STEAM API DATA EXTRACTION

Online Data and Collection Management

Team 8
Cas van Dijk (2029757)
Rob Esenkbrink (2070006)
Anoesjka Raateland (2074753)
Mihai Vlad Serbanescu (2064441)
Chokie Tang (2086560)

Steam API Data Extraction

1. Motivation

1.1 *For what purpose was the dataset created? Was there a specific task in mind? Was there a specific gap that needed to be filled? Please provide a description.*

In 2019, GlobalData reports indicated that the video game industry's value would exceed $300 billion by 2025, becoming one of the world's biggest industries worldwide (Lanier, 2019). The Covid-19 pandemic was as unpredictable as it was beneficial for the video game industry, with MarketWatch analyses reporting massive financial gains due to the accommodating pandemic conditions (TrtWorld, 2021). As a result, Accenture reports that as of mid-2021 the video game industry had already exceeded its forecasted $300 billion value expected for 2025 (Nolibois, 2021).

These are all impressive financial achievements that show the general public's interest in video games as an entertainment media. However, despite this remarkable growth, the video game industry at large is highly secretive with its information. This is not limited only to publisher or developer studios, but even game distribution platforms such as Steam, PlayStation Store, and others. This has been a noted problem since 2017, leaving third party groups as major procurers of data and estimations (Dring, 2017). This indicates a clear issue for any academic or industry participant that would like to investigate statistical phenomenons within the industry as the data they may require is hard to find or non-existent. Interestingly (and not necessarily related to this project), the industry's secrecy is also pointed inwards towards employees and developers with negative consequences (Schreier, 2018).

As a result of the scarce availability of video game industry data, available research is consequently influenced. Research observing how video games as media affects us mentally or cognitively is far more prevalent and varied than statistical studies done within the industry. Furthermore, statistical studies performed within the industry may very well rarely use complete datasets (archival data) due to their scarcity, opting instead for the usage of surveys and other instances of field data.

To aid against the lack of available data on the video game industry, this team proposes the creation of datasets through the use of Steam API. Steam API enables the collection of various data points pertaining to users and video games available in its online retail store. This project specifically will focus on the collection of software datapoints as provided by Steam in an attempt to create a dataset that includes a comprehensive list of software available for sale.

It is important to note several limitations in using Steam and its API. To begin with, while Steam is the leading gaming PC online store, it is by no means the only available one (i.e. Epic Store, GOG.com, etc.). Furthermore, Steam is only available on PC, other gaming platforms such as PlayStation or Nintendo Switch utilize different online stores to facilitate game sales. This can be highly relevant for some games, as they may be platform or store exclusives. Lastly, as Steam is an online store exclusively, this data will not cover physical retail sales. These are all important caveats to understand before using this dataset, as assuming similarity between game stores (either offline or online for each different platforms) is quite possibly erroneous.

The reason Steam as an online store has been chosen is not only due to its leading position on PC platforms, but also due to the fact their store is web based and uses a web API to provide data. The availability of this web API and their ISteamApps API enable this project.

We believe that such rich data from a leading business would be most helpful for entering development/publisher studios, researchers, and other prospective stakeholders.

1.2 *Who created this dataset (e.g., which team, research group) and on behalf of which entity (e.g., company, institution, organization)?*

The dataset is created by Cas van Dijk, Rob Esenkbrink, Anoesjka Raateland, Mihai Vlad Serbanescu, Chokie Tang (Team 8) of Tilburg University. It is created for the Online Data Collection and Management course of the master Marketing Analytics. The instructor for this course is Hannes Datta.

1.3 *Who funded the creation of the dataset? If there is an associated grant, please provide the name of the grantor and the grant name and number.*

No company nor person funded the creation of this dataset. Therefore, this team is completely independent in creating the data extraction code and datasets, and making them publicly available.

2. Composition

2.1 *What do the instances that comprise the dataset represent (e.g., documents, photos, people, countries)? Are there multiple types of instances (e.g., movies, users, and ratings; people and interactions between them; nodes and edges)? Please provide a description.*

Each instance in the dataset represents a software product from the Steam website. The term 'software' is used as an umbrella as Steam has the following software products: games, demos, downloadable content (DLC), mods, music, videos, and episodes. These different instances of software do not have a direct interaction nor correlation with each other. However, all of DLC, demos, episodes, mods (some, not all), and music are always related to a base game.

