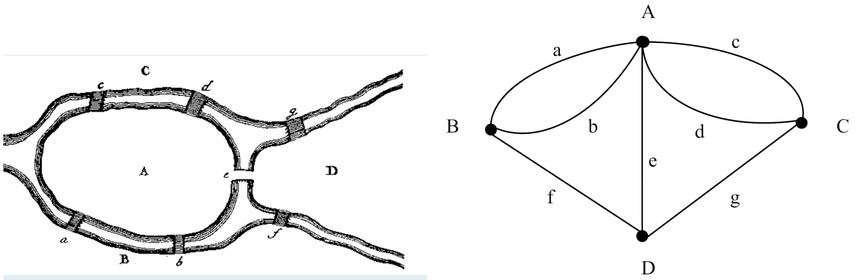
2.2 Analyzing a Network

Twitch is a network of streamers and viewers. They can interact with each other and play games together. To analyze this network it is important to understand how streamers are connected and interact with each other. Therefore, the focal point is set on the friend list of streamers. Every registered user on Twitch can follow other streamers. Based on these connections, a network can be created to see who is following whom. With a tool like SNA, we can analyze the social structure of this network in order to find key members. Therefore, the focus is to find these streamers, called influential streamers or in short, influencers. A method suitable for analyzing streamer behavior in the network is social network analysis (SNA). This method is based on the relationships between users and enables the ability to identify key members of the network. SNA was introduced by Tichy et al. (1979) to map a social network in a graph model. In this model, every user or streamer is represented as a node, also called a vertex, in the graph and their relations to each other as edges, or links, connecting these nodes. Transform a social network to a graph model brings several advantages through the use of networks and graph theory. In order to find the key metrics in the streamer network these mathematical calculations can be applied on the social network. Before we step deeper into SNA we give a summary of graph theory in the following section.

2.2.1 Graph Theory

The first paper written about graph theory was *The Seven Bridges of Königsberg* by Euler (1741). The city of *Königsberg* is a city, divided by a river in two large islands and the mainland. The islands are connected but also to the mainland by seven bridges. However, the problem Euler wrote about was to walk through the city by using all bridges once and only once. The relevant information is the connecting bridges, represented as edges, between the parts of the city, appearing as nodes. This is illustrated in Figure 2.8a and Figure 2.8b.

This problem maps a geographical structure, shown in Figure 2.8a, to mathematical graph (Figure 2.8b). With this model, called graph, Euler



(a) A geographic representation of *Königsberg*. A and D marked as the islands connecting the main land B and C by the bridges a - g.

(b) Königsberg mapped to the graph. The vertices A to D representing the islands and main land. Edges connecting the vertices are symbolizing the bridges.

Figure 2.8: The *Königsberg* example is adapted from Schubert (2012)

showed that the possibility of crossing all bridges only once depends on the degree of the nodes. This model can also be applied in our case to analyze the behavior of streamers and their connection to each other. Every streamer is represented as a vertex and all his followers and follows are the connection to other streamers, represented as an edge.

2.2.2 Graph Definitions and Properties

After mapping a social network into a graph layout, some important properties used to analyze the network, are introduced and described in this section. In general, a graph G is defined as an ordered pair of vertices and edges $G = (V, E)$ where V is a unique set of vertices and $E \subseteq x, y | (x, y) \in V \wedge x \neq y$ represent a set of vertices which define an edge from one vertex to another one. A graph can be **directed** or **undirected**. In other words, a graph is called directed if an edge is oriented which indicates the direction is important. In opposition, as in the example of *Königsberg* (Figure 2.8b), the only importance is if nodes are connected. Hence, an undirected graph is used in which an edge has no orientation. In this paper, the information of which direction the relationships of our streamer are going is important. Therefore, we reference only directed graphs in this work.

Path: A path contains an ordered sequence of vertices which are connected by edges:

$$v_1, e_1, v_2, e_2, v_3, e_3$$

Many problems are related to finding the shortest path in a graph. It is an attribute to determine how well a graph is connected.

Density: Density is a value in the range of $0 \leq \text{density} \leq 1$ and indicates the percentage of relationships in a graph (Yuan et al., 2018). It takes the number of the existing connections of all nodes into account and is divided by the maximal possible connections to all other nodes. The density value of the graph $D(G)$ is defined as

$$D(G) = \frac{K}{N \dot{(} N - 1)}$$

where K is the existing number of relationships and N the total number of nodes in the graph. A higher density value indicates a more complex graph structure.

Strongly Connected Components: A strongly connected graph is defined as a graph, in which each vertex is reachable from every other vertex (Nuutila & Soisalon-Soininen, 1994). This is illustrated in Figure 2.9. In other words, every node is connected with the main graph. The edges used in the path of a directed graph can only be used in the right direction.

Largest Connected Components: The largest connected component is the biggest strongly connected sub graph. In the example of Figure 2.9 the graph is divided in three components. Therefore, the largest connected component of this sample graph is simply the largest component on the left side.

Closeness Centrality: The idea behind the closeness centrality is to assign nodes a high centrality values that are closer to all other nodes (Cohen et al., 2014). In other words, a node with a low distance to all other nodes is more central. Bavelas (1950), Beauchamp (1965) defines the closeness as

$$C(x) = \frac{N - 1}{\sum_{y \in V} d(y, x)}$$

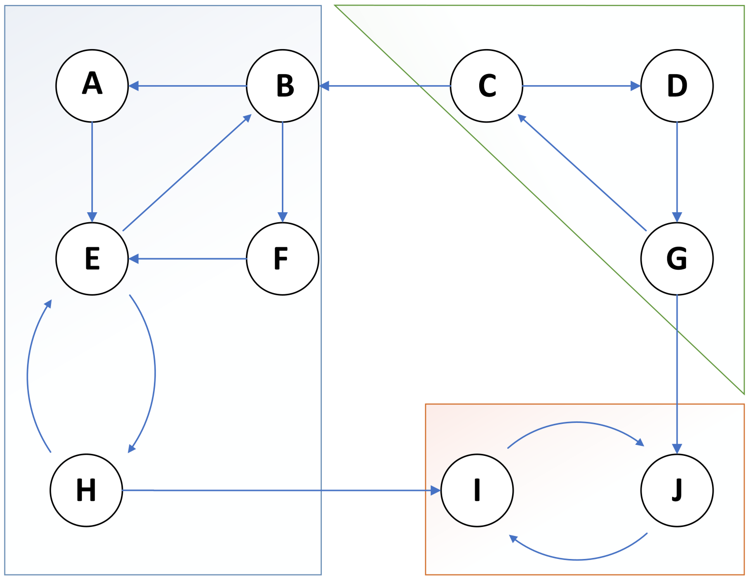


Figure 2.9: Connected components: This graph contains three connected components in which every vertex is access able from every other vertex. The largest connected component is the component on the left site containing the vertices *A*, *B*, *E*, *F*, *H*.

where $d(x, y)$ is the distance between the nodes x and y and N the total count of vertices in the graph. This value indicates an important measurement of centrality.

Betweenness Centrality: This algorithm measures the centrality based on the shortest path. Vertices with the shortest path from one user to another going through the vertex receive a higher score. Therefore, it can be assumed that nodes connected to these edges have a larger influence on the network flow (Freeman, 1977).

Eigenvector Centrality: As a concept for this method, connections to higher-ranked nodes receive a higher score than lower-ranked nodes. Therefore, a high score is reached by being connected to *important* nodes. The Google Page Rank algorithm is a variation of the Eigenvector centrality (Maharani et al., 2014).

Pagerank: Pagerank is an algorithm first used by Google to order search results (Page et al., 1999). The importance of a node depends on its in-degree, the number of incoming links. All outgoing links are weighted with the score of the node. Therefore, nodes with a high in-degree are ranked with a higher score and are identified as influencing nodes (Canossa et al., 2019).

Degree Centrality: The degree of a vertex v is the sum of incoming and outgoing connections of v. Calculating the total degree of a graph, each edge is counted twice - once for each end of the connection. In a directed graph, the degree can be divided into in-degree and out-degree. For the in-degree, only incoming edges are taken into account. In a social network graph the in-degree represents the followers of a user. The out-degree counts only outgoing edges which give an indication of the number of follows of a user in a social network graph (Srinivas & Velusamy, 2015).

A basic understanding of graph theory was given to highlight the important properties of a graph and to apply this knowledge to social networks. In the next section, SNA is introduced in more detail and its field of applications in different areas is mentioned.

2.2.3 Social Network Analysis (SNA)

SNA is a method to analyze a network structure or a community based on the earlier mentioned graph theory. Tichy et al. (1979) introduced this method and it found various applications in analyzing different networks. In the work of Krause et al. (2007) various applications of SNA are presented in the field of behavior science such as disease transmission and information transfer. One of the most important application fields are obviously social networks such as *Twitter* (Ediger et al., 2010; Teutle, 2010) and *Facebook*[17] (Akhtar et al., 2013; Nazir et al., 2008; Ugander et al., 2011). These social networks, mentioned before, provide a huge amount of public accessible data. Therefore, it is interesting to analyze these networks. The SNA applied on *Twitter* deals with the whole network structure of *Twitter* and it's dynamic.

[17] https://www.facebook.com/

By calculating key metrics of the network graph, the authors were able to provide a better perspective of the network expansion. Similar is the analysis of the *Facebook* network. By using a dataset, key metrics are calculated and analyzed to define the structure and features of the network. As a result, some user behaviors are defined based on the network structure. Beside the use of SNA in traditional social networks it was also used for the version control *GitHub*[18] by Leibzon (2016). In this work, the software development supporting the platform *GitHub* was analyzed based on SNA methods. By analyzing open-source software projects, the authors can determine the health and success of a project. In contrast to online web sites, SNA also found application in the analysis of team sports (Lusher et al., 2010) or economic geography (Ter Wal & Boschma, 2009) in which relations are analyzed and some behaviors are found based on the structure of the network graph.

As we can see, SNA is used in a wide variety of applications across different areas. With the growth of online gaming, SNA found also application in this field. Social networks in games play an important role. Our work is focused on the analysis of the social network Twitch and the streaming behavior of gamer. As already mentioned, SNA is a method to measure and evaluate the development of a network structure. Algorithms can show network trends like most played games on Twitch or discovering influencing users in the social network. In general, SNA shows special characteristics but also changes in the network. Before an overview is given to the previous analysis of Twitch, insight is given into how SNA can be applied to games in the next chapter.

2.2.4 Application of Social Analysis in Games

The social structure of games have become more interesting to analyze. Understanding player communities, detecting player formations, and interaction can improve the game experience but also game development. Therefore, some examples of SNA applied to some popular games are listed.

[18] https://www.github.com/

SNA on Tom Clancy's The Division[19]

As has been done before on other games such as *World of Warcraft*[20] or *League of Legends*[21], Canossa et al. (2019) applied SNA to identify potential influencers in the online multiplayer shooter *Tom Clancy's The Division*. When social networks started to grow, SNA found large applications in identifying influencers and key members in a network. In the last years, the game communities have also expanded and become an interesting target for social analysis. There is no general rule of how to identify an influential person. Therefore, some rules must be defined for users to be qualified as an influencer. In the work of Canossa et al. (2019) measurement of the entertainment value is the time spent playing the game and social play, the time playing with other gamers. By applying these restrictions on the sample data with over 14 million players, a subgraph was gained from all the samples. For identifying potential influencers, six different measures of centrality were calculated: closeness, betweenness, eigenvector, in-degree, out-degree, and Pagerank. Only players, satisfying all these six conditions were chosen as an influencer. The resulting subset contains 49 players. Some analytical tests were made focusing on the influencer subset and the authors found out that these 49 players have more impact on the behavior of other players than other users.

Network Visualization in Dark Souls 3[22]

Gandolfi (2017) visualize the network of the action third-person shooter *Dark Souls 3*. For this analysis, the game *Dark Souls 3* was chosen as a testing ground due to its success, similar features, and temporal closeness. The dataset was collected manually. Over a period of time, every day at midnight the top 10 streamers on Twitch were recorded by video-recording software. For every stream, only three to five screenshots were stored. From this screenshot, information about the mood of the streamer, area of the game,

---

[19] Massive Entertainment , 2016. https://www.ubisoft.com/en-us/franchise/the-division/.

[20] Blizzard Entertainment , 2004. https://worldofwarcraft.com/.

[21] Riot Games , 2009. https://www.leagueoflegends.com/.

[22] FromSoftware , 2016. https://en.bandainamcoent.eu/dark-souls/dark-souls-iii.

current viewers, and type of performance are gained. The same data was collected on the platform *Steam*[23]. The resulting information was visualized with *Gephi*[24]. Some peaks of viewer count were detected due to the presence of popular streamers streaming the game. Furthermore, some behavior of streamers was discovered during the analysis. In the case of *Dark Souls 3* a streamer often starts the first play with a play-walk-through. After getting some experience with the game-play, new strategies are tried out. Finally, a streamer joins competitive multiplayer games or tries to finish the game as fast as possible. Another important impact on the viewer count is the performer who attracts the viewer.

The previous example shows the importance of SNA in-game communities. SNA in games focuses on the association of players in games and around the playing activity (Jia et al., 2015). Players and game developers can profit from the resulting information.

2.2.5 Application of Social Analysis on Twitch

Many social networks have been analyzed such as *Twitter* and *Facebook*. Due to the large availability of digital data, social networks are often analyzed. In our work, the focus is put on Twitch to find key members in the streaming community. In the following section, some examples are listed of previous social analysis of the streaming platform Twitch.

Professional streamer on Twitch

The authors Kaytoue et al. (2012) analyzed Twitch audiences. To obtain an appropriate dataset, a crawler collected data from active live streams with their number of viewers in a time interval of five minutes. The resulting analysis from this data is where in the world the most streams were started and a weekly overview on which weekdays and at what time the most viewers are online. Similarly, a histogram shows the influence of the viewer

[23]https://store.steampowered.com
[24]https://gephi.org/

count during Esport events such as *Blizzard Cup*[25]. Another challenge answered in this paper is the prediction of streamers' popularity and ranking them. Due to the listing of the streams with the most viewers on the main page of Twitch, streams with low popularity have the main disadvantage of becoming popular. As mentioned above, the viewer count also depends on the weekly streaming time. All this was taken into account by calculating the ranking. Three different methods were used and compared with all its advantages and errors. The results show that the future audience of a stream session can be predicted. Furthermore, a ranking between the popularity of the streamer can be created by applying a Condorcet method.

Player / Game relationships on Twitch

Another research about the social structure of Twitch was made by Churchill and Xu (2016). Motivated by the large growth and popularity Twitch gained in the last few years, the authors decided to analyze the player and gaming relationship. The dataset of this research was collected by using the Twitch API service. To study the franchise, a small subset of twenty-one franchises was chosen but excluding some larger ones due to hardware limitations. A selected set of streamers who can be identified with a franchise are taken into account. Many streamers were found through leader boards or Twitch statistic websites. By using the Twitch API, additional information is gathered such as followers. This graph was stored in graph definition file (GDF)[26] format and rendered with *Gephi*. The visualized results show that products of the same or similar companies address similar audiences. Furthermore, subcultures were also analyzed and the results show that the subculture *Causal Games* has a larger audience due to the simplicity of the games.*Causal Games* are often played for fun and are easy to play. In *Speed Runners* or *Competitive Games* some difficulties are involved to become a good player. Therefore the community is accordingly smaller.

[25] https://esports.blizzard.com
[26] https://gephi.org/users/supported-graph-formats/gdf-format

Audience Influence in Gaming Live-Streams

Lessel et al. (2017) analyzed in their work the impact and influence of a live streamer audience on the performing streamer. The analysis is performed on two different streaming platforms, namely *Rocket Beans TV*[27] and Twitch. The study on Twitch focuses on the game *Pokémon*[28]. In order to get a better understanding of the influence of the audience in live streaming games, the selection is not made of the *mainstream* channels, but rather of the smaller channels. For this experiment, the authors created a channel where the audience alone decides the course of the game. This type of gameplay is called *Twitch Plays*, due to Twitch users dictating the course in the game through voting. This type of channel became popular on Twitch and by now several *Twitch Plays* streams are available. In this study, the behavior of the users is investigated and how they become organized in order to achieve a common goal. *Twitch Installs Arch Linux*[29] was an experiment to install *Arch Linux* controlled by the viewership, which succeeded. By inspecting the chat history during the project, the user crowd could be grouped in several roles like *trolls* who are posting off-topic content. In general, the audience is organizing itself to master these challenges together. An individual viewer might not influence the course of the stream but grouped together, the streamer's behavior can be affected.

Overview on studies performed on Twitch

A more detailed overview on papers analyzing Twitch is published in the work of Harpstead et al. (2019). The authors picked over 40 papers and articles out of a large collection published between 2012 and 2018. All papers focus either on a stream, viewer or platform analysis. A table shows a listing of these papers with the focus on what the paper is analyzing, the method used to achieve results and how the data was collected. The conclusion of this summary paper is that there is a large space of possibilities in the game stream research area on Twitch. The examples listed above shows that SNA

[27]https://rocketbeans.tv/
[28] Game Freak Inc. , 1996. https://www.pokemon.com/.
[29]https://www.twitch.tv/twitchinstallsarchlinux

	Focus	Method	Use Case
Kaytoue et al. (2012)	Streamer	Condorect Method, SNA	Social and behaviour science, Rank streamer by popularity
Churchill and Xu (2016)	Streamer, Game	SNA	Player behaviour on different games, identify player communities
Lessel et al. (2017)	Streamer, Viewer	Statistical Analyze	Determine the influence of audience on a streamer
Harpstead et al. (2019)	Twitch General	Literature Review	Overview of 40 different studies performed on Twitch

Table 2.2: This table lists a overview of papers and shows the different methods used to analyze Twitch

is a method to measure and evaluate a development of a network structure. Algorithms can show network trends like most played games or discovering influencing user in the social network. The results in Table 2.2 shows the variety of research fields in which SNA is used.