2.2 *How many instances are there in total (of each type, if appropriate)?*

The app_id list consists of around 138.766 software instances as of March 25, 2022. New products are often added or removed from the Steam store, which means that the number may well vary from day to day.

Due to the large amount of software instances and limits imposed by the Steam APIs, a total of 10.000 software instances have been collected. Nonetheless, the data extraction code can be changed and run manually by any user to collect more, or even all the software data. Current known limitations are a max of 10 calls per 10 seconds and 100.000 calls per day. This has been taken in account within the code by implementing 'time.sleep(1)' to avoid crossing this limit and being blocked.

2.3 *Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set? If the dataset is a sample, then what is the larger set? Is the sample representative of the larger set (e.g., geographic coverage)? If so, please describe how this representativeness was validated/verified. If it is not representative of the larger set, please describe why not (e.g., to cover a more diverse range of instances, because instances were withheld or unavailable).*

As previously mentioned, 10.000 software instances out of a total 138.766 possible have been processed and collected within this repository. It is of relevance to note that not all Steam software requests result in success, as some software is not available, either due to them not being released/public yet or having been deleted previously. To this point, the extraction code first checks whether the data retrieval was successful, and then processed it while failure results in no data object. This is achieved due to the Steam API response including a boolean success value, and the code checking for it specifically before processing the data.

2.4 *What data does each instance consist of? "Raw" data (e.g., unprocessed text or images) or features? In either case, please provide a description.*

The raw data consists of the (10.000) software instances that have been collected, with the output being saved as unprocessed text to reflect the .json output provided by the Steam APIs.

The code contained within the repository *('src/collection/parsing nested json into dataframes and excel files.py')* further makes use of this output by creating a raw dataset with the collected surface level .json attributes, and several smaller other datasets containing the nested .json attributes. Each dataset contains the column '*steam_appid*', so that any user can merge the datasets together as they wish for their own purposes. This section will further explain the datasets:

Table 1 'data/dataframe_raw.csv'

is created by collecting every surface .json attribute. The nested variables are also present as unprocessed text, which will be highlighted further. The following table describes each variable contained:

Table 1: 'dataframe_raw.csv' variables.

Variable	Description	Data Type
type	Categorical variable detailing the type of software. Can be: game, demo, dlc, mod, movie, episode or music.	String
name	Name of the piece of software.	String
steam_appid	The unique identifier number associated with each software instance.	Numeric
required_age	The required age to view the store page of the respective software.	Numeric
is_free	Indicates whether the software is free to use or not. A TRUE value indicates that the software is free, while FALSE indicates that the software must be paid for.	Boolean
detailed_description	Description as displayed in the Steam store page.	String
about_the_game	Additional software description.	String
short_description	Short description about the software.	String
supported_languages	Unprocessed text with details regarding languages supported by the software.	String
header_image	Link to the header image displayed on the software's page in .jpeg format.	String
website	Link to the website of the software.	String
mac_requirements	Presents system requirements to run the software on a Mac system (if applicable).	String
linux_requirements	Presents system requirements to run the software on a Linux system (if applicable).	String
developers	The developer/development studio of the software.	String
publishers	The business' name under which the software is published.	String
package_groups	Nested item available in 'dataframe_package_groups.csv'. More details provided in a follow-up table.	String
categories	Nested item available in 'dataframe_categories.csv'. More details provided in a follow-up table.	String
genres	Nested item available in 'dataframe_genres.csv'. More details provided in a follow-up table.	String
screenshots	Nested item available in 'dataframe_screenshots.csv'. More details provided in a follow-up table.	String

movies	Nested item available in 'dataframe_movies.csv'. More details provided in a follow-up table.	String
background	Link to the image displayed on the background of the respective software's store page in .jpeg format.	String
background_raw	Link to the full image displayed on the background of the respective software's store page in .jpeg format.	String
pc_requirements.minimum	Presents minimum system requirements to run the software on a PC system (if applicable).	String
platforms.windows	Indicates whether the software is compatible with a Windows system. A TRUE value indicates that the software is compatible, while FALSE indicates incompatibility.	Boolean
platforms.mac	Indicates whether the software is compatible with a Mac system. A TRUE value indicates that the software is compatible, while FALSE indicates incompatibility.	Boolean
platforms.linux	Indicates whether the software is compatible with a Linux system. A TRUE value indicates that the software is compatible, while FALSE indicates incompatibility.	Boolean
release_date.coming_soon	Indicates whether the release date has yet to be announced. A TRUE value indicates that the software does have an unannounced release date, while FALSE indicates an announced release date.	Boolean
release_date.date	Indicates the release date of the software in a date format.	String
support_info.url	Link to the support page of the related software.	String
support_info.email	E-mail to contact the support team in charge of the respective software.	String
content_descriptors.ids	A number related to specific content descriptors (i.e. violence, drug-use etc.). Each number represents a unique descriptor.	String
content_descriptors.notes	Additional content descriptors for the software as text.	String
collection_details.created_by	Signature indicating the scrapper used to collect the data.	String
collection_details.created_at	Date and time of data scrapping.	String
packages	Indicates to which discount/bundle package the software is related to (if any).	String
pc_requirements.recommended	Presents the recommended system requirements to run the software optimally on a PC system (if applicable).	String
price_overview.currency	Presents the purchase currency used on the software's page. Influenced by the geographical location from which the user runs the data collection code.	String
price_overview.initial	The full price of the software (divide number by 100).	Numeric
price_overview.final	The current price of the software (different only if on discount) (divide number by 100).	Numeric
price_overview.discount_percent	The current price discount out of a maximum 100 (if any).	Numeric
price_overview.initial_formatted	The full price of the software, formatted. Only has a value if the initial price is different than the final price.	String
price_overview.final_formatted	The final price of the software, formatted.	String
recommendations.total	The total amount of reviews submitted by individual Steam users.	Numeric
achievements.total	The total number of achievements related to the software (if any).	Numeric

achievements.highlighted	Nested item that doesn't occur often and therefore isn't taken into account while parsing. Although it can be manually added.	string
fullgame.appid	The steam_appid of the software that serves as the 'parent' of this software (if type is DLC, demo, episode, mod, and music).	Numeric
fullgame.name	The name of the 'parent' software (if type is DLC, demo, episode, mod, and music).	String
mac_requirements.minimum	Presents minimum system requirements to run the software on a Mac system (if applicable).	String
mac_requirements.recommended	Presents the recommended system requirements to run the software optimally on a Mac system (if applicable).	String
linux_requirements.minimum	Presents minimum system requirements to run the software on a Linux system (if applicable).	String
linux_requirements.recommended	Presents the recommended system requirements to run the software optimally on a Linux system (if applicable).	String
legal_notice	Legal notice of copyright ownership of software related assets.	String
controller_support	Indicates whether the software has any controller support included.	String
reviews	Displays reviews from external review websites.	String
metacritic.score	Displays the user-score of the software on Metacritic (if available).	Numeric
metacritic.url	Link to the Metacritic page of the software.	String
demos	Demo-related text displayed on the software store page.	String
dlc	Lists the steam_appids of DLCs related to the software (if any).	String
pc_requirements	Presents system requirements to run the software on a Windows system (if applicable).	String
ext_user_account_notice	Additional text indicating whether the Steam user needs to have a separate account on another website to utilize the software.	String
drm_notice	Additional text indicating whether the software utilizes digital rights management (DRM) software.	String
price_overview.recurring_sub	Indicates a steam_appid if the software requires a subscription to another software.	Numeric
price_overview.recurring_sub_desc	Text displaying the amount to be paid upon subscription renewal.	String

Table 2 'data/dataframe_package_groups.csv'

is created by collecting the nested attributes contained within 'packages' of 'dataframe_raw.csv'. The following table describes each variable contained:

Table 2: 'dataframe_package_groups.csv' variables.

Variable	**Description**	**Type of data**
steam_appis	The unique identifier number associated with each software instance.	Numeric
name	Value that takes the form "default", unless given a different name by the software publisher/developer	String
title	Title of the package(s) to which the software is bundled to.	String
description	Additional description of the package.	String
selection_text	Text used to display the purchase options of the package.	String
save_text	Additional text that indicates the amount of money saved by the user if the package is purchased instead of buying all software separately.	String
display_type	Indicates whether the package is displayed on the software's store page. 0 indicates that the package is not displayed, while 1 indicates that it is.	Numeric
is_recurring_subscription	Indicates whether the package is a recurring subscription. A TRUE value indicates that the package is a recurring subscription, while FALSE indicates it is not.	Boolean
subs	Nested attributes that display more data about the package. The following attributes are included in the variable: "packageid", "percent_savings_text", "percent_savings", "option_text", "option_description", "can_get_free_license", "is_free_license" and "price_in_cents_with_discount".	String