2.3 Identifying Influencer

In every community are key members who are keeping the community alive. These key members are called influencer. A user, identified as an influencer, is a person who influences the behaviour of other people. Studying influencers in a network helps to better understand certain patterns and trends in social networks. Marketing companies especially are interested in the success of their advertisements. Feeding influential key members with customized advertisement makes marketing more efficient due to the higher impact on the network than through a common user.

2.3.1 Overview

In general, there is no formula or measure to find influential users in a community. This makes it harder to find influential pattern in networks (Srinivas & Velusamy, 2015). Nevertheless, several studies on influential persons on social networks like *Twitter* have been done by Canossa et al. (2019), Riquelme and González-Cantergiani (2016), Srinivas and Velusamy (2015). Facing the problem that every network is different and no general formula can be applied for all networks, every study uses different methods to achieve their goals.

Teutle (2010) analyzed properties in the *Twitter* network in order to detect trends in the rapidly growing social network. By observing the network dynamics and information flow, key members, also called influencer, in the network can be identified. Often, these key members in a network are identified by applying SNA and centrality measures on the network. Recommender systems identify influencers in networks in order to suggest user customized material based on the users activities. A study on the movie online streaming platform *Flixter*[30] pointed out that recommendations work because of social influence (Jamali & Ester, 2010). People tend to relate to people with similar interests and therefore the influence of these people becomes stronger. The social aspect can be found in all social networks, independent of the type or topic the network is focusing on.

Cha et al. (2009) studied the image hosting network *Flickr*[31] to find out how wide and quick information is distributed in this social network. The findings of the research proves that the dominant method to distribute information through a network are social links between users. A *Twitter* study by Romero et al. (2011) confirms the results from Cha et al. (2009). In *Twitter*, the use of hashtags of an individual user depends on the usage pattern of their connected neighbors. Moreover, different types of tags affect the mechanism of the spread of information in the network. Another study based on *Twitter* data was introduced by Lalani et al. (2019) targeting Indian politicians. Information about who is following which politician is valuable. Due to the high degree of their popularity, they have a major impact on their followership. As we have seen, identifying influencers has found broad applications in social networks for various tasks.

2.3.2 Identifying Influencer

Sharma and Cosley (2016) mentioned the difference between personal preferences and social influence in several online social networks based on activity feeds. Some user just mimic or copy influential material and get high attractiveness by following users but the originator of this content and real influencer is in the background. This shows the difficulty in correctly identifying and choosing influencers in a network. Generally, becoming an influencer in a network takes some effort.

Cha et al. (2010) pointed out that the way to become an influencer is hard and requires a lot of personal investment. It is not done by only achieving a high follower crowd but also by inspiring people. Influencers can attract people in different areas and therefore influencers can be identified for various

[30]https://www.flixster.com/
[31]https://www.flickr.com/

fields. Riquelme and González-Cantergiani (2016) divided influencers into several groups like *inventor* for people who start new topics or discussions, *idea starters* who are connected with many followers, *connectors* who are connecting *idea starters* and many more. Furthermore, influencers have an impact on other users in the network in different ways. Therefore, users with a high follower crowd are not necessarily influential people. Likewise, Cha et al. (2010) achieved similar results by identifying influential users of Twitter. The authors mentioned that the in-degree alone, in other words the follower count, represents little information about the influence of users in the network.

In our work we used SNA to identify influencers. The calculation is based on centrality. The idea behind our approach is to find the most central streamer in the network by calculating different centrality measures. As mentioned earlier, we used in-degree centrality, out-degree centrality, pagerank centrality, eigenvector centrality and closeness centrality to identify potential influencer based on five strategies. All these calculation aim to find the most central nodes in a network but the definition of the most central nodes always differs. While in- or out-degree only focus on the follows or follower count, the remaining calculations take the importance of the nodes into account. In the last step we evaluate features of the potential influencer and their follower crowd based on the results between an influencer and a potential influencer.

As it can be seen, every centrality calculation favours different nodes, which leads to various results for every computation and only a small subset are really influential streamers. First, each result of the centrality calculation is compared with the others by intersecting the outcome. This helps to find similarities between the ranking of the different methods and illustrates which methods favour the same streamers. Next, an intersection of all centrality measures is done to find a subset of streamer who are ranked high in all calculations. The resulting subset of streamer is the so-called influencer subset. Finally, after identifying the influential streamers in the whole graph, the impact of other streamers has to be measured. Therefore, a new network is created containing the influencer subset and all their followers. For every follower of an influencer in the subset, the streamer behaviour before a friendship with the influencer has been forged is analyzed. This observation takes played games, stream time, viewer and follower count into account.

The same observation is performed from the begin of their friendship. The results of both observations are compared and indicates if the streamer changed his behaviour and was affected by the influencer's behaviour. Applying this method to the influencer network with their follower, patterns can be identified of how certain influencers affect their followers. Finally, the ranking of the influencers in the centrality calculations are compared to create relations between the centrality calculation and the impact on streamers in the network.

2.4 Summary

This chapter gave a short overview and brief introduction of fundamental topics relevant for this thesis. In the first section, we showed the importance of analyzing a network in order to understand the network structure, network trends and how key members of the community can be identified. We thus introduced a method to analyze a network, namely SNA. After some theoretical background information about graph theory we used in association with SNA, we focused on SNA in more detail and its application in various fields. First we looked at SNA analysis on popular games like *Tom Clancy's The Division* or *Dark Souls 3*. As a result, player behaviour and influential gamer were identified. Next we inspected some analysis on our target platform Twitch which deals with ranking streamers by popularity, the social behaviour or identifying player communities. After explaining SNA and describing some of its application, we switched our focus to the streamer platform Twitch. We presented some statistics of Twitch which points out the importance and influence of the platform. A comparison with other streaming platforms showed that Twitch is by far the most popular one. One reason for its popularity might be Esport events hosted on Twitch which is watched by a large audience. The following chapter dealt with the behaviour of streamers. Success and acceptance in the community are reasons why people broadcast live streams and also continue their shows. Another important aspect of streaming is the social component and interaction with the performers which attract the viewership. Finally, we close with a section about identifying influencers in network. There is no formula to calculate influencer or key members of a community. Therefore, we proposed some methods in order to find key member in a network, which is an important part of our work.

3 Dataset

In this chapter, the process of gaining all the necessary data for this thesis and its metric is described. First, all the data has been crawled from an application programming interface (API) of Twitch. In the next step, the key metrics about the information of the data and the relational database schema, where all the data is stored is explained. Finally, we take a closer look at the relevant features in the dataset.

3.1 Overview

For this thesis we used our own collected dataset which was our first big task defined in Section 1. All the data we used in this thesis was extracted from an API of Twitch by using a python script. The dataset provides information about users and how they are connected. A user can follow or be followed by other streamers but also create and join teams. Furthermore, we collected data about the gaming and streaming activity of streamers. Every streamer has a channel where all videos and clips, owned by the streamer, are stored. For every stream, game, video, and clip we stored the viewer count in periodic time intervals to score how successful a streamer is. After collecting some user-related data, such as tags, users, channels, clips, videos, and games, we started to focus on streams. In the following Table 3.1 we listed the row count of the Features we collected. Additional to the collected Twitch users, 75 Million user IDs were detected without collecting detailed information about the user. This shows that the network contains at least 135 Million users. The *User* and *Streamer* dataset is needed for the SNA. *Collected Streams* contains various stream data from collected users. Streams are usually tagged which are stored in a separate table linked with the stream data. The *Game Statistic* holds information about game trends in

Twitch Data Set	
Feature	Data Count
User	60 Mio.
Collected Streams	53.3 Mio.
Game Statistics	18 Mio.
Clips	15 Mio.
Videos	3 Mio.
Streamer	850,000
Games	7,653
Teams	4,800
Stream Tags	786

Table 3.1: Key Features of the collected Twitch data.

Twitch. Every 30 minutes all currently played games and their popularity are stored in the database. *Clips* and *Videos* contains information about recorded streams which was mainly used for *affiliate* and *partner* streamer. In addition, for each collected user, information about their *Teams* was stored.

Besides collecting general data from Twitch, we analyze Twitch's social network of streamers and users. For this task we collected more than 60 Million users whereof about one million are part of the *affiliate* or *partner* program. The social network structure is needed to analyze influencers and find the key members in the network.

3.2 Crawl the Dataset

The first challenge in this thesis was obtaining a suitable dataset with enough detailed information. This happened in two separate steps which will be discussed in detail in this section. Furthermore, we explain our strategy of how we achieved this goal. Next, the focus is on the Twitch API we used to get data from Twitch. Finally, the network crawler will be explained in detail.

3.2.1 Overview

Before we go into further detail, we want to give a brief overview of how we crawled the data from Twitch. For this task we used *Python* for various reasons. First, the programming language is perfectly suitable for the requirements of collecting and processing large datasets. Furthermore, libraries already exist for the Twitch API but also for several types of databases. Another advantage of *Python* is the fact that it is environment independent. The task, collecting data from Twitch, required also the need for a database in which the collected data is stored. In our work, two different databases were used. A relational database, namely *PostgreSql*[1], is used in order to store general Twitch data. *PostgreSql* is an open-source database and suitable to storing the relation between the different objects collected from Twitch. For our network analysis we needed a snapshot of the whole Twitch social network. Therefore, we only needed user objects from the API without relation to other objects. Moreover, the performance of simple ID-lookup queries and the dynamic of a database was very important for this task. Taking all this into account, *PostgreSql* was not a suitable database for network analysis and therefore we set up a new database. For the network analysis task we chose *MongoDb*[2] for several reasons. Dealing with large datasets means performance problems by accessing and processing the dataset. Generally, accessing a database is expensive and needs many resources. Therefore, the database was only used for storing the dataset, not performing complex queries on it. The queries, performed on the database, are mainly ID-lookups and insert statements. Jung et al. (2015) compared the performance of *MongoDb* and *PostgreSql* databases within view of big data. Processing large datasets are connected to performance and cost problems. A relational database management system such as *PostgreSql* has its advantages dealing with structured data. In the case of unstructured datasets, a relational management system must convert the data into a relational structure which costs performance. On the other hand, *MongoDb* is an unstructured database and very dynamic in its design. Multiple times, the layout of our table changed or got extended during the development process which was easily done with *MongoDb*. The performance of the database has

[1] https://www.postgresql.org/
[2] https://www.mongodb.com

also satisfied our needs.

The crawler is hosted on a virtual machine of Amazon Web Service (AWS)[3]. It is equipped with 4 GB Memory and 2 virtual CPUs. In the beginning, the virtual server had enough computational power to satisfy our needs. To improve our performance, this machine was not suitable anymore and we switched to a stronger virtual machine provided by *Hetzner*[4] due to pricing reasons. Finally, the crawler is running on two virtual machines at the same time to achieve a good performance.

3.2.2 The Twitch API

In order to collect our datasets we used the Twitch API (Twitch, 2020d). The Twitch API provides several endpoints to retrieve public data of the streamer. Most endpoints are offered by the older API version v5, hence we mainly used this version. Besides, we used *Python*[5] as the programming language for our crawler. As an interface to the Twitch API we used an already existing open-source library on GitHub [6].

Generally, for requesting data from the API, an authorization method is necessary which is implemented with a client ID generated Twitch developer account. This client ID authorizes the user to request which is not personalized. To obtain more information such as a mail addresses, an authentication scope is needed. Personalized information was not of any interest in this work and therefore we used the simple client ID. Besides the limited data provided by the API, the request limit per minute is restricted. Each client ID is allowed to send 800 requests per minute (Twitch, 2020b). However, it quickly became apparent that collecting data from the API with a rate limit of 800 requests per minute is unfeasible. To solve this problem, we implemented a load balancer which distributed the requests of multiple client IDs. This was a key factor to achieving average rate limits between 2,000 and 4,000 requests per minute.

[3]https://aws.amazon.com/ec2/
[4]https://www.hetzner.com/cloud
[5]https://www.python.org/
[6]https://www.github.com/tsifrer/python-twitch-client

The Twitch API returns the requested result as *Json* format as specified by Twitch (2020c). Parsing of the request is done by the open-source library mentioned earlier. This simplified the evaluation of the result. After receiving the requested data, the results are preprocessed by the *Python* script before it is stored in the database. Another problem that occurred during the development was that for a certain request, the cursor was not available to request further data. Therefore some changes were made to the open-source library to receive the complete dataset.

The structure of a request to the Twitch API always remained the same. The client ID is mandatory and is stored in the request header namely *Cliend-ID*. In general, all endpoints offer a response limited to 100, which defines the number of returned entries. Furthermore, a paging mechanism is implemented to receive the complete dataset. Therefore, a cursor or an offset is returned to request the following entries.

```
{
   "total": 12345,
   "data":
   [
      {
         "from_id": "171003792",
         "from_name": "IIIsutha067III",
         "to_id": "23161357",
         "to_name": "LIRIK",
         "followed_at": "2017-08-22T22:55:24Z"
      },
      {
         "from_id": "113627897",
         "from_name": "Birdman616",
         "to_id": "23161357",
         "to_name": "LIRIK",
         "followed_at": "2017-08-22T22:55:04Z"
      },
      ...
   ],
   "pagination":{
     "cursor": "eyJiIjpudWxsLCJhIjoiMTUwMzQ0MTc3NjQyNDQyMjAwMCJ9"
   }
}
```


3.2.3 Crawler

The crawler is the core component to collecting the dataset. As previously mentioned, for accessing the Twitch API we used a *Python* library from *GitHub*. The crawler is designed to collect as much data as possible in a short time. Therefore we introduced multi-threading to run multiple instances at the same time. To achieve the right performance to collect enough data was one of the hardest tasks.

Collecting General Twitch Dataset

The first part of the dataset collected in the *PostgreSql* database focuses on collecting as much data as possible from Twitch. Therefore, data is requested from the API. After parsing the response, the resulting data is cleaned to store it in the database. The cleaning process includes the removal of unimportant fields in the received data such as URLs and links. Furthermore, relations to other tables are resolved before the data is stored. The tasks are separated into different components that are running in parallel. As a server we used an AWS instance to host our crawler and also in the same instance, our database service. In over 2 months we collected around 17 GB data from Twitch. A more detailed view of the data is listed in Table 3.1. The description of features follows in Section 3.3.

Collecting Social Network of Twitch

The second part of our data retrieval task was to recreated the whole social network of Twitch. Therefore some reorganization was necessary. The same crawler as described was used but the performance needed to be optimized. We were unable to find numbers about the size of the Twitch network to estimate the effort and time for collecting all users. To store a social network, we do not need a relational database structure. Therefore, we switched the database from *PostgreSql* to *MongoDb* for this task. A strength of *MongoDb* is the flexibility and dynamic it provides. Furthermore the performance of simple queries is better compared to a relational database schema.

For every user in the network we stored the unchanged request from the Twitch API in a *MongoDb* collection. In addition to mapping the associations between users in the network, we stored the list of follows for every user. Our first attempt to store the follower list failed due to the limitation of the document size by 16 MB (MongoDB, 2020). The crawler run on an AWS instance was limited by 4 GB RAM and 2 CPUs and we did not achieve the desired performance. By requesting another virtual machine with 8 CPUs and 32 GB RAM the performance increased but still far beyond all expectations.

In the next step we migrated the local *MongoDb* to a MongoDb Atlas[7] database in the Cloud. This enabled new possibilities for running crawler instances on two virtual machines and feeding the online database. The next problem we ran into was the limitation of the database. Through the high write and read traffic with several thousands per second, the database collapsed under this load. In order to solve this problem, the design of collecting data in the crawler was changed to minimize the load of the database. An important step in this optimization was to separate the read and write requests on the database. Therefore we implemented in our crawler a new task which collected 1,000 ids in a list and stored it in the *id_list* collection. With this improvement, a crawler thread requested the first unmarked collection, containing the ID list and marked it with a flag. This reduced the load of the database enormously. Furthermore, the lookup request, if a user is already in the database, had to move to an in-memory-lookup. This performance increasing modification needed some effort to be realized. After trying different *Python* libraries dealing with large datasets such as *Pandas*[8] or *Numpy*[9], the lookup in memory was too slow to use it in an efficient way with 60 Million entries. The performance of *Pandas.isin*[10] function caused the bottleneck in the collecting process. By facing these problems, the use of *Cython*[11] appeared. *Cython* combines the advantages of *Python* with the speed of *C*. realead (2020) presented and implemented a solution to this problem related to *Cython* called *khash*. A comparison between *khash* and other libraries can be found in Figure 3.1. The *khash* implementation performs the lookup, if an array is contained in another in constant time, $\mathcal{O}(1)$, compared to *Pandas* which needs linear run-time. This modification increased the speed of the whole crawling process multiple times. The final version of the crawler runs on two virtual machines with more than 30 threads at the same time. Every minute around 5,000 Twitch API requests are processed. This results in a throughput of around 280,000 user IDs for a single thread per day.