Table 3 'data/dataframe_categories.csv'

is created by collecting the nested attributes contained within '*categories'* of *'dataframe_raw.csv'*. The following table describes each variable contained:

Table 3: 'dataframe_categories.csv'

Variable	**Description**	**Type of data**
steam_appid	The unique identifier number associated with each software instance.	Numeric
id	A number related to specific category description (i.e. Single-player, Steam Cloud, Steam Trading Cards etc.). Each number represents a category descriptor.	Numeric
description	The category description related to the id number displayed as text.	String

Table 4 'data/dataframe_genres.csv'

is created by collecting the nested attributes contained within *'genre'* of *'dataframe_raw.csv'*. The following table describes each variable contained:

Table 4: 'dataframe_genres.csv' variables

Variable	**Description**	**Type of data**
steam_appid	The unique identifier number associated with each software instance.	Numeric
id	A number related to specific genre (i.e. Action, Horror etc.). Each number represents a genre.	Numeric
description	The genre related to the id number displayed as text.	String

Table 5 'data/dataframe_screenshots.csv'

is created by collecting the nested attributes contained within *'screenshots'* of *'dataframe_raw.csv'*. The following table describes each variable contained:

Table 5: 'dataframe_screenshots.csv' variables.

Variable	**Description**	**Type of data**
steam_appid	The unique identifier number associated with each software instance.	Numeric
id	The id refers to the order of the images displayed from the store for a certain software, with 1 marking the first image shown.	Numeric
path_thumbnail	The link to an image displayed on the store page of the respective software in .jpeg format.	String
path_full	The link to the full-size version of the image on the same row in .jpeg format.	String

Table 6 'data/dataframe_movies.csv'

is created by collecting the nested attributes contained within *'movies'* of *'dataframe_raw.csv'*. The following table describes each variable contained:

Table 6 dataframe_movies

Variable	**Description**	**Type of data**
steam_appid	The unique identifier number associated with each software instance.	Numeric
id	The id refers to the order of the movie/trailers video displayed from the store for a certain software, with 1 marking the first video shown.	Numeric
name	The name of the video.	String
thumbnail	The link to the thumbnail image of the respective video as displayed on the store before plying it in .jpeg format.	String
highlight	Indicates whether the video is shown in the software store page. A TRUE value indicates that the video is shown, while FALSE indicates that it isn't.	Boolean

webm.480	Link to the respective video in webm.480 format.	String
webm.max	Link to the respective video in webm.max format.	String
mp4.480	Link to the respective video in mp4.480 format.	String
mp4.max	Link to the respective video in mp4.max format.	String

2.5 *Is there a label or target associated with each instance? If so, please provide a description.*

The *'steam_appid'* is the unique identifier that is associated with each instance of software. This identifier is also featured on every single dataset within this repository created by the parsing code. Giving users the opportunity to merge datasets with different data together as needed.

2.6 *Is any information missing from individual instances? If so, please provide a description, explaining why this information is missing (e.g., because it was unavailable). This does not include intentionally removed information, but might include, e.g., redacted text.*

The code collects data on a multitude of attributes of many software instances that differ from one another. As such, data that may appear to be missing is simply data that does not exist for a specific entry's store page. A software *'type'* game and another *'type'* movie will have inherent differences in the attributes they feature. Not only that, but some games may not even feature, for example, achievements, hence its respective data in the dataset does not exist.

2.7 *Are relationships between individual instances made explicit (e.g., users' movie ratings, social network links)? If so, please describe how these relationships are made explicit.*

Software instances that do have a relationship are easily identified via *"fullgame.appid"*. This is due some software, such as DLC, requiring a base software to be accessible. For example, you may want to play 'Outer Wilds – Echoes of the Eye' because it is a fantastic DLC. Even though you may be able to buy it, you will not be able to access the content because you do not own the equally fantastic 'Outer Wilds' base game. Hence, 'Outer Wilds – Echoes of the Eye' would have a *'fullgame.appid'* in *'dataframe_raw.csv"* due to the dependence.

2.8 *Are there any recommended data splits (e.g., training, development/validation, testing)? If so, please provide a description of these splits, explaining the rationale behind them.*

The repository offers a .json file that contains all the collected data, with this any user can customize what data they would like to collect. The *'dataframe_raw'* file contains the specific .json information as a readable dataset. Given the multitude of software and attributes collected, there are many splitting possibilities, some of which can be ideal for certain research. A highly recommended split would be based on the *'type'* and *'categories'* variables in case a user desires to monitor only a specific type or certain categories of software. Naturally, many other splits are possible, such as splitting based on the TRUE and FALSE values of *'is_free'* or separating software on whether they have a value in *'fullgame.appid'* to keep only software that has dependencies.