Besides the process of storing user data into the *follower* collection, several

[7]https://www.mongodb.com/cloud/atlas
[8]https://pandas.pydata.org/
[9]https://numpy.org/
[10]https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.isin.html
[11]https://cython.org/

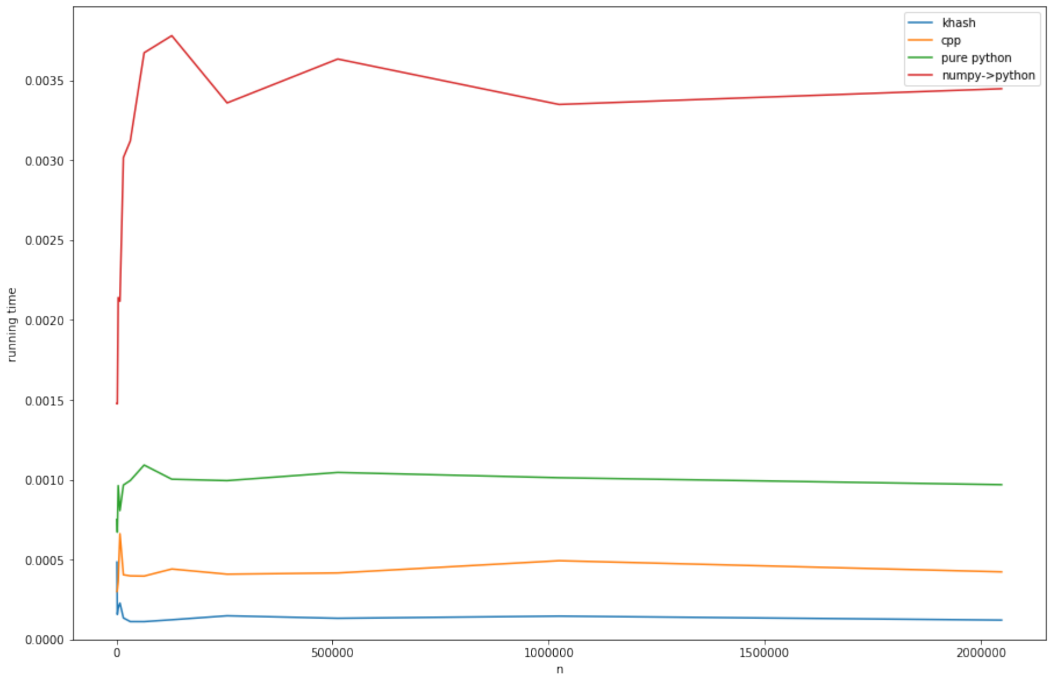


Figure 3.1: This illustration shows the run-time performance of *khash*, *C++*, *Python* and *Python-Numpy*. Illustration adapted from ead (2018).

threads were searching for new user IDs on Twitch to support the crawler with new user IDs. For handling a running instance of the crawler we introduced a new collection called *appconfig*. The collection contains several configurations to start or stop the crawler or to regulate the active thread count.

3.3 Feature Description

The Twitch API provides various data of all streamers. By collecting as much data as possible, our focus was on the most popular streamers on Twitch who can easily be found at TwitchTracker (2020e). The following listing describes the collected features in more detail. Moreover, Figure 3.2 shows a screenshot from the Twitch streamer *itsjavachip*[12] playing *Assassins Creed: Odyssey*[13]. The green rectangles with numbers are referenced in the listing below to see where the features are visible in on Twitch. Figure 3.3 shows the overview of recorded videos and clips of the user. Next, we describe the single pages and reference the parts of the screenshots with numbers to the description.

User

1 **Name**: The Streamers Login Name. It can be changed after 30 days.
- **Display Name**: The display name is shown to other streamers on Twitch.
- **Type**: Defines the status of a user. Possible values are *partner*, *affiliate*, *oCPM* or empty value for ordinary user.

2 **Bio**: The Streamer can fill in a biography that is visible to everyone.
- **Follows**: *Follower* contains all users, who the current user is following. In addition, we stored the creation date of every friendship.
- **Follows Count**: The number of users, who the current user is following.

[12]https://www.twitch.tv/itsjavachip/
[13] Ubisoft Quebec , 2018. https://assassinscreed.ubisoft.com/game/en-us/home.

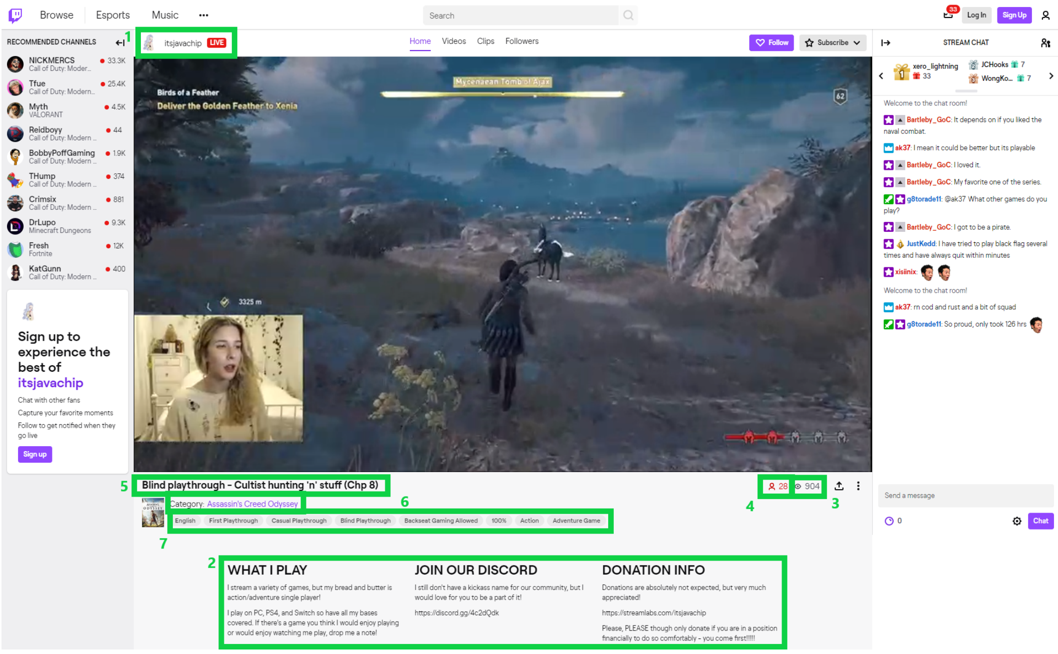


Figure 3.2: The screenshot shows the main screen showing several features such as name, game, viewer.

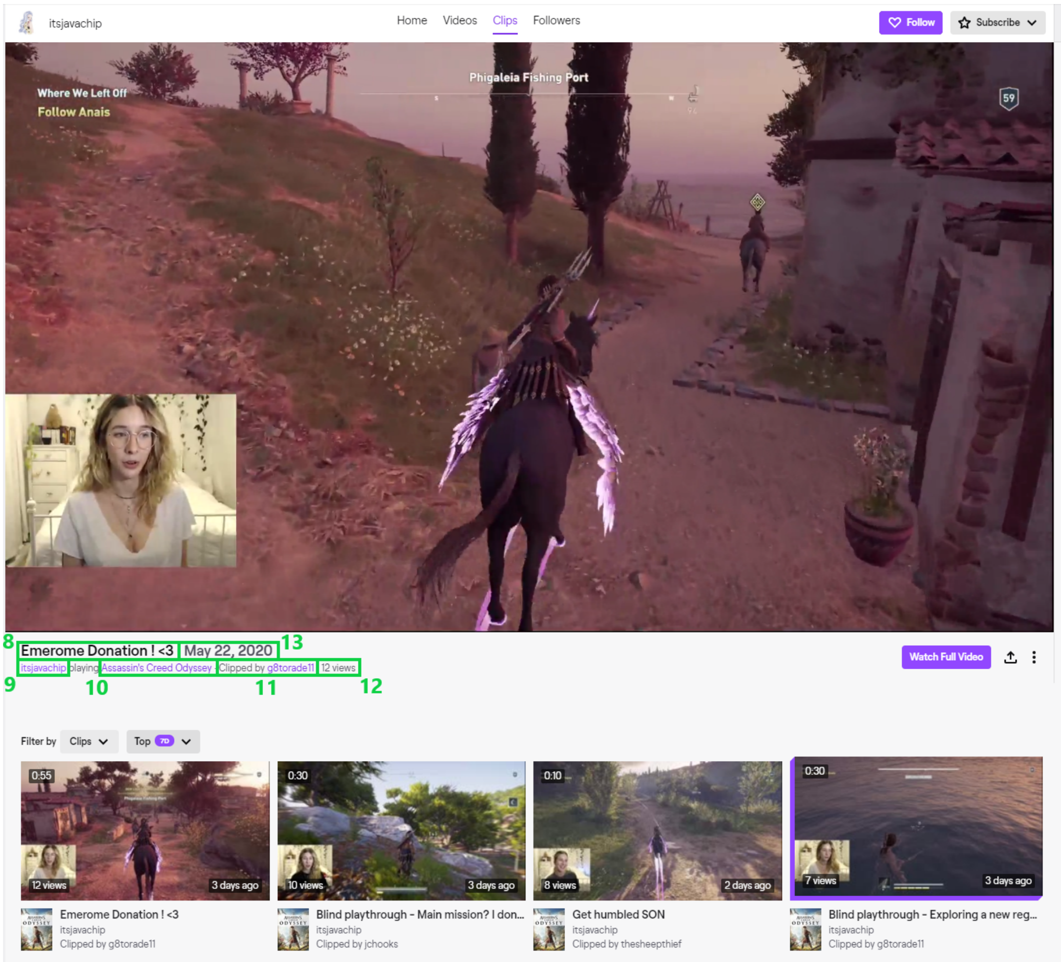


Figure 3.3: The screenshot shows the main screen showing several features such as name, game, viewer.

- **Follower Count**: The number of users, who are following the current user.
- **Language**: The language the streamer is broadcasting.

3 **Views**: The total number of viewers who have visited the channel.

- **Created at**: The date when the streamer account was created.
- **Updated at**: The date when the streamer account was updated the last time.

Channel

Every user owns a channel where videos and clips are stored and the streams can be watched. The channel contains following information:

- **Mature**: Is set true if the channel contains mature content.

1 **Display Name**: The display name is shown to other streamers on Twitch.

- **Language**: The language the streamer is broadcasting.

2 **Description**: Streamer can specify a description of the channel.

3 **Views**: The total number of viewers who has visited the channel.

5 **Status**: Streamer can specify a status for the channel.

6 **Game**: Last played game.

- **Broadcaster Language**: Defines the language of the streamers broadcasts.
- **Broadcaster Software**: Defines the software of the streamers broadcasts if the information is available.
- **Created at**: The date when the channel was created.
- **Updated at**: The date when the channel was updated the last time.
- **Partner**: Is set true if the streamer is a partner of Twitch.tv
- **Followers**: The number of streamer who are following this channel.
- **Broadcast Type**: Possible values for "*Broadcast Type*" are "archive", "highlight" or "upload". A "highlight" is a cut from past broadcasts. Streamer can also upload prerecorded videos which will be marked as "upload".
- **Private Video**: Is set true
- **Privacy Options Enabled**:

Game

Additionally, we stored popularity, viewers, and channel count for each game. This happens in a periodic interval of 30 minutes to get an accurate timeline of how the game's popularity is changing.

- **Popularity**: The popularity of the game.

4 **Viewers**: The number of current viewers who are watching this game.

6 **Name**: Name of the game in English.

- **Channels**: The number of channels that are streaming this game at the moment.

Clip

Furthermore, a channel contains videos and clips of streams. For videos and clips we collected following information:

8 **Name**: Name of clip.

9 **Broadcaster**: Broadcaster of clip.

10 **Game**: Game which is played in clip.

11 **Curator**: Curator of clip.

- **Video on Demand**: Referencing the video where the clip is from.

8 **Title**: Title of clip.

- **Language**: Language of the stream from which the clip was created.

13 **Created At**: Date when clip was created.

12 **Views**: Number of times clip has been viewed.

- **Duration**: Duration of clip.

Video

For *affiliate* or *partner* streamer, the video section contains all broadcasts from the previous two month. The information for the videos are similar to the clip response.

- **Title**: Title of the video.
- **Description**: Description of video.

- **Broadcast Type**: Broadcast type of video.
7 **Tags**: A list of tags the video has been tagged.
8 **Status**: Status of video.
10 **Game**: Game which is played in the video.
12 **Views**: Number of times the video has been viewed.
- **Language**: Language of the video.
- **View-able**: Indicates whether the video is publicly view-able.
- **Published At**: Date when the video was published.
13 **Recorded At**: Date when the video was recorded.
- **Duration**: Duration of the video.
- **Channel**: User who owns and uploaded the video.

Stream

The request for streams returns streams specific details such as current viewer count, game, and tags. A more detailed overview is shown below.

- **Broadcast Platform**: Stream title.
- **Video quality**: Properties of video.
- **Created At**: Start date of stream.
- **Is Playlist**: Stream is from playlist.
- **Stream Type**: *Stream Type* can be *live* or empty.
4 **Viewer**: Number of current viewers who are watching the stream at the time of the query.
5 **Title**: Title of stream.
6 **Game**: Game which is streamed.
7 **Stream Tags**: Tags of stream.
9 **Channel**: Corresponding channel.

3.4 Summary

In this chapter we described in detail, how we collected our dataset. First, we started with an overview of the whole process. The chosen programming language is *Python* due to its suitability to our requirements and its many libraries on offer. Due to different datasets and their varying structures, two separate database technologies are used, namely *PostgreSql* and *MongoDb*. The structural database management system *PostgreSql* was needed to store the structured dataset of general Twitch data. For the network analysis task, the dataset has no relation to other objects anymore and therefor we switched the database technology to *MongoDb*. The crawler is hosted on a virtual server provided by *Amazon*. Due to performance and pricing issues, the cloud service provider was changed to *Hetzner*. There we use two instances to achieve good performance for collecting and analyzing the datasets. Second, the Twitch API is described in detail and how it was used to gain the datasets. Next, the logic of the crawler and its design is explained in more detail. The crawler is designed to work as efficiently as possible and works in parallel on multiple instances. The first dataset, containing general data of Twitch, ran over 2 months and collected data in the size of around 17 GB. For the social network dataset, some essential structural changes were made to fit the performance requirements. During the development process, several problems and limitations occurred which we had to overcome. A lot of effort was put into the performance optimization, which was quite challenging, to meet the requirements to crawl a large network with more than 70 million users. This section closed with the feature description of the collected datasets. This data collection is a main part of our thesis in order to provide a dataset for further researches.

4 Data Preparation and Processing

As described in the previous chapter, the datasets were collected from the Twitch API, preprocessed, and stored in a database. In this chapter we will explain the preparing, processing, and visualization of the datasets.

4.1 Overview

In the previous chapter we described how the datasets were collected from the Twitch API. The resulting dataset for our network analysis contained over 70 Million registered Twitch users. The network crawler is still running and searching for more users. To work with a reasonable amount of data we defined some restrictions to retrieve a smaller sub-graph to work with. As already mentioned, in our work we define registered users as a *streamer* if they have reached at least the *affiliate* state. This restriction leads to a dataset size around 850,000 streamers connected by 78.5 Million edges. The size of the streamer network is still huge. It took some effort to find a suitable graph library to satisfy our needs. Therefore we looked at different libraries to perform our calculations on the network. The graph was created by using a graph markup language (GML) file. After the whole graph was loaded into the network analyzing tools, some centrality calculations were performed to define the most central streamer in our network. In the last step, we used the graph visualization tool *Gephi*[1] to print the graph structure and its features. Figure 4.1 gives the overall picture of the work we have done. The whole process of this analysis is visualized in Figure 4.1 starting with

[1]https://gephi.org/

crawling the data from Twitch, cleaning and storing them into a database, to the conclusion of various analyzes performed on the dataset.

4.2 Processing the Data

The processing step required time and a lot of computational power. We broke the data processing into three parts, namely *Transformation*, *Calculation* and *Visualization*. In the *Transformation* phase, the data was taken from the database, transformed into the right data format, and finally loaded into the graph library. Next, in the *Calculation* phase all calculations including centrality measures are performed. In the final step *Visualization*, the results were evaluated and graphically presented.

4.2.1 Phase 1: Transformation

At the beginning of this phase, a lot of effort had to be put into research to find the appropriate graph library to process a massive network. Due to the high vertex and edge count, not many graph libraries remained to meet our expectations. The best graph libraries for *Python*, which satisfy our requirements, are *igraph*[2] and *SNAP*[3]. *igraph* is an open-source library and provides several tools for network analysis. It is designed to be efficient and to handle massive networks and it is easy to use. The implementation is available for several programming languages including *Python* (igraph core team, 2020). On the other hand, *SNAP* was developed by *Stanford University*[4] and is an analysis tool for large graphs. In particular, the performance of this tool and compact graph representation makes the tool attractive for large network sizes (Leskovec, 2019).

[2]https://igraph.org/python/
[3]https://snap.stanford.edu/snappy/
[4]https://www.stanford.edu/

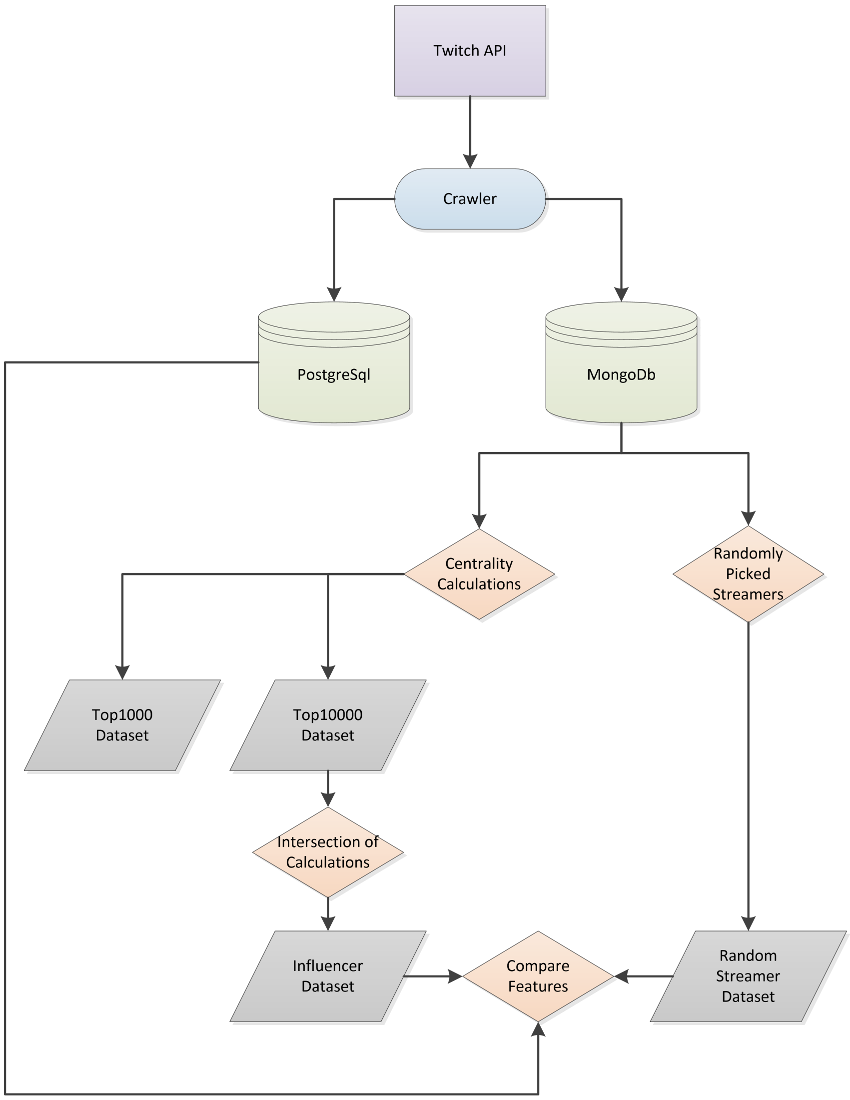


Figure 4.1: This diagram shows an overview of our work starting with collecting and storing the data from Twitch to analyzing the results.

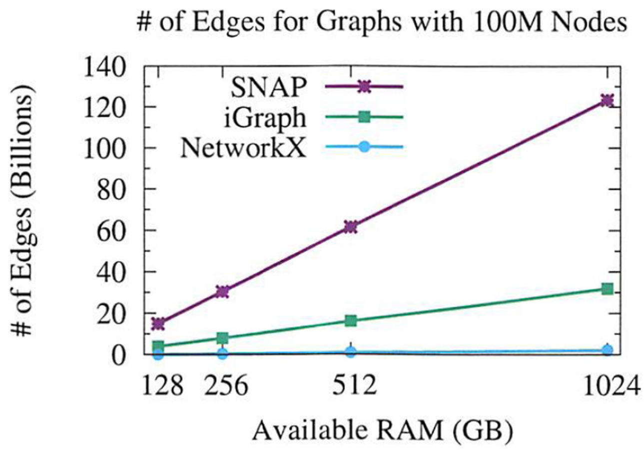


Figure 4.2: This diagram shows the memory consumption of three different graph libraries at which *SNAP* performs the best. Diagram adapted from Leskovec and Sosič (2016).

igraph vs. SNAP

Leskovec and Sosič (2016) compared in their work different graph libraries including *igraph* and *SNAP*. According to this work *igraph* consumes four times more memory than *SNAP*, illustrated in Figure 4.2. Therefore, *SNAP* is the most memory efficient package presented in the paper. Comparing graph algorithms, *igraph* tends to be a little more efficient than *SNAP* but both results are good. The performance of the graph library *networkx*[5] was far beyond *SNAP* and *igraph* with respect to algorithm calculation time and memory consumption.

After studying these two libraries in detail we decided to use both of them to benefit from the advantages of each library. Concerning *igraph* we want to point out the advantage of the easy use of the library and working with attributes. Furthermore, some algorithms for directed graphs are implemented such as Pagerank or Eigenvector centrality, which are not implemented in *SNAP*. An advantage of *SNAP* which we used was that the calculation of closeness centrality is implemented iteratively. After a first attempt we realized that to calculate the closeness centrality for such a huge

[5] https://networkx.github.io/

network would take approximately 60 days with *igraph*. With the iterative implementation of this algorithm in *SNAP* it was possible to distribute the calculation and speed it up so that it only took 6 days.

Loading the Graph into the Library

The process of graph creation was associated with a lot of issues. Before we started working with the dataset, some modifications had to be made on the database to increase the performance. As already mentioned before, for the social network analysis only streamers in the network are relevant for our research. Therefore, to reduce the query time in the database, a new table in the database was created which contains only the streamers. Instead of querying 70 million documents, only the small subset of streamers with around 850,000 document entries was used which significantly improved the performance.

In the next step we loaded the graph into the *Python* graph library. By taking into account the illustration of Figure 4.2 and our network size of 850,000 users and over 78.5 million connections, a lot of memory is needed. Furthermore, we have to consider additional memory for attributes for every vertex and edge. Our first attempt, creating the graph from scratch with *igraph* failed due to the long creation time. According to Leskovec and Sosič (2016), the creation time of a graph with *igraph* is twice as fast than *SNAP*. Therefore, a graph markup language file is used, called *GML*, to define the whole network. The file format is shown in Figure 4.2. The *igraph* library was able to load the graph file within 10 minutes but needed almost 60 GB of memory to perform the import process. Fortunately, we had the necessary resources for this operation. After the graph was loaded from the GML file, the memory consummation was reduced to approximately 6 GB. We used the function *write_pickle* of the *igraph* library to store the graph object to the disc. Loading the dumped graph from the disc requires temporally about 15 GB memory but after loading only 6 GB are needed. Due to the available resources we did not try any test of loading the graph with all attributes into the *SNAP* library. As already mentioned, the computation of the closeness centrality was done within the *SNAP* tool. In order to do so, the graph without attributes was exported from the *igraph* library in a

Pajek[6] file format and imported by *SNAP*. The graph creation process took much longer than the *igraph* library but only a little memory was necessary. After successfully building the graph once, the graph object was dumped to the disk and loaded from there which saved memory and time.

Creating the GML File

Concerning our large dataset, we tried to reduce the file size of the GML file a bit. Additionally, we avoided the text field in the GML file and encoded the streamer type and game ID as numerical values. *Affiliate* streamers are represented as 1 and streamers who joined the *partner* program as 2. The language field we kept, as it was due to their small text size. For the game field, we realized that the game IDs are differently encoded. Some game IDs were stored as numbers and some as a game names due to the use of different Twitch API versions. The older version (V5) returns all data we needed in a single request and the game ID is encoded as game name such as *Fortnite*. During the development and crawling process, different approaches were tried out, and therefore also the new Twitch API endpoints were used as well. The advantage of this request was that 100 IDs could be requested at once, but therefore not all necessary data was returned. To obtain all the needed data, many request were required which was slower in the end and we did not use it anymore. Therefore, we decided to create a game lookup table to be able to resolve this encoding without making a Twitch API request. In the next step, all game IDs were brought into the same numerical format. The edges contain the additional attribute *created* which stores the creation time of the friendship. The date is simply encoded as an eight-digit number representing the year, month, and day.

4.2.2 Phase 2: Perform Calculations

In the previous step the graph was loaded into the *Python* graph library by using a GML file. Next, the calculation of different centrality measures was performed. The centrality values indicates the most central vertices based

[6] http://vlado.fmf.uni-lj.si/pub/networks/pajek/doc/pajekman.pdf

on different definitions of *central*. We used this approach and calculated five centrality measures to obtain varying results. Finally, we looked for a small subset contained in every centrality calculation. As already mentioned, the calculation of the closeness centrality was performed with the *SNAP* library and all other calculations with *igraph*.

In Degree Centrality

This centrality calculation is based on the incoming degree of a node and defines a node as more central if the in-degree is higher. Applied to our network it means the user with the highest follower count is evaluated as the most central node. The in-degree centrality C_{Din} of a user is defined as

$$C_{Din} = \text{indeg}(v) \tag{4.1}$$

Out Degree Centrality

Similar to the in degree, the calculation is based on the connection of a vertex but for the out-degree centrality only the outgoing connections are taken into account. For the Twitch network it means that the user who is following the most users is evaluated as the most central node. The out-degree centrality C_{Dout} of a user is defined as

$$C_{Dout} = \text{outdeg}(v) \tag{4.2}$$

Closeness Centrality

This algorithm, assign nodes a high centrality values that are closer to all other nodes (Cohen et al., 2014). In other words, a node with a low distance to all other nodes is more central (Bavelas, 1950). The closeness centrality is expressed as

$$C_c(x) = \frac{N-1}{\sum_{y \in V} d(y,x)} \tag{4.3}$$

where $d(x,y)$ is the shortest path between the nodes x and y and N the total count of vertices in the graph. As already mentioned, this calculation was performed by the *SNAP* tool due to the iterative implementation of the algorithm. The virtual machines together provided ten threads. Therefore the total number of users was split into ten almost equal fragments and were processed in parallel. This reduced the computation time from approximately 60 days to only six days. The result is stored on the disk and it can be loaded back into the *igraph* library.

Betweenness Centrality:

The betweenness centrality is like closeness centrality based on shortest path. The vertices with the most shortest path from a user to another one passing through the vertex get a higher score. The betweenness centrality can be expressed as

$$C_b(x) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{4.4}$$

σ_{st} describes the shortest path between the vertex s and t. Similar, $\sigma_{st}(v)$ describes the shortest path between the nodes s and t but the path must pass the vertex v. This ratio is called betweenness centrality. The calculation of the betweenness for our network was stopped after four weeks of calculation time. Despite the improvements made on the algorithm, the high complexity and the high number of links between the nodes made it impossible to achieve results for this centrality measure (Brandes, 2001).

Eigenvector Centrality

As concept for this method, connections to higher ranked nodes receive a higher score than lower-ranked nodes. Therefore, a high score is reached by being connected to *important* nodes. The calculation took place in three steps:

1. Creating the adjacency matrix A. If the vertex v is connected to vertex w the entry in the matrix $a_{v,w} = 1$ otherwise 0.
2. Resolve the Eigenvector equation $Ax = \lambda x$
3. The eigenvector centrality of the vertex v_i at index i is contained in the eigenvector x on the i-th position.

Pagerank

This algorithm used by the search engine to rank search results can also be used to rank nodes in a network. As an initial state every node in the network is assigned the same ranking. By iterating a defined number of times through the algorithm, the ranking in the network is shifted. The importance of a node depends on its in-degree, the number of incoming links. All outgoing links are weighted with the score of the node. Therefore, nodes with a high in-degree are ranked with a higher score and impact all nodes on an out-going connection with its value.

4.2.3 Phase 3: Visualize Features

In the last step, after calculating the main centrality measures, we visualized our results. The resulting sub-graph with 37,500 streamers and 75,000 connections is relatively small compared to the original one. Therefore we were able to use visualization software to bring the graph to the screen. As simulation software we chose *Gephi* to visualize the sub-graph. The *Python* libraries *SNAP* and *igraph* contains both a visualization feature to plot a graph structure. *SNAP* points out in the manual of the plotting function that it is only suitable for graphs with less than 100 nodes. Almost the same limitations are valid for the *igraph* tool and it was not practicable to use for our graph. *Gephi* is an open-source and platform-independent software tool to visualize graphs. Moreover it offers many already implemented algorithms and features relevant for network analysis. Bastian et al. (2009) introduced in their work the *Gephi* software and highlighted the features.

Gephi Input Format

Gephi offers many different input formats based on XML, tabular, or text. Depending on the use case, a suitable data format can be chosen. In Figure 4.3 the supported graph formats of Gephi are shown. The choice of a file format is based on the features which are needed for the visualization.

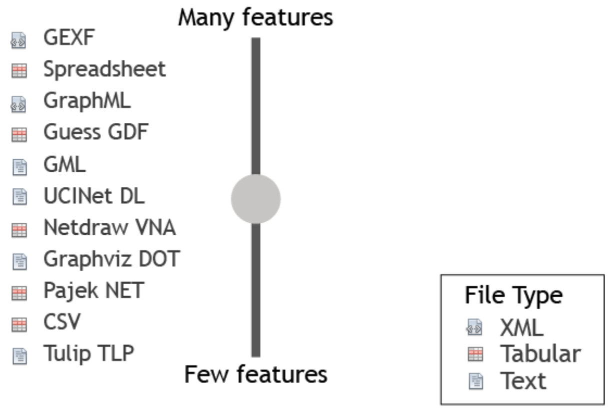


Figure 4.3: Gephi File Formats: This figure shows a list of supported file formats. The different formats are ordered by complexity and supported features which is illustrated on the right side. Illustration adapted from Gephi (2017).

The Tulip TLP or CSV are the simplest input file formats which is basically only a list of nodes and edges. No additional features are supported or provided. For our social network of streamers we wanted to include more information in our graph such as played games, language, viewer count, follower count and so on. In order to use additional attributes we must take a closer look at a more complex model. Based on the listing in Figure 4.3 we tried the *GEXF*, shown in Figure 4.1, input format which fully satisfied our needs. A disadvantage of this file format is the XML structure which produces a large overhead and therefore increases the file size. However, we finally decided to use graph modeling language (GML) format because of its simplicity and efficient way to encode our data. Moreover, the *igraph* library supports to write the graph to the disc in GML format.

Listing 4.1: First verbatim

```
<?xml version="1.0" encoding="UTF-8"?>
<gexf xmlns="http://www.gexf.net/1.2draft" version="1.2">
    <meta lastmodifieddate="2009-03-20">
        <creator>Gexf.net</creator>
        <description>A hello world! file</description>
    </meta>
    <graph mode="static" defaultedgetype="directed">
        <nodes>
            <node id="0" label="Hello" />
            <node id="1" label="Word" />
        </nodes>
        <edges>
            <edge id="0" source="0" target="1" />
        </edges>
    </graph>
</gexf>
```

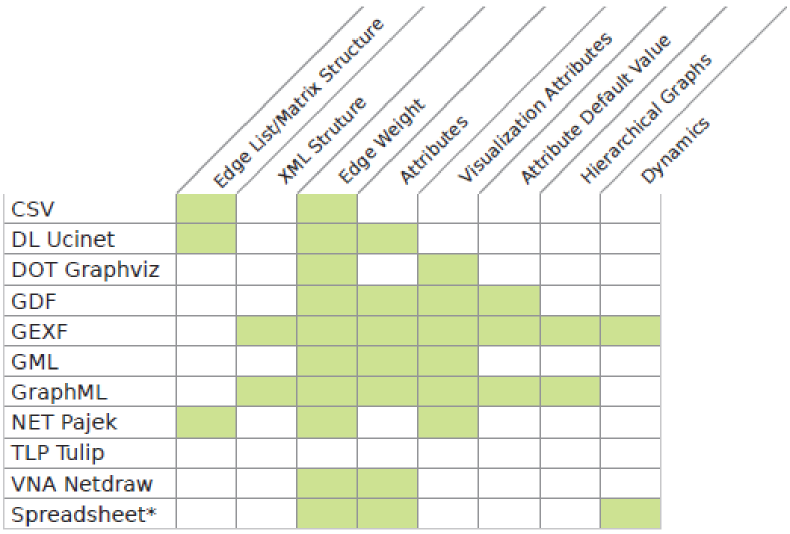


Figure 4.4: A list of different Gephi input formats compared by supported features. In our case the column attributes is an important feature we need for our graph analysis. Illustration adapted from Gephi (2017).

Listing 4.2: First verbatim

```
graph
[
  directed 1
  node
  [
    id 0
    ln en
    type 1
    follower 86
    game 32399
    views 2801
    follows 19
    name 7
  ]
  node
  [
    id 1
    ...
  ]
  edge
  [
    source 0
    target 1
    weight 10
  ]
  edge
  [
    source 1
    target 0
    weight -2
  ]
  ...
]
```

Visualizing the Graph

After creating the input file for the visualization software, containing all the desired attributes, the file was loaded into *Gephi*. Due to the high amount of vertices and edges, the program needed up to 10 GB memory to work on the graph. *Gephi* offers various methods to visualize the social network. Therefore we used different coloring for clusters in the network. To visualize the game cluster, nodes containing the identical game attribute are colored with the same colors. Moreover, *Gephi* offers a various number of different layout algorithms to replace the nodes in the networks based on properties. We used different algorithms to retrieve the desired layout.

Other Statistics

Besides visualizing the main features of the graph, calculation of the streamer attributes was performed. Therefore, different sub-networks were created to find patterns. For each sub-networks some key statistics were calculated such as average viewer, follows, and followers but also rankings were created of most played games, used languages, and type of streamers. These statistics were compared for all sub-networks to find relations between them. This sub-networks consisted of the most central streamer, calculated from the centrality measures. On the smaller sub-network we could perform more calculations such as diameter, density, or largest connected components, due to the size of the network.

4.3 Summary

It took some effort to find a suitable graph library to satisfy our needs. Therefore we looked at different tools and decided to use *igraph* and *SNAP*. Next, the graph was loaded into the *Python* library by using a GML file. This special graph definition file offers the possibility to load the graph into the library instead of creating a new graph from scratch. After the graph was loaded into the library, we were able to perform the centrality measures calculations. The advantages of each library were used to achieve a good performance on the calculations. All results were stored on the disk to reload it quickly for further use. Finally, the results of the calculations were evaluated and visualized with *Gephi*. Therefore, we used the same GML file format to export the network from the *Python* library and load it again into the *Gephi* software. The software visualized our sub-graph of influencers, highlighting features such as played games, spoken languages, streamer types, or different counts of viewer and followers. Furthermore, various sub-graphs were created based on the calculated centrality measures and some statistics were applied to these influencer networks to discover some patterns in it. The results are presented in the following chapter.

5 Analysis and Results

In this chapter, we want to examine the results we acquired through our work. First, an overview of the results of our analysis is given. Next, the outcome of applying SNA on the streamer network and the resulting influencers is presented. Finally, the impact of influencers on their follower is introduced.

5.1 Overview

Our analysis is based on social network analysis. By applying SNA to our network, several centrality calculations were made, introduced in Chapter 4. In this chapter, we compare the results of the centrality measures and set them in contrast with each other. Additionally, for every centrality calculation, node attributes and their distribution are analyzed in more detail. Moreover, some statistics of viewer numbers and follower count are created to determine the outcome of the different centrality methods. Every centrality calculation favors other streamers based on the interpretation of *centrality*. Therefore, the ranked dataset of a centrality measure is intersected with all others to find overlaps in the ranking. Based on the small subset, resulting from the overlap of all five centrality measures, some further analysis is performed. The small subset are the identified influencers in the network. Two new graphs are created based on the influencer dataset to find their impact on their followers. Moreover, the same evaluation of node attributes such as played games, spoken languages, and streamer types are also performed on this influencer networks and compared. Furthermore, *Gephi* is used to visualize the results in the influencer networks, highlighting the key members in the network. In Table 5.1 we listed the different networks we created for our analyses.

dataset	Streamer Network	Influencer Follower	Top10000	Random Streamer
Nodes	856,056	75,670	32,934	15,716
Nodes in LCC	827,820	73,115	31,936	11,050
Edges	78,442,688	6,437,987	5,041,638	284,807
Edges in LCC	76,651,024	6,366,590	4,859,420	233,704
Average degree	183.27	170.16	306.17	36.24
Diameter	3	9	11	17

Table 5.1: List some characteristic values of the networks we created for our analysis.

5.2 Results of Social Network Analysis

The following section presents the outcome of the social network analysis. Therefore, we calculated five different centrality measures, namely closeness, in-degree, out-degree, eigenvector, and Pagerank centrality. For each centrality calculation, a ranking was created to find the most central streamer in each method. The first analysis is based on streamer attributes.

5.2.1 Analysing Streamer Attributes

For this analysis, we took the top-ranked streamer from our previous calculations into account. Our first dataset contains the first 1,000, referenced as *Top1000* and the second dataset contains the first 10,000 streamer, referenced as *Top10000*, for each centrality calculation.

Game Analysis

In this first analysis, we identified the gaming behavior of the influencers, based on the centrality calculations. The results are shown in Table 5.2. For every calculation, the most played game is *Fortnite* and the social category *Just Chatting*, where the performer is just talking with or to the audience. Comparing the *Top1000* and *Top10000* dataset, the most played game did not change and there are only tiny changes in the ranking. The games *Call of Duty: Modern Warfare*, *Apex Legends*[1] and *League of Legends* appeared

[1] Respawn Entertainment , 2019. https://www.ea.com/games/apex-legends.

multiple times in almost all calculations. These games belong to the most played games on Twitch which is reflected in the ranking. *Grand Theft Auto*[2] appears only once in the eigenvector calculation of the *Top1000* dataset but is not listed in the *Top10000* ranking. This indicates that *Grand Theft Auto* is very popular among a smaller group of streamers. By comparing the methods closeness and eigenvector centrality we can see there are almost no games in common. Whereas by closeness calculation shooters like *Fortnite*, *Apex Legends* or *Call of Duty* are more popular, eigenvector is ranking social channels higher. The strategy game *League of Legends* is ranked the highest at in-degree and Pagerank centrality. Moreover, we can observe, that the most played games listed in Table 5.2 are shooter games. *Just Chatting* is present in every centrality calculation which highlights the social aspect of streaming. Beside *Just Chatting*, *Art* is the other social channel listed in the eigenvector results which indicates that these methods rank social channels higher. The only strategy game in this result is *League of Legends*.

Language Analysis

Another property we inspected on our streamer data is the language spoken during the streams. Clearly *English* encoded as *en* is the most spoken language on Twitch listed in Table 5.3. Beside *English*, *German* encoded as *de* and Spanish encoded as *es*, appeared often in the ranking. Therefore, English is the most spoken language on Twitch but the importance of other languages must not be ignored. The results also reflect the statistics of Twitch activities focusing on languages (TwitchTracker, 2020b). Comparing the different centrality calculations closeness, eigenvector and out-degree identify English as the most important languages with values around 90%. Pagerank and in-degree on the other hand contain only 75% English streamers which means that also other languages fill up the remaining 25%.

Streamer Type Analysis

The last attribute we studied is the streamer type distribution of the centrality calculations shown in Table 5.4. The Table shows the distribution of

[2] Rockstar North , 2013. https://www.rockstargames.com/V.

Method	*Top1000* dataset		*Top10000* dataset	
	Game	[%]	Game	[%]
closeness	Fortnite	12.6%	Fortnite	10.31%
	Call of Duty: Modern Warfare	5.8%	Apex Legends	5.8%
	Apex Legends	5.5%	Call of Duty: Modern Warfare	4.8%
	Just Chatting	4.8%	Just Chatting	4.1%
pagerank	Just Chatting	15.1%	Just Chatting	12%
	League of Legends	8.1%	League of Legends	6.6%
	Counter-Strike: Global Offensive	6.7%	Fortnite	5.5%
	Fortnite	6.7%	Counter-Strike: Global Offensive	4.1%
eigenvector	Just Chatting	14.7%	Just Chatting	9.5%
	Grand Theft Auto V	4.7%	Art	4.3%
	Art	4.6%	Fortnite	4.2%
	Fortnite	4.2%	Apex Legends	3.1%
out degree	Fortnite	9.8%	Fortnite	10.3%
	Just Chatting	7.2%	Just Chatting	5.7%
	Apex Legends	4.7%	Apex Legends	4.7%
	Call of Duty: Modern Warfare	4.6%	League of Legends	3.7%
in degree	Just Chatting	13.3%	Just Chatting	11.9%
	Fortnite	9.7%	Fortnite	8.1%
	League of Legends	8.0%	League of Legends	6.2%
	Counter-Strike: Global Offensive	6.9%	Counter-Strike: Global Offensive	4.5%

Table 5.2: In this table the most played games for all five centrality calculation are listed.

Method	*Top1000* dataset		*Top10000* dataset	
	Language	[%]	Language	[%]
closeness	en	92.8%	en	92.01%
	de	0.15%	de	1.54%
	fr	0.10%	es	1.11%
pagerank	en	85.7%	en	75.85%
	ko	3.0%	ko	4.36%
	de	2.7%	de	3.59%
eigenvector	en	99.4%	en	98.15%
	ko	0.3%	de	0.38%
	fi	0.1%	fr	0.23%
out degree	en	90.4%	en	86.19%
	de	1.7%	es	2.89%
	fr	1.6%	pt	2.72%
in degree	en	86.6%	en	74.92%
	de	3.2%	es	4.66%
	es	2.8%	pt	4.15%

Table 5.3: The most used languages are listed for every centrality calculation.

Method	Type	Top1000 dataset [%]	Top1000 dataset [%]
closeness	affiliate	96.8%	96.10%
	partner	3.2%	3.90%
pagerank	affiliate	1.5%	11.92%
	partner	98.5%	88.08%
eigenvector	affiliate	3.8%	45.13%
	partner	96.2%	54.87%
out degree	affiliate	93.7%	91.41%
	partner	6.3%	8.59%
in degree	affiliate	2.0%	15.68%
	partner	98.0%	84.32%

Table 5.4: This Table shows the distribution of streamer types for each centrality measure.

affiliate and *partner* streamer in our influencer network. An interesting occurrence is that *affiliate* streamers dominate only in the *closeness* and *out-degree* calculations. All other results indicate influencers who joined the *partner* program. Another interesting incident is that either *partner* or *affiliate* clearly dominate the results except in the calculation of the *Top10000* result of *eigenvector* centrality. Therefore, the proportion of *affiliate* and *partner* streamers is almost equally distributed, depending on the method. As showed later on, a small subset of *partner* streamers are ranked high in all calculations which may cause the high percentage of *partner* streamers.

5.2.2 Characteristic Numbers

After analyzing the distribution of the attributes on every calculated centrality measure some characteristic numbers are calculated to get an overview on follower and viewer counts. Furthermore, the subset of streamer based on all centrality calculations was evaluated.

Method	Follower			Follows			Views		
	max	min	avg	max	min	avg	max	min	avg
closeness	445,904	46	2,125.75	2,002	0	295.91	23,354,212	146	81,277.98
pagerank	7,548,258	301	420,071.05	1,931	0	210.18	434,955,643	3,363	26,565,101.12
eigenvector	7,548,258	2,398	284,939.41	2,250	0	436.72	434,955,643	33,380	17,169,694.88
out degree	1,192,890	56	5,260.43	3,684	1,211	1,765.30	29,459,374	279	191,098.59
in degree	7,548,258	8,275	444,817.64	1,998	0	223.90	434,955,643	15,909	24,944,548.46

Table 5.5: The Table lists a detailed statistic of *Top1000* streamer calculations.

Method	Follower			Follows			Views		
	max	min	avg	max	min	avg	max	min	avg
closeness	1,659,632	2	2,604.73	3,222	0	255.76	116,566,889	1	87,019.29
pagerank	7,548,258	11	93,161.12	2,250	0	233.98	719,019,558	102	5,278,043.11
eigenvector	7,548,258	0	59,131.01	2,886	0	387.70	719,019,558	302	3,255,509.38
out degree	1,192,890	36	4,699.04	9,703	525	990.40	53,892,103	95	167,779.17
in degree	7,548,258	1,183	96,295.28	2,886	0	279.66	719,019,558	2,287	5,204,693.66

Table 5.6: In this Table the *Top10000* streamer calculations are listed with characteristically numbers of follower, follows and viewer counts.

Calculating Streamer Statistics

Based on every streamers' viewer, follows and follower count in a calculated set, some statistical values are evaluated. Therefore we took the minimum, maximum and average of all these values and compared them in Table 5.5 and Table 5.6. At both datasets, the ratio of an average follower is the same with the highest average by *in-degree* and the lowest by *closeness* calculation. Therefore, the *out-degree* centrality produced the highest average follows count. The average view count is dominated by *pagerank*, *in-degree* and *eigenvector* calculations.

As we can see, all centrality measures highlight different aspects of streamer properties. Moreover, each calculation ranks the streamers differently which results in the various outcomes. Based on this observation, we continued working with this set by intersecting the outcomes and list the overlap. Each centrality calculation result is compared with the outcome of the other methods in order to find similarities between these methods. In Table 5.7 the results of the *Top1000* dataset are compared. The *eigenvector* centrality to *in-degree* and *pagerank* calculation has an overlap around 50% and *pagerank*

	closeness	out degree	in degree	eigenvector	pagerank
closeness		2%	0.2%	0.2%	0.2%
out degree			0.8%	3.7%	0.5%
in degree				48.6%	77.4%
eigenvector					51.7%
pagerank					

Table 5.7: This table shows the overlapping of each centrality measure compared to the other methods in the *Top1000* dataset.

	closeness	out degree	in degree	eigenvector	pagerank
closeness		7.57%	3.13%	6.74%	2.1%
out degree			10.34%	18.01%	7.20%
in degree				54.52%	77.50%
eigenvector					54.26%
pagerank					

Table 5.8: The intersection of all centrality calculations of the *Top10000* dataset are listed in this table.

to *in-degree* even 76%. The remaining comparisons evaluate almost intersections with less than 4%. The same process has been performed with the *Top10000* dataset and is illustrated in Table 5.8. The high numbers for *pagerank*, *eigenvector* and *in-degree* centrality did not change at all. The small values increased a bit to the range between 2% and 18%.

In the next step, we tried to find an overlap including all five method calculations. For the dataset *Top1000* which represent the top 1,000 results of all five calculation methods, no intersections are found. Therefore, the same process is applied to the first 10,000 results of all centrality methods, contained in the *Top10000* dataset, and 43 streamers are found, which are referenced as a potential influencers.

Type	Potential Influencer		Influencer Follower	
	Description	Count [%]	Description	Count [%]
Games	Fortnite	8.5%	Fortnite	13.4%
	Just Chatting	7.5%	Just Chatting	4.9%
	League of Legends	4.6%	Apex Legends	4.0%
	Apex Legends	4.2%	Call of Duty: Modern Warfare	3.1%
Languages	en	84.0%	en	89.2%
	es	2.6%	fr	1.8%
	de	2.3%	de	1.6%
Streamer Type	affiliate	66.3%	affiliate	94.7%
	partner	32.7%	partner	5.3%

Table 5.9: The *potential influencer* dataset is put in contrast with the *influencer follower* dataset.

5.3 Influencer Subgraph

In the following steps, we focus on potential influencers, found in the previous evaluations. Therefore, we created and explored two new subgraphs based on our findings. To obtain the relevant streamers, the top 10,000 ranked streamers for all five centrality measures were taken. The resulting subset of 32,934 streamers was achieved by merging this ranked set. In this work, the dataset is addressed as *potential influencer* network. A plot of this network is shown in Figure 5.1. In this illustration, streamers are clustered by played games. All influencers are highlighted in the network plot. In the second dataset, we want to analyze is the *influencer follower* set; it was created from the centrality calculations by intersecting all sets. The remaining subset contained 43 influencers in the network. To see the influence of their followers, we included all the following streamers in the network. The resulting network contains 75,670 streamers.

In the next step, we compared the relevant features of the network as we have done before in the previous section. For both datasets, the most played games, most spoken languages, and streamer types were evaluated. The results are shown in Table 5.9. As expected, the outcome of both evaluations is similar and not conflicting.

The game *Fortnite* is currently one of the most popular games on Twitch which can also be seen on TwitchTracker (2020a). An interesting aspect we

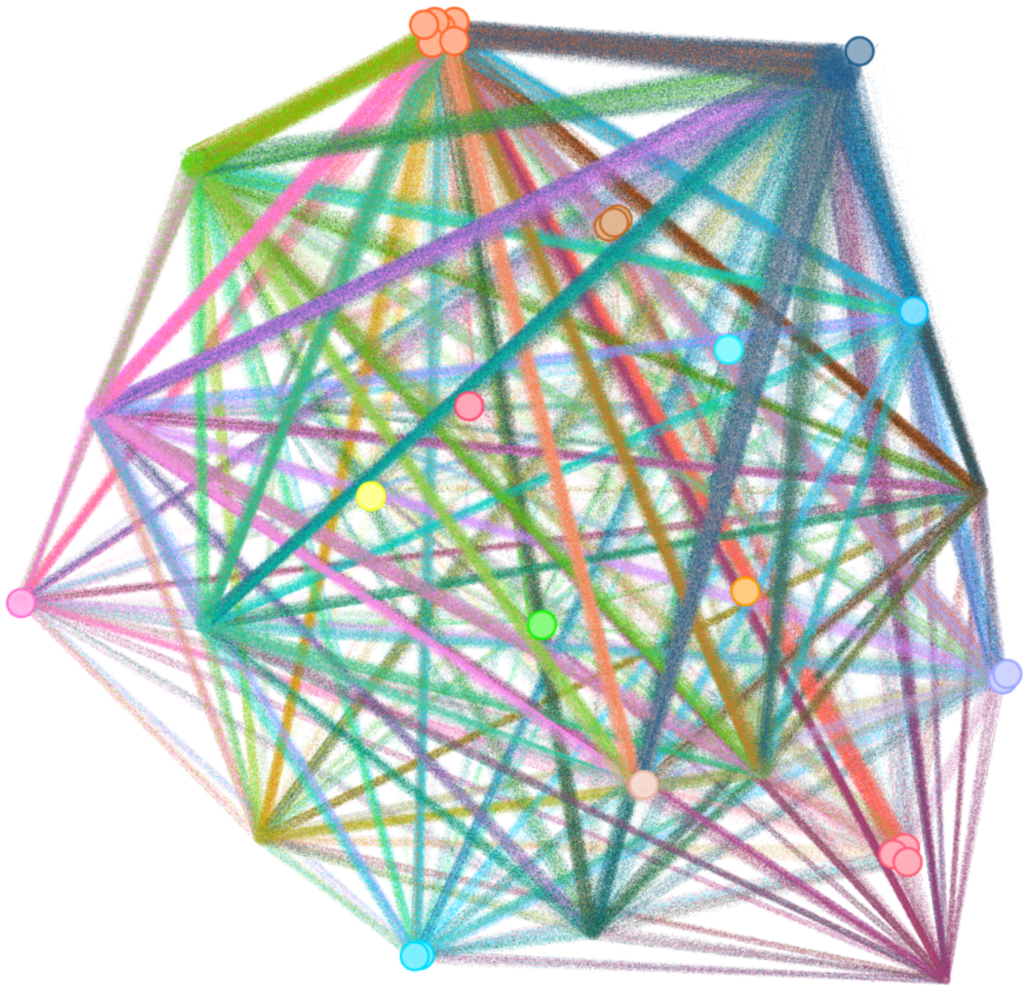


Figure 5.1: The graph shows the influencer network clustered by played games. The highlighted nodes are influential streamers. The graph contains 32,934 streamers and they are connected with 5,041,626 edges. The main graph is strongly connected except a few outliers.

found in our analysis is that most identified influencers are active in social channels such as *Just Chatting*, *Art*, and *Music*. This highlights the social aspects of Twitch, which we mentioned earlier in this thesis. Needless to say, influencers were found as well for the most popular games such as *Fortnite*. Another observation we want to point out is that the majority, about three quarters, of the 43 influencers have joined the *partner* program on Twitch. In contrast, the follower crowd of the 43 influencer are *affiliate* streamers as we can see on the results of Table 5.9.

5.3.1 Influencer Impact on Following Streamer

In the previous sections, we identified through centrality calculation a small subset of 43 influencers. In the following steps, the impact on the followership of the influencer subset is measured. Therefore, the network *influencer follower* consists of the 43 influencers including all their following streamers. To detect changes in streamer behavior, additional data was taken into account. For this experiment, previously collected stream data is used to get information about the streaming behavior. The observation focuses on different categories to detect changes in streaming behavior. The categories are divided into *New Games*, *New Viewer*, *New Follower* and *Stream Time*. The first category includes followers who start to play a new game which was played before by the influencer. *New Viewer* and *New Follower* show a rapid increase in the viewers or follower count. The threshold for marking a follower in this category is an increase of over 30% of the numbers after the friendship was forged. The same guidelines were used for observing increasing stream times after the friendship was made. The results for all influencers are shown in Table 5.10.

In order to compare the impact of influence on their followers, another network was created by picking 43 random streamers and their followers. The same analysis as described before was performed on this random streamer dataset. The results are listed in Table 5.11. Compared to the results of the influencer dataset, the random dataset is significantly smaller. The total network only contains 15,716 streamers. Another major difference affects changes of the stream time whereas at the random dataset almost none streamer were influenced. In general, influencing no streamers appear

Influencer	NEW GAMES		NEW VIEWER		NEW FOLLOWER		LONGER STREAM TIME		FOLLOWING STREAMER
mrcreeep	573	35.13%	135	8.28%	84	5.15%	325	19.93%	1631
leland	583	57.61%	116	11.46%	75	7.41%	167	16.50%	1012
kitsch	466	41.91%	40	3.60%	24	2.16%	312	28.06%	1112
grandmazcookies	276	42.33%	118	18.10%	64	9.82%	200	30.67%	652
sevinth	412	32.70%	115	9.13%	86	6.83%	234	18.57%	1260
lepslair	1,923	73.96%	176	6.77%	108	4.15%	623	23.96%	2600
porkmarshmallow	861	74.87%	127	11.04%	78	6.78%	380	33.04%	1150
radderssgaming	1,047	53.31%	169	8.60%	102	5.19%	433	22.05%	1964
raquel	1,622	39.91%	505	12.43%	385	9.47%	896	22.05%	4064
diverdragoon	1,398	79.34%	179	10.16%	101	5.73%	545	30.93%	1762
tashnarr	938	50.24%	122	6.53%	72	3.86%	393	21.05%	1867
clamtaco	1,693	63.60%	186	6.99%	120	4.51%	525	19.72%	2662
forkgirl	406	38.45%	151	14.30%	91	8.62%	234	22.16%	1056
frankthepegasus	1,821	55.65%	496	15.16%	316	9.66%	685	20.94%	3272
juganza22	1,493	41.06%	585	16.09%	467	12.84%	321	8.83%	3636
goobers515	1,153	58.59%	122	6.20%	76	3.86%	486	24.70%	1968
newowlhoodis	1,047	43.37%	229	9.49%	167	6.92%	618	25.60%	2414
blossomingsun	495	49.70%	61	6.12%	43	4.32%	208	20.88%	996
rubytrue	1,212	34.53%	413	11.77%	298	8.49%	634	18.06%	3510
chipwhitehouse	553	32.36%	83	4.86%	56	3.28%	324	18.96%	1709
mikethebard	3,777	56.09%	574	8.52%	390	5.79%	1,640	24.35%	6734
thehunterwild	2,151	43.83%	309	6.30%	207	4.22%	1,112	22.66%	4908
larryfishburger	949	49.82%	421	22.10%	296	15.54%	366	19.21%	1905
mermaidunicorn	1,756	43.74%	608	15.14%	441	10.98%	933	23.24%	4015
yosoykush	1,161	60.94%	142	7.45%	111	5.83%	343	18.01%	1905
wolvesandpizza	2,618	56.47%	958	20.66%	584	12.60%	1,109	23.92%	4636
maral	338	33.17%	69	6.77%	47	4.61%	180	17.66%	1019
dnp3	3,184	25.22%	2,670	21.15%	1,794	14.21%	4,098	32.46%	12625
cafeela	1,407	54.79%	398	15.50%	266	10.36%	511	19.90%	2568
chelsgoat	634	58.22%	156	14.33%	109	10.01%	206	18.92%	1089
greendumpling	2,291	43.44%	779	14.77%	532	10.09%	1,682	31.89%	5274
ryuthered	1,496	54.46%	170	6.19%	140	5.10%	633	23.04%	2747
starlet_blossom	994	52.29%	143	7.52%	97	5.10%	354	18.62%	1901
msashrocks	2,076	67.78%	371	12.11%	244	7.97%	792	25.86%	3063
derptyme	457	37.40%	84	6.87%	64	5.24%	230	18.82%	1222
guaconmysock	604	58.53%	205	19.86%	155	15.02%	279	27.03%	1032
meeeows	1,124	46.35%	360	14.85%	243	10.02%	540	22.27%	2425
toky	1,173	65.75%	135	7.57%	108	6.05%	357	20.01%	1784
tumbledorez	502	55.35%	117	12.90%	79	8.71%	234	25.80%	907
doubleagentsmith	1,417	53.15%	372	13.95%	225	8.44%	857	32.15%	2666
d__rich	1,142	59.14%	182	9.43%	139	7.20%	392	20.30%	1931
eldirtysquirrel	940	59.76%	177	11.25%	89	5.66%	417	26.51%	1573
xshumbax	583	33.88%	219	12.73%	164	9.53%	407	23.65%	1721

Table 5.10: This Table shows the impact of influencers on their followers. The last column shows the count of friendships to other streamers. The highlighted rows are the best-ranked streamers in the centrality calculations who also have the most following streamers.

multiple times in the random dataset whereas in the influencer results it is not present. Therefore we can indicate that the influencers have a greater impact on their follower than the randomly picked streamers.

Finally, the ranking of the influencer in the centrality calculations, are compared to create relations between the centrality calculation and the impact on the streamer in the network. In Table 5.12 the observed influencers are listed with the ranking within each centrality calculation. The highlighted influential streamer got the best rankings by calculating the median. We observed that the best-ranked streamers also have the most following streamers as can be seen in the last column at Table 5.10. By comparing the different centrality measures, no other correlation could be found with this method in view of changing behavior and impact on followers. Nevertheless, most followers started to play new games after becoming friends with an influencer which is also played by the influencer. Moreover, the viewer count on most following streamer channels increased over 30%. The best-ranked influencer in these statistics is *dnp3* who is ranked within the Top 1000 in every centrality calculation except for the out-degree calculation. This user also gained the most follower in the past months.

The Table 5.13 represents an overview comparing the random streamer network with the influencer network. This includes the average number of viewers and follower counts of all followers in the influencer and random network. As expected, the numbers for the influencer network are significantly higher than for the random network. Surprisingly, the follower count decreased in the first calculations. Further research showed that the follower count can have remarkable changes over time as it is shown in Figure 5.2. As an example, the Twitch user *gosoncio*[3] has been followed by more than 81,000 users in February but the numbers cut back to about 18,500 followers. At the time the data was collected, *gosoncio* had about 60,000 followers which resulted in a high decrease of followers compared to the current state. In order to obtain correct numbers, these users were eliminated in our calculations if the decrease of followers was more than 10% of the total count. Moreover, the viewer count on streams also depended on the previously selected data and due to high peaks in previous streams, the viewer count could decrease after following a target streamer. Com-

[3]https://www.twitch.tv/gosoncio

Influencer	NEW GAMES		NEW VIEWER		NEW FOLLOWER		LONGER STREAM TIME		FOLLOWING STREAMER
agentjkezoor	248	64.92%	45	11.78%	27	7.07%	1	0.26%	382
xhxyze	20	43.48%	12	26.09%	12	26.09%	0	0.00%	46
bigb4iley	3	50.00%	1	16.67%	1	16.67%	0	0.00%	6
killinme2	10	52.63%	2	10.53%	3	15.79%	0	0.00%	19
xander1965	12	75.00%	4	25.00%	4	25.00%	0	0.00%	16
falconfreak	4	36.36%	2	18.18%	0	0.00%	0	0.00%	11
dropthebombtv	86	56.21%	13	8.50%	7	4.58%	0	0.00%	153
thebilleh	37	72.55%	4	7.84%	6	11.76%	0	0.00%	51
inseitz	22	68.75%	7	21.88%	5	15.62%	1	3.12%	32
dewrevolution	5	26.32%	4	21.05%	3	15.79%	0	0.00%	19
rescuephlax	0	0.00%	3	27.27%	3	27.27%	0	0.00%	11
lesveinou	3	42.86%	1	14.29%	1	14.29%	0	0.00%	7
deeni4	10	58.82%	3	17.65%	1	5.88%	0	0.00%	17
spina1gaming	2	25.00%	1	12.50%	0	0.00%	0	0.00%	8
matthewkheafy	3,137	37.69%	1,111	13.35%	816	9.80%	3	0.04%	8323
consortiumgamer	29	54.72%	1	1.89%	1	1.89%	0	0.00%	53
autimatictv	3,319	63.22%	604	11.50%	480	9.14%	0	0.00%	5250
scrsy	0	0.00%	0	0.00%	0	0.00%	0	0.00%	4
esl_brekan	199	58.70%	96	28.32%	83	24.48%	0	0.00%	339
patriklezowich	1	33.33%	1	33.33%	0	0.00%	0	0.00%	3
gojiras_daddy	4	21.05%	6	31.58%	5	26.32%	0	0.00%	19
aj666666	21	67.74%	11	35.48%	7	22.58%	1	3.23%	31
schjsm	19	57.58%	8	24.24%	6	18.18%	0	0.00%	33
globailgntv	14	56.00%	3	12.00%	1	4.00%	0	0.00%	25
juanyyyyyyy	0	0.00%	0	0.00%	0	0.00%	0	0.00%	4
skysenberg1	8	29.63%	4	14.81%	3	11.11%	0	0.00%	27
kaneoser	11	18.33%	6	10.00%	1	1.67%	0	0.00%	60
theeb8	0	0.00%	1	33.33%	0	0.00%	0	0.00%	3
work_in_game	1	10.00%	5	50.00%	4	40.00%	0	0.00%	10
rippydoggy	5	71.43%	2	28.57%	3	42.86%	0	0.00%	7
xampyyyy	20	55.56%	11	30.56%	6	16.67%	0	0.00%	36
banana320510	86	68.80%	18	14.40%	15	12.00%	0	0.00%	125
toolgood	9	47.37%	2	10.53%	3	15.79%	0	0.00%	19
franchize1140	376	69.89%	29	5.39%	17	3.16%	0	0.00%	538
alecmadman	38	59.38%	14	21.88%	7	10.94%	0	0.00%	64
zoxx	0	0.00%	3	5.26%	5	8.77%	0	0.00%	57
lukiinho	0	0.00%	1	50.00%	1	50.00%	0	0.00%	2
whylace	23	42.59%	18	33.33%	18	33.33%	0	0.00%	54
linkster_br	7	28.00%	3	12.00%	3	12.00%	0	0.00%	25
emk_kalanta	17	65.38%	6	23.08%	5	19.23%	0	0.00%	26
munkmimalmira	7	25.93%	15	55.56%	9	33.33%	0	0.00%	27

Table 5.11: This Table shows the impact of the randomly picked streamers on their follower. The last column shows the count of friendships to other streamers.

Influencer	Closeness	Pagerank	Eigenvector	In degree	Out degree	Score
mrcreeep	5,418	3,753	597	4,891	2,115	3,753
leland	6,299	6,079	1,017	8,721	2,459	6,079
kitsch	5,784	5,369	701	7,793	6,766	5,784
grandmazc00kies	1,682	1,019	4,172	5,102	38	4,172
sevinth	7,168	3,737	1,366	6,685	7,182	6,685
lepslair	6,254	8,271	1,888	2,685	591	2,685
porkmarshmallow	3,739	7,871	1,147	7,487	8,059	7,487
radderssgaming	9,817	2,059	321	3,822	9,439	3,822
raquel	7,819	1,052	289	1,538	8,022	1,538
diverdragoon	4,906	4,286	371	4,429	20	4,286
tashnarr	2,890	3,109	378	4,102	7,325	3,109
clamtaco	9,708	2,386	589	2,600	5,989	2,600
forkgirl	2,179	8,891	2,114	8,252	7,393	7,393
frankthepegasus	7,382	2,336	261	2,030	7,396	2,336
juganza22	4,243	2,425	5,352	1,769	4,650	4,243
goobers515	1,238	3,153	325	3,812	4,572	3,153
newowlhoodis	2,910	5,473	545	2,940	2,199	2,910
blossomingsun	9,134	8,149	1,020	8,832	4,735	8,149
rubytrue	2,658	1,991	646	1,861	5,911	1,991
chipwhitehouse	9,675	3,069	341	4,598	5,145	4,598
mikethebard	9,323	542	72	792	1,123	792
thehunterwild	2,623	1,046	64	1,196	8,517	1,196
larryfishburger	6,523	4,871	8,058	3,993	8,139	6,523
mermaidunicorn	6,216	1,358	306	1,569	2,282	1,569
yosoykush	9,213	7,331	3,066	3,990	1,167	3,990
wolvesandpizza	7,303	4,511	2,905	1,291	4,913	4,511
maral	8,579	3,550	2,622	8,625	6,524	6,524
dnp3	291	705	688	342	9,388	688
cafeela	4,530	2,747	456	2,731	2,665	2,731
chelsgoat	9,360	5,643	1,501	7,981	2,748	5,643
greendumpling	3,615	1,557	141	1,078	135	1,078
ryuthered	3,902	3,280	537	2,497	1,044	2,497
starlet_blossom	2,525	3,556	466	3,996	9,894	3,556
msashrocks	8,476	2,689	338	2,195	7,058	2,689
derptyme	8,601	3,681	1,187	6,963	8,516	6,963
guaconmysock	3,189	8,981	4,604	8,505	3,002	4,604
meeeows	452	5,189	1,265	2,923	9,788	2,923
toky	4,381	4,237	986	4,376	6,318	4,376
tumbledorez	6,540	7,825	1,258	9,935	1,967	6,540
doubleagentsmith	5,509	5,571	860	2,595	2,843	2,843
d__rich	300	5,358	4,228	3,910	5,930	4,228
eldirtysquirrel	470	4,104	460	5,127	5,411	4,104
xshumbax	5,003	4,809	4,037	4,559	5,638	4,809

Table 5.12: The Table lists the ranking of influencers for each centrality measure. The last column displays the calculated score by taking the mean value of the other rankings. The highlighted streamers are ranked best and also have the most streamer followers.

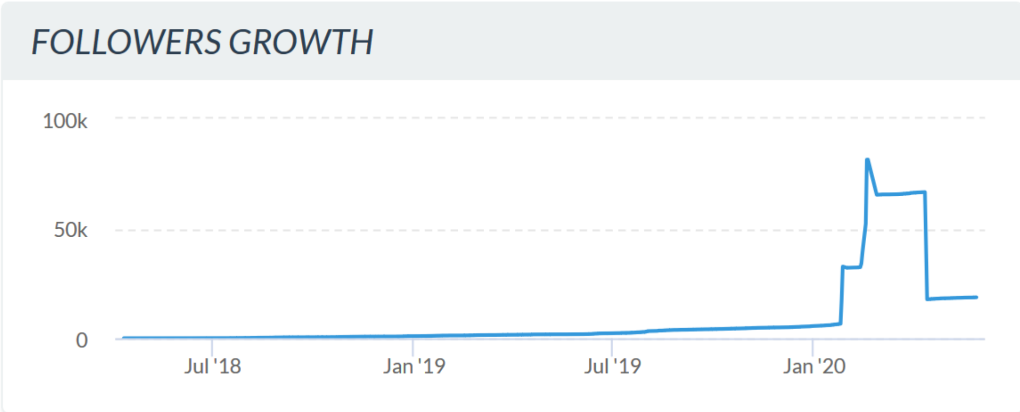


Figure 5.2: This statistic shows the follower count over time for the Twitch user *gosoncio* (TwitchTracker, 2020d).

	Influencer Follower			Random Streamer		
	Average	Median	Changed	Average	Median	Changed
Average Viewer	172,610	6,297	11.20%	109,105	4,654	11.40%
Average Follower	4,498	399	14.35%	3,375	186.5	4.31%
Average Viewer on Streams	107	34	35.71%	55	14.5	9.69%
Average Stream Time [Min.]	188	174	−0.43%	30	8.1	−14.09%

Table 5.13: The numbers in this table are the average values of the follower crowd of the influencer and random dataset. The numbers in percentage give the ratio of how the numbers changed after friendships were forged.

paring the average viewer count for both datasets the increase was about 11 % but the influencer follower dataset gained significantly more average followers, average viewers for every stream, and longer stream times than the random dataset. Nevertheless, the changes in the influencer network showed a greater impact on their follower compared to the random streamer network.

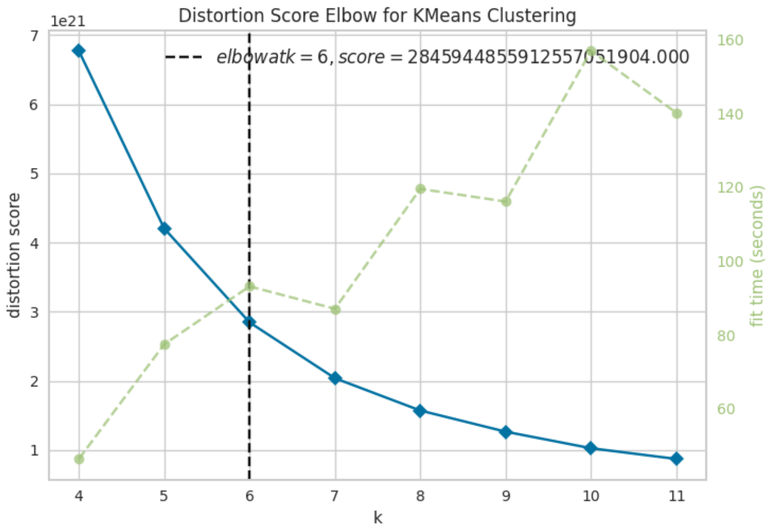


Figure 5.3: This diagram shows the distortion score elbow for the *k*-means clustering. The resulting cluster count is six where the distortion score intersects the fit time line. For our dataset we calculated the optimal *k*-score from four clusters to 11.

5.3.2 Cluster Analysis of Streamer

Regarding the previous analysis, where we could not find any correlations between a centrality calculation and features, we introduced a cluster analysis using *k*-means clustering. For this analysis, several features were taken into account like follower count, viewer count, played games, and stream time. By using the Elbow Method the best number of clusters was estimated (Marutho et al., 2018; Syakur et al., 2018). For this calculation, a Python library from *scikit-yb* developers[4] was used. The resulting diagram, shown in Figure 5.3, indicated six clusters as the optimal *k*-means cluster count.

The *k*-means algorithm[5] grouped all streamers in the six previously defined clusters according to their features. The Table 5.14 lists the result of the analysis. By comparing the six clusters we could determine three larger clusters, namely *Cluster 1*, *Cluster 2* and *Cluster 4*, with more than 16,000

[4]https://www.scikit-yb.org/en/latest/api/cluster/elbow.html
[5]https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

streamers. An interesting finding we discovered by looking at the played games was that in these clusters only the first game, *Fortnite*, was played the most by 19 % of the streamers. In all other clusters, a much higher percentage of streamers have played a single game. This leads to the assumption that the more equally the played games are distributed, the more streamers are clustered which is represented by *Cluster 1*. The cluster with the second-highest viewer count, *Cluster 4*, *Fortnite* was played by 15.5 % of the streamers whereas *Cluster 2* with 19 % is ranked third. In the remaining clusters, 3 and 6, about 40 % and more have played *Fortnite*. In *Cluster 5* the second and third, most played games also had high values which led to a smaller variety of played games. Another interesting discovery we found is that the average viewer and average stream length were also significantly higher than in the other clusters with lower streamer count. These findings combined let us conclude that a wider range of played games influences the viewer count and increased it, which may also lead to longer stream times.

	Cluster 1		Cluster 2		Cluster 3		Cluster 4		Cluster 5		Cluster 6	
Streamer Count	20,259		16,284		6,580		17,379		3,091		10,957	
Avg. Viewer	42		21		18		25		13		17	
Avg. Length[6]	197		181		160		183		154		170	
Played Games	Fortnite	9.57 %	Fortnite	19.90%	Fortnite	48.23 %	Fortnite	15.51 %	Fortnite	27.01 %	Fortnite	37.50 %
	Twitch Sings	6.01 %	Twitch Sings	5.83 %	Twitch Sings	6.66 %	Twitch Sings	4.97 %	Twitch Sings	12.55 %	Twitch Sings	5.02 %
	PUBG	4.34 %	PUBG	4.62 %	Just Chatting	5.02 %	PUBG	4.63 %	Just Chatting	8.18 %	Just Chatting	4.66 %
	Just Chatting	3.35 %	Just Chatting	3.99 %	Apex Legends	3.29 %	Just Chatting	3.62 %	CoD: MW	4.49 %	PUBG	3.98 %
	GTA V	3.31 %	Overwatch[7]	3.26 %	CoD: MW	2.94 %	GTA V	2.77 %	Apex Legends	4.35 %	Apex Legends	3.11 %
	LoL	2.89 %	DbD	2.81 %	PUBG	1.74 %	DbD	2.71 %	DbD	1.97 %	CoD: MW	2.58 %
	DbD	2.51 %	GTA V	2.65 %	DbD	1.73 %	Overwatch	2.67 %	M & P Arts	1.82 %	DbD	2.50 %
	Overwatch	2.47 %	Apex Legends	2.41 %	GTA V	1.43 %	CS: GO	2.49 %	GTA V	1.52 %	Overwatch	1.89 %
	Apex Legends	2.12 %	TCRS	2.02 %	Overwatch	1.23 %	Apex Legends	2.41 %	PUBG	1.51 %	TCRS	1.73 %
	CS GO	2.00 %	CoD: MW	1.88 %	Minecraft[8]	1.13 %	LoL	2.21 %	CoD: BO4	1.49 %	GTA V	1.71 %

Table 5.14: The results of the *k*-means analysis are shown in this table. For each cluster the streamer count, average viewer, average stream length in minutes and played games are listed and compared to each other. Shortcuts used in the table: CoD: MW = *Call of Duty: Modern Warfare*, CS: GO = *Counter-Strike: Global Offensive*, TCRS = *Tom Clancy's Rainbow Six: Siege*[9], GTA V = *Grand Theft Auto V*, PUBG = *PLAYERUNKNOWN'S BATTLEGROUNDS*[10], DbD = *Dead by Daylight*[11], LoL = *League of Legends*, CoD: BO4 = *Call of Duty: Black Ops 4*[12]

6 The average stream length in minutes
7 Blizzard Entertainment , 2016. https://playoverwatch.com/.
8 Mojang Studios , 2013. https://www.minecraft.net/.
9 Ubisoft Montreal , 2015. https://www.ubisoft.com/en-gb/game/rainbow-six/siege.
10 Bluehole, Inc. , 2017. https://www.pubg.com.
11 Behaviour Interactive , 2016. https://deadbydaylight.com/en.
12 Treyarch, Beenox, Raven Software , 2018. https://www.callofduty.com/blackops4.

5.3.3 Answering Research Questions

In our work about detecting influencers on Twitch, we wanted to discover the impact of a certain group of streamers on other streamers in the network. By using SNA we identified a small subgroup of key members and analyzed how the streaming behavior of their followers has changed. In order to compare the results with other streamers, another network of randomly picked streamers was created. Then we answered the research questions defined in Chapter 1 using the results of these datasets.

RQ 1: How can we identify influential streamers on Twitch?

By comparing the results of the influencer and random streamer dataset we could see that influencers have an impact on played games, viewer counts, new followers, and playtime. Our analysis showed that after a streamer started following an influencer, the viewer and follower count increased in many cases, as shown in Table 5.10. In contrast to the randomly picked streamers, their influence was not so strong (Table 5.11). An overview of the results can be found at Table 5.13.

RQ2: How can the fastest growing influencer be determined?

For every influencer, we compared the number of new followers, viewer count, and streaming behavior. We found out that the streamer *dnp3* gained the most followers during the last few months. In January, around 60,000 users followed *dnp3*. The current follower count is more than 108,000 followers. This is an increase of 80% of the follower crowd within four months. As the calculated median score in Table 5.12 shows, *dnp3* is the best ranked influencer in our centrality calculations. Moreover, the viewer count on streams also grew with new followers. Further analysis showed that the recent stream activities were mainly social channels like *Just Chatting* and *Marbles On Stream*[13]. *Marbles On Stream* is a game in which streamer host marble races and viewers can participate by placing marbles in each race and

[13] Pixel by Pixel Studios Inc. , 2018. http://pixelbypixelcanada.com/mos.html.

receive points by their finishing position. Streamer and viewers interact with each other during the game on Twitch. These findings for *dnp3* indicated that streaming social games can lead to higher popularity on Twitch.

RQ 3: What impact do the influential streamers have on other streamers?

Comparing the ranking of the centrality calculations (Table 5.12) and the listing of influencers features (Table 5.10), we found a correlation regarding the following streamer count. High ranked influencers tended to have more following streamer than others. This led to the assumption that the actions of a high ranked streamer have a greater impact on the behavior than other streamers due to their centrality in the network. Also, the *k*-means cluster analysis showed the influence of played games on the viewership and stream length. The findings in Table 5.14 clearly indicated that a wider range of played games leads to a higher average viewer count and also longer stream time.

In this section, the results of our analysis are presented and discussed. First, several sub-networks are created to identify potential influencers. Next, the different results and its properties are compared and a small subset of 43 influencers was chosen based on the centrality calculations. Finally, the influencer graph and its impact on their follower-ship are surveyed. Moreover, all new friendships are analyzed in detail and changes in the streaming behavior are detected to measure the impact of the influencer. In the next section, some limitations of our analysis are listed.

5.4 Limitations

First, it has to be highlighted that only a snapshot of the Twitch network within a period between December 2019 and April 2020 was created. Once a user was collected, the numbers were not updated anymore. Due to the length of time needed to collect the data for the network, the popularity of a streamer may have changed. Furthermore, it cannot be guaranteed that all users were found in the Twitch network. In our work, we collected more than 60 million users from Twitch. About 820,000 are part of the *affiliate* program and another 40,327 Twitch *partner*. Another 70 million user IDs were detected but not included in this analysis. According to the high usage and popularity of Twitch, the active user count is estimated above 150 million users. Another issue that came up was the inaccurate data consistency returned by the Twitch API. This led to variant results according to the different labeling for the same games. In other words, the popular shooter gamer *Counter Strike* appeared mainly as *Counter Strike: Global Offensive* but also shortened as *Counter Strike: GO* or even *CS:GO*. During our analysis, we came across the phenomena of enormous gaps in the follower count that can appear as it is shown in Figure 5.2. The follower count on Twitch can be increased by buying follower and viewer bots (Kelly, 2019, 2020). We suspected, that the follower count was manipulated by the use of bots and the administrators of Twitch recognized it and deleted the created followers. It is hard to detect these manipulations in the analysis. Some users with high changes in their follower count are part of our results but distort the outcomes of our calculations. In our analysis, we detected several cases that we had to remove to obtain the correct results. Additionally, the received data from the Twitch API returned only the current or last played game names. There is no simple way to find out all played games for users. In our work, we used collected data provided by *own3dtv* as well, to achieve better results. The results still strongly depended on the time when the data was collected which led to one-sided results. However, further analysis can be done focusing on the identified influencer to understand their impact in more detail.

5.5 Summary

In this chapter the results of the calculation and processing performed on the dataset, as described in Chapter 4, were presented. First, a general overview was given of the analysis performed on the dataset. Next, the results of the social network analysis were introduced in more detail. By creating two datasets with 1,000 and 10,000 potential influencer, some analyses were performed and key features were compared. Starting with the comparison of most played games listed for all centrality calculations, the most played games were identified in the datasets which are uncontroversially *Fortnite* and *Just Chatting*. In the next step, the same comparison was performed for spoken languages and streamer types. According to the numbers, *English* is the main language in this sub-network followed by *German* and *Spanish*. Surprisingly, the proportion of *partner* and *affiliate* streamers was equally distributed in the overall calculations. In a single centrality calculation, except of *eigenvector* centrality, either *partner* or *affiliate* streamer dominated. After analyzing the occurrences of some streamer attributes, characteristic numbers of the viewer, follower, and follows count were calculated and contrasted for both datasets. Due to the different results depending on the centrality calculation, we tried to identify the overlaps in order to distinguish between streamer and influencer. For the first dataset containing the first 1,000 streamer, no overall overlap was found; moreover, this overlap between most calculations was relatively low. The intersection between the top 10,000 streamer was more successful, which resulted in a list of 43 streamers which are referenced as influencers. As a further step, two new subnetworks based on our findings were created. The first graph contained the first 10,000 streamers of all five centrality calculations. The second subgraph was based on the 43 influencers and their follower-ship to determine the impact of the influencer on their follower. The results were visualized using *Gephi* by highlighting the key member in the network. Furthermore, changes in the streaming behavior of the follower-ship of an influencer were evaluated. In order to measure the impact on the follower crowd, a compare able dataset consisting of 43 randomly picked players with their followers was chosen. All followers of the chosen 43 streamers were analyzed by observing the streaming behavior including played games, viewer, follower, and stream time. With this information about when the friendship had started, we were able to contrast the behavior streaming before and after the friendship. The results of the influencer dataset showed a significantly higher impact in the follower crowd than in the random streamer dataset. These changes are referable to the impact on the followers of the identified influencers. Furthermore, we found an association between the best-ranked streamers of the centrality calculations and the most popular influencers in our dataset. The last observation on the influencer dataset was a cluster analysis to detect behavior patterns by using a k-means algorithm and determined six clusters. The findings showed that the more equal the played games are distributed, the more streamers are clustered together. Moreover, we discovered that clusters with a higher streamer count also have more viewers on average and a longer stream time. Concluding this chapter, limitations of our study were pointed out. In respect of this, our results were based only on the collected dataset, which is a snapshot of the Twitch network. The actual size of Twitch is estimated as smuch bigger than what we collected. Nevertheless, the most central part of the network was collected including streamer with the most activities.

6 Conclusion and Outlook

In the last chapter of this thesis, we point out our achieved results and give an outlook for further research. The previous chapter listed all results and findings including some limitations. Next, the conclusions will be described in detail in this chapter. Finally, a perspective of further analysis concerning this study is given.

6.1 Conclusion

In this thesis, the streaming platform Twitch was analyzed to find key members in the network and their impact on the network. Therefore, a suitable dataset was collected from the streaming platform using the Twitch API. The resulting social network contained more than 60 million which included 40,348 partner streamer. According to Twitch statistics by Iqbal (2020), there are currently around 41,000 active partner which leads to the assumption that a main part of the network was collected. Analyzing this social network by applying SNA provided a small subset of streamers who can be identified as influencers in the network. To measure the impact of these influencers in the network, the streaming behavior of their follower was analyzed to detect the changes caused by the influencers. The results showed that most streamers who became friends with an influencer started playing new games which were played before by the influencer. Moreover, the viewer counts on the streamers's channel increased for these players. This thesis showed and compared different methods to calculate the most central nodes in a network. The outcome indicated that every centrality calculation favored other streamers in the network. Finally, 43 random streamers with their followers were picked from the whole network in order to compare the influence of the sstreamers on their follower. The findings show that the numbers of influencer followers are significantly higher than the numbers of random streamer followers. In other words, the influencer has a greater impact on their follower crowd than other streamers in the network. The following cluster analysis on the influencer data set showed us that playing multiple games addressed more viewers than focusing on a single game.

6.2 Future Work

In future work, another snapshot should be collected and analyzed in the same way. This will enable the possibility to determine changes in the network such as game trends and streaming behaviors. Moreover, the behavior of influencers and their impact on the streaming community on Twitch can be measured. This can lead to new key members in the social network who might influence another part of the audience. Another possible network that could be created is to take the influencer from the second snapshot including their follower crowd and then an analysis can be performed to find out why a user started to follow an influencer. Besides the gaming behavior, the proportion of *affiliate* and *partner* streamers in the influencer network can be detected as well. It would be interesting to find out in general which of them has a greater impact on their followership. Moreover, no numbers were found about the size of the current social network of Twitch, which might be another topic of study for the evolution and expansion of Twitch. Furthermore, a more detailed study and analysis on the small influencer set can be executed to keep track of their streaming. The Twitch API offers so-called *Webhooks*(Twitch, 2020e) to track activities of a streamer like a changed stream or new followers. Besides, data for a specific user is available as we pointed out in Chapter 3 e.g. lists of videos or joined teams. Considering this informations, a more detailed analysis of specific streamer and their behavior can be performed. A more detailed analysis can be performed on the clustered influencer dataset by taking more data into the clustering algorithm. An interesting observation would be to find out how the 43 influencers are distributed in the clusters and how the findings of this study match with the results. We detected a rapid growth of average viewer on Twitch starting from April 2020 to June

2020 (TwitchTracker, 2020c). The numbers are by far the highest recorded viewer counts in Twitch history which might be correlated with the ongoing *COVID-19* pandemic. It might be interesting to analyze changes in the streaming behavior caused by the global pandemic.

List of Figures

Bibliography

Akhtar, N., Javed, H., & Sengar, G. (2013). Analysis of facebook social network, In *2013 5th international conference and computational intelligence and communication networks*. (Cit. on pp. 2, 21).

Andre, L. (2020). *Number of Gamers Worldwide 2020: Demographics, Statistics, and Predictions*. Retrieved April 28, 2020, from https://financesonline.com/number-of-gamers-worldwide/. (Cit. on p. 4)

Bastian, M., Heymann, S., & Jacomy, M. (2009). *Gephi: An open source software for exploring and manipulating networks*. http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154. (Cit. on p. 57)

Bavelas, A. (1950). Communication patterns in task-oriented groups. *The Journal of the Acoustical Society of America*, *22*(6), 725–730. https://doi.org/10.1121/1.1906679 (cit. on pp. 19, 56)

Beauchamp, M. A. (1965). An improved index of centrality. *Behavioral Science*, *10*(2), https://onlinelibrary.wiley.com/doi/pdf/10.1002/bs.3830100205, 161–163. https://doi.org/10.1002/bs.3830100205 (cit. on p. 19)

Brandes, U. (2001). A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, *25*(2), 163–177 (cit. on p. 56).

Bründl, S., & Hess, T. (2016). Why do users broadcast? examining individual motives and social capital on social live streaming platforms., In *Pacis*. (Cit. on p. 16).

Bründl, S., Matt, C., & Hess, T. (2017). Consumer use of social live streaming services: The influence of co-experience and effectance on enjoyment, 1775–1791 (cit. on p. 9).

Canossa, A., Azadvar, A., Harteveld, C., Drachen, A., & Deterding, S. (2019). Influencers in multiplayer online shooters: Evidence of social contagion in playtime and social play, In *Proceedings of the 2019 chi conference on human factors in computing systems*, Glasgow, Scotland Uk, Association for Computing Machinery. https://doi.org/10.1145/3290605.3300489. (Cit. on pp. 21, 23, 28)

Cha, M., Haddadi, H., Benevenuto, F., & Gummadi, K. P. (2010). Measuring user influence in twitter: The million follower fallacy, In *Fourth international aaai conference on weblogs and social media*. (Cit. on pp. 29, 30).

Cha, M., Mislove, A., & Gummadi, K. P. (2009). A measurement-driven analysis of information propagation in the flickr social network, In *Proceedings of the 18th international conference on world wide web*, Madrid, Spain, Association for Computing Machinery. https://doi.org/10.1145/1526709.1526806. (Cit. on p. 29)

Churchill, B. C. B., & Xu, W. (2016). The modem nation: A first study on twitch.tv social structure and player/game relationships, In *2016 ieee international conferences on big data and cloud computing (bdcloud), social computing and networking (socialcom), sustainable computing and communications (sustaincom) (bdcloud-socialcom-sustaincom)*. (Cit. on pp. 1, 25, 27).

Cohen, E., Delling, D., Pajor, T., & Werneck, R. F. (2014). Computing classic closeness centrality, at scale, In *Proceedings of the second acm conference on online social networks*, Dublin, Ireland, Association for Computing Machinery. https://doi.org/10.1145/2660460.2660465. (Cit. on pp. 19, 55)

Dux, J. (2018). Social live-streaming: Twitch. tv and uses and gratification theory social network analysis. *Computer Science & Information Technology*, 47–61. https://doi.org/10.1080/1091367X.2010.495559 (cit. on pp. 1, 9)

ead. (2018). *Pandas pd.series.isin performance with set versus array*. Retrieved April 30, 2020, from https://stackoverflow.com/questions/50779617/pandas-pd-series-isin-performance-with-set-versus-array. (Cit. on p. 41)

Earnings, E. (2020a). *Esports Earnings*. Retrieved April 22, 2020, from https://www.esportsearnings.com/tournaments. (Cit. on p. 12)

Earnings, E. (2020b). *Top Games Awarding Prize Money*. Retrieved April 22, 2020, from https://www.esportsearnings.com/games. (Cit. on p. 14)

Ediger, D., Jiang, K., Riedy, J., Bader, D. A., Corley, C., Farber, R., & Reynolds, W. N. (2010). Massive social network analysis: Mining twitter for social good, In *2010 39th international conference on parallel processing*. (Cit. on pp. 2, 21).

Euler, L. (1741). Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 128–140 (cit. on p. 17).

Freeman, L. C. (1977). A set of measures of centrality based on betweenness. *Sociometry*, 35–41 (cit. on p. 20).

Gandolfi, E. (2016). To watch or to play, it is in the game: The game culture on twitch.tv among performers, plays and audiences. *Journal of Gaming & Virtual Worlds*, *8*(1), 63–82. https://doi.org/doi:10.1386/jgvw.8.1.63_1 (cit. on p. 5)

Gandolfi, E. (2017). Enjoying death among gamers, viewers, and users: A network visualization of dark souls 3 's trends on twitch.tv and steam platforms. *Information Visualization*, *17*, 147387161771707. https://doi.org/10.1177/1473871617717075 (cit. on p. 23)

Gephi. (2017). *Supported graph formats*. Retrieved April 30, 2020, from https://gephi.org/users/supported-graph-formats/. (Cit. on pp. 58, 59)

Harpstead, E., Rios, J. S., Seering, J., & Hammer, J. (2019). Toward a twitch research toolkit: A systematic review of approaches to research on game streaming, In *Proceedings of the annual symposium on computer-human interaction in play*, Barcelona, Spain, Association for Computing Machinery. https://doi.org/10.1145/3311350.3347149. (Cit. on pp. 1, 26, 27)

Hilvert-Bruce, Z., Neill, J., Sjöblom, M., & Hamari, J. (2018). Social motivations of live-streaming viewer engagement on twitch. *Computers in Human Behavior*, *84*, 58–67. https://doi.org/10.1016/j.chb.2018.02.013 (cit. on p. 15)

igraph core team. (2020). *Igraph - the network analysis package*. Retrieved April 30, 2020, from https://igraph.org/. (Cit. on p. 50)

Iqbal, M. (2020). *Twitch Revenue and Usage Statistics (2020)*. Retrieved April 22, 2020, from https://www.businessofapps.com/data/twitch-statistics. (Cit. on pp. 4, 10, 12, 87)

Jamali, M., & Ester, M. (2010). A matrix factorization technique with trust propagation for recommendation in social networks, In *Proceedings of the fourth acm conference on recommender systems*, Barcelona, Spain, Association for Computing Machinery. https://doi.org/10.1145/1864708.1864736. (Cit. on p. 29)

Jia, A. L., Shen, S., Bovenkamp, R. V. D., Iosup, A., Kuipers, F., & Epema, D. H. J. (2015). Socializing by gaming: Revealing social relationships in multiplayer online games. *ACM Trans. Knowl. Discov. Data*, *10*(2). https://doi.org/10.1145/2736698 (cit. on p. 24)

Juho, H., & Max, S. (2017). What is esports and why do people watch it? *Internet Research*, *27*(2), 211–232. https://doi.org/10.1108/IntR-04-2016-0085 (cit. on p. 14)

Jung, M.-G., Youn, S.-A., Bae, J., & Choi, Y.-L. (2015). A study on data input and output performance comparison of mongodb and postgresql in the big data environment. https://doi.org/10.1109/DTA.2015.14. (Cit. on p. 35)

Kaytoue, M., Silva, A., Cerf, L., Meira, W., & Raissi, C. (2012). Watch me playing, i am a professional: A first study on video game live streaming, In *Proceedings of the 21st international conference on world wide web*, Lyon, France, Association for Computing Machinery. https://doi.org/10.1145/2187980.2188259. (Cit. on pp. 24, 27)

Kelly, T. (2019). *Twitch Viewer Bot – Read Before Twitch Shuts you DOWN!* Retrieved May 22, 2020, from https://twitchfollowers.com/de/twitch-viewer-bot/. (Cit. on p. 84)

Kelly, T. (2020). *How to get Free Twitch Followers?* Retrieved May 22, 2020, from https://twitchfollowers.com/free-twitch-followers/. (Cit. on p. 84)

Krause, J., Croft, D. P., & James, R. (2007). Social network theory in the behavioural sciences: Potential applications. *Behavioral Ecology and Sociobiology*, *62*(1), 15–27 (cit. on p. 21).

Lalani, F. M., Kommiya Mothilal, R., & Pal, J. (2019). The appeal of influencers to the social media outreach of indian politicians, In *Conference companion publication of the 2019 on computer supported cooperative work and social computing*, Austin, TX, USA, Association for Computing Machinery. https://doi.org/10.1145/3311957.3359484. (Cit. on p. 29)

Leibzon, W. (2016). Social network of software development at github, In *2016 ieee/acm international conference on advances in social networks analysis and mining (asonam)*. (Cit. on pp. 2, 22).

Leskovec, J. (2019). *Snap.py - snap for python*. Retrieved April 30, 2020, from https://snap.stanford.edu/snappy/. (Cit. on p. 50)

Leskovec, J., & Sosič, R. (2016). Snap: A general purpose network analysis and graph mining library (2016/10/03) [1[PII]]. *ACM transactions on intelligent systems and technology*, *8*(1), 1. https://doi.org/10.1145/2898361 (cit. on pp. 52, 53)

Lessel, P., Mauderer, M., Wolff, C., & Krüger, A. (2017). Let's play my way: Investigating audience influence in user-generated gaming live-streams, In *Proceedings of the 2017 acm international conference on interactive experiences for tv and online video*, Hilversum, The Netherlands, Association for Computing Machinery. https://doi.org/10.1145/3077548.3077556. (Cit. on pp. 26, 27)

Lusher, D., Robins, G., & Kremer, P. (2010). The application of social network analysis to team sports. *Measurement in Physical Education and Exercise Science*, *14*(4), https://doi.org/10.1080/1091367X.2010.495559, 211–224. https://doi.org/10.1080/1091367X.2010.495559 (cit. on p. 22)

Maharani, W., Adiwijaya, & Gozali, A. A. (2014). Degree centrality and eigenvector centrality in twitter, In *2014 8th international conference on telecommunication systems services and applications (tssa)*. (Cit. on p. 20).

Marutho, D., Hendra Handaka, S., Wijaya, E., & Muljono. (2018). The determination of cluster number at k-mean using elbow method and purity evaluation on headline news, In *2018 international seminar on application for technology of information and communication*. (Cit. on p. 79).

MongoDB. (2020). *Document*. Retrieved April 30, 2020, from https://docs.mongodb.com/manual/core/document/. (Cit. on p. 39)

Nazir, A., Raza, S., & Chuah, C.-N. (2008). Unveiling facebook: A measurement study of social network based applications, In *Proceedings of the 8th acm sigcomm conference on internet measurement*, Vouliagmeni, Greece, Association for Computing Machinery. https://doi.org/10.1145/1452520.1452527. (Cit. on pp. 2, 21)

Nuutila, E., & Soisalon-Soininen, E. (1994). On finding the strongly connected components in a directed graph. *Information Processing Letters*, *49*(1), 9–14. https://doi.org/https://doi.org/10.1016/0020-0190(94)90047-7 (cit. on p. 19)

Ortiz-Ospina, E. (2019). *The rise of social media*. Retrieved April 30, 2020, from https://ourworldindata.org/rise-of-social-media. (Cit. on pp. 5, 6)

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). *The pagerank citation ranking: Bringing order to the web.* (Technical Report No. 1999-66) [Previous number = SIDL-WP-1999-0120]. Stanford InfoLab. Previous number = SIDL-WP-1999-0120. Stanford InfoLab. http://ilpubs.stanford.edu:8090/422/. (Cit. on p. 21)

Pires, K., & Simon, G. (2015). Youtube live and twitch: A tour of user-generated live streaming systems, In *Proceedings of the 6th acm multimedia systems conference*, Portland, Oregon, Association for Computing Machinery. https://doi.org/10.1145/2713168.2713195. (Cit. on pp. 7, 16)

realead. (2020). *Cykhash - cython wrapper for khash*. Retrieved April 30, 2020, from https://www.github.com/realead/cykhash. (Cit. on p. 40)

Riquelme, F., & González-Cantergiani, P. (2016). Measuring user influence on twitter: A survey. *Information Processing & Management*, *52*(5), 949–975. https://doi.org/https://doi.org/10.1016/j.ipm.2016.04.003 (cit. on pp. 28, 30)

Romero, D. M., Meeder, B., & Kleinberg, J. (2011). Differences in the mechanics of information diffusion across topics: Idioms, political hashtags, and complex contagion on twitter, In *Proceedings of the 20th international conference on world wide web*, Hyderabad, India, Association for Computing Machinery. https://doi.org/10.1145/1963405.1963503. (Cit. on p. 29)

Schubert, M. (2012). Leonhard euler und die 7 brücken von königsberg. In *Mathematik für informatiker: Ausführlich erklärt mit vielen programmbeispielen und aufgaben* (pp. 363–404). Wiesbaden, Vieweg+Teubner Verlag. https://doi.org/10.1007/978-3-8348-1995-6_15. (Cit. on p. 18)

Sharma, A., & Cosley, D. (2016). Distinguishing between personal preferences and social influence in online activity feeds, In *Proceedings of the 19th acm conference on computer-supported cooperative work & social computing*, San Francisco, California, USA, Association for Computing Machinery. https://doi.org/10.1145/2818048.2819982. (Cit. on pp. 2, 29)

Sjblom, M., & Hamari, J. (2017). Why do people watch others play video games? an empirical study on the motivations of twitch users. *Comput. Hum. Behav.*, *75*(100), 985–996. https://doi.org/10.1016/j.chb.2016.10.019 (cit. on p. 5)

Smith, T., Obrist, M., & Wright, P. (2013). Live-streaming changes the (video) game, In *Proceedings of the 11th european conference on interactive tv and video*, Como, Italy, Association for Computing Machinery. https://doi.org/10.1145/2465958.2465971. (Cit. on p. 16)

Srinivas, A., & Velusamy, R. L. (2015). Identification of influential nodes from social networks based on enhanced degree centrality measure, In *2015 ieee international advance computing conference (iacc)*. (Cit. on pp. 21, 28).

Staff, S. (2019). *With Viewership and Revenue Booming, Esports Set to Compete with Traditional Sports*. Retrieved April 22, 2020, from https://onlinebusiness.syr.edu/blog/esports-to-compete-with-traditional-sports/. (Cit. on p. 13)

Syakur, M. A., Khotimah, B. K., Rochman, E. M. S., & Satoto, B. D. (2018). Integration k-means clustering method and elbow method for identification of the best customer profile cluster. *IOP Conference Series: Materials Science and Engineering*, *336*, 012017. https://doi.org/10.1088/1757-899x/336/1/012017 (cit. on p. 79)

Ter Wal, A. L., & Boschma, R. A. (2009). Applying social network analysis in economic geography: Framing some key analytic issues. *The Annals of Regional Science*, *43*(3), 739–756 (cit. on p. 22).

Teutle, A. R. M. (2010). Twitter: Network properties analysis, In *2010 20th international conference on electronics communications and computers (conielecomp)*. (Cit. on pp. 2, 21, 28).

Tichy, N. M., Tushman, M. L., & Fombrun, C. (1979). Social network analysis for organizations. *Academy of Management Review*, *4*(4), 507–519. https://doi.org/10.5465/amr.1979.4498309 (cit. on pp. 17, 21)

Twitch. (2019). *Twitch affiliate agreement*. Retrieved April 30, 2020, from https://www.twitch.tv/p/legal/affiliate-agreement/. (Cit. on p. 10)

Twitch. (2020a). *Twitch Affiliate Program FAQ*. Retrieved April 22, 2020, from https://help.twitch.tv/s/article/twitch-affiliate-program-faq?language=en_US. (Cit. on p. 9)

Twitch. (2020b). *Twitch api guide*. Retrieved April 30, 2020, from https://dev.twitch.tv/docs/api/guide/#rate-limits. (Cit. on p. 36)

Twitch. (2020c). *Twitch api reference*. Retrieved April 30, 2020, from https://dev.twitch.tv/docs/api/reference. (Cit. on p. 37)

Twitch. (2020d). *Using the twitch api v5*. Retrieved April 30, 2020, from https://dev.twitch.tv/docs/v5. (Cit. on p. 36)

Twitch. (2020e). *Webhooks Guide*. Retrieved May 5, 2020, from https://dev.twitch.tv/docs/api/webhooks-guide. (Cit. on p. 88)

TwitchTracker. (2020a). *Most Streamed Games on Twitch*. Retrieved May 5, 2020, from https://twitchtracker.com/games/average-channels. (Cit. on p. 71)

TwitchTracker. (2020b). *Twitch Languages Statistics*. Retrieved May 5, 2020, from https://twitchtracker.com/languages. (Cit. on p. 65)

TwitchTracker. (2020c). *Twitch Statistics and Charts*. Retrieved April 22, 2020, from https://twitchtracker.com/statistics. (Cit. on pp. 7, 8, 89)

TwitchTracker. (2020d). *Twitch Statistics Gosoncio*. Retrieved May 22, 2020, from https://twitchtracker.com/gosoncio. (Cit. on p. 78)

TwitchTracker. (2020e). *Twitch streamers with the most followers*. Retrieved May 5, 2020, from https://twitchtracker.com/channels/most-followers. (Cit. on p. 42)

Ugander, J., Karrer, B., Backstrom, L., & Marlow, C. (2011). The anatomy of the facebook social graph. *CoRR*, *abs/1111.4503*arXiv 1111.4503. http://arxiv.org/abs/1111.4503 (cit. on pp. 2, 21)

Yuan, J., Chen, K., Li, W., Ji, C., Wang, Z., & Skibniewski, M. J. (2018). Social network analysis for social risks of construction projects in high-density urban areas in china. *Journal of Cleaner Production*, *198*, 940–961. https://doi.org/https://doi.org/10.1016/j.jclepro.2018.07.109 (cit. on p. 19)

Zhang, C., & Liu, J. (2015). On crowdsourced interactive live streaming: A twitch.tv-based measurement study, In *Proceedings of the 25th acm workshop on network and operating systems support for digital audio and video*, Portland, Oregon, Association for Computing Machinery. https://doi.org/10.1145/2736084.2736091. (Cit. on p. 7)

Zhao, Q., Chen, C.-D., Cheng, H.-W., & Wang, J.-L. (2018). Determinants of live streamers' continuance broadcasting intentions on twitch: A self-determination theory perspective. *Telematics and Informatics*, *35*(2), 406–420. https://doi.org/https://doi.org/10.1016/j.tele.2017.12.018 (cit. on pp. 15, 16)

Ludography

Behaviour Interactive. (2016). *Dead by Daylight* [Behaviour Interactive, Montreal, CA]. Montreal, Canada, Behaviour Interactive, Starbreeze Studios, 505 Games, Koch Media. (Cit. on p. 81).

Blizzard Entertainment. (2004). *World of Warcraft* [Blizzard Entertainment, Irvine, CA, USA.]. Irvine, CA, USA, Blizzard Entertainment. (Cit. on p. 23).

Blizzard Entertainment. (2010). *Star Craft II* [Valve, Irvine, CA, USA]. Irvine, CA, USA, Blizzard Entertainment.

Blizzard Entertainment. (2016). *Overwatch* [Blizzard Entertainment, Irvine, CA, USA.]. Irvine, CA, USA, Blizzard Entertainment. (Cit. on p. 81).

Bluehole, Inc. (2017). *PlayerUnknown's Battlegrounds* [Bluehole, Inc., Seongnam, South Korea]. Seongnam, South Korea, Bluehole, Inc. (Cit. on p. 81).

Epic Games. (2017). *Fortnite* [Epic Games, Raleigh, North Carolina]. Raleigh, North Carolina, Epic Games, Warner Bros. Interactive Entertainment. (Cit. on p. 12).

FromSoftware. (2016). *Dark Souls 3* [Bandai Namco Entertainment, Tokyo, Japan]. Tokyo, Japan, Bandai Namco Entertainment. (Cit. on p. 23).

Game Freak Inc. (1996). *Pokémon* [The Pokémon Company, Minato, Tokyo, Japan.]. Tokyo, Japan, Nintendo, The Pokémon Company. (Cit. on p. 26).

Infinity Ward. (2019). *Call of Duty: Modern Warfare* [Activision, Santa Monica, USA.]. Encino, CA, USA, Activision. (Cit. on p. 11).

Massive Entertainment. (2016). *Tom Clancy's The Division* [Ubisoft, Montreuil, FR]. Malmö, Sweden, Ubisoft. (Cit. on p. 23).

Mojang Studios. (2013). *Minecraft* [Mojang Studios, Stockholm, Sweden]. Stockholm, Sweden, Mojang Studios. (Cit. on p. 81).

Pixel by Pixel Studios Inc. (2018). *Marbles On Stream* [Pixel by Pixel Studios Inc., Toronto, CA]. Toronto, Canada, Pixel by Pixel Studios Inc. (Cit. on p. 82).

Respawn Entertainment. (2019). *Apex Legends* [Electronic Arts, Redwood City, CA, USA.]. Beverly Hills, CA, USA, Electronic Arts. (Cit. on p. 64).

Riot Games. (2009). *League of Legends* [Riot Games, Los Angeles, CA, USA.]. Los Angeles, CA, Riot Games. (Cit. on p. 23).

Rockstar North. (2013). *Grand Theft Auto V* [Rockstar Games, New York City, USA.]. Edinburgh, Scotland, Rockstar Games. (Cit. on p. 65).

Treyarch, Beenox, Raven Software. (2018). *Call of Duty: Modern Warfare* [Activision, Santa Monica, USA.]. Santa Monica, CA, USA, Activision. (Cit. on p. 81).

Ubisoft Montreal. (2015). *Tom Clancy's Rainbow Six: Siege* [Ubisoft, Montreuil, FR]. Montreal, Canada, Ubisoft. (Cit. on p. 81).

Ubisoft Quebec. (2018). *Assassin's Creed Odyssey* [Ubisoft, Montreuil, FR]. Québec, Québec, Canada, Ubisoft. (Cit. on p. 42).

Valve. (2013). *Defence of the Ancients 2* [Valve, Bellevue, Washington, US]. Bellevue, Washington, US, Valve. (Cit. on p. 12).

Valve, Hidden Path Entertainment. (2012). *Counter-Strike: Global Offensive* [Valve, Bellevue, Washington, US]. Bellevue, Washington, US, Valve.