2.9 *Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)? If it links to or relies on external resources, a) are there guarantees that they will exist, and remain constant, over time; b) are there official archival versions of the complete dataset (i.e., including the external resources as they existed at the time the dataset was created); c) are there any restrictions (e.g., licenses, fees) associated with any of the external resources that might apply to a future user? Please provide descriptions of all external resources and any*

restrictions associated with them, as well as links or other access points, as appropriate.

The output (both .json and datasets) created by this repository's code is completely reliant on Steam API and no other sources. However, the output is not entirely self-contained, as it commonly links to external websites, external media-hosting sites, and features information from external sources. Below are some examples out of many:

- The 'website' variable for example displays a URL link related to the relevant piece of software (though not always, depends on developer/publisher choice).
- Many variables, such as 'background', display a URL link to the media-hosting website 'Akamai' from which the Steam store is able to obtain the relevant images to be displayed.
- The '*metacritic.score*' is a numeric variable that is obtained from the respective software's 'Metacritic' page (if the software has one).

As such, it can be seen that the availability of variables is dependent on a mix between developer/publisher choices and external website information. There is no guarantee that such data will always exist in this form for any and all software instances. Thumbnails, images, and videos can be changed, developers/publishers can decide what information to make available or remove, even Metacritic scores can frequently change. Fortunately, this repository includes 10.000 software instances worth of data, providing some degree of archival – though no full external archival is used or referred to in this project. As of March 25, 2022, there are no hard legal/technical restrictions that are applicable to this specific instance of data collection.

2.10 *Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctor-patient confidentiality, data that includes the content of individuals non-public communications)? If so, please provide a description.*

The dataset does not contain data that might be considered confidential. The data in the dataset is also publicly available from their website. It is possible to use the Steam APIs to collect data about the users. For this process it is necessary to obtain an API key as a Steamworks partner, for which an NDA needs to be signed and payment is required.

2.11 *Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? If so, please describe why.*

To our knowledge no content is offensive, insulting, threatening, or otherwise cause anxiety, if viewed directly. However, however steam does include age restriction ('*required_age*') for some of its software – as such, some link images may contain inappropriate content.

2.12 *Does the dataset relate to people? If not, you may skip the remaining questions in this section.*

The dataset does not relate to people. Therefore, questions 2.12 till 2.15 are not answered and have been removed.

3. Collection Process

3.1 *How was the data associated with each instance acquired? Was the data directly observable (e.g., raw text, movie ratings), reported by subjects (e.g., survey responses), or indirectly inferred/derived from other data (e.g., part-of-speech tags, model-based guesses for age or language)? If data was reported by subjects or indirectly inferred/derived from other data, was the data validated/verified? If so, please describe how.*

The data associated with each instance was acquired through '*src/collection/collect appdetails.py*'. The data was immediately observable as a .json file raw text, as the attributes used were to some extent self-explanatory and understandable.

3.2 *What mechanisms or procedures were used to collect the data (e.g., hardware apparatus or sensor, manual human curation, software program, software API)? How were these mechanisms or procedures validated?*

Throughout the process of collecting data only the Steam APIs were used. To validate the resulting data (whether as a raw .json file or in .csv format), the team observed several software store pages on Steam and compared them with the code output. The APIs had proven to be reliable in collecting the correct data – though it must be noted as previously that this data most certainly can change at any time.

3.3 *If the dataset is a sample from a larger set, what was the sampling strategy (e.g., deterministic, probabilistic with specific sampling probabilities)?*

The datasets contained within this repository are a sample of the most recently released 10.000 pieces of software made available on Steam. It is important to note though that users can re-run the code or change the collection limit token (in *'src/collection/collect appdetails.py')* to get more, or even all, of the software data made available by Steam.

3.4 *Who was involved in the data collection process (e.g., students, crowdworkers, contractors) and how were they compensated (e.g., how much were crowdworkers paid)?*

Other than the creators of this project, no other person or entity was involved in facilitating this data collection. As no crowdfunding or employments were necessary for this project, monetary compensation is not a subject that is applicable here.

3.5 *Over what timeframe was the data collected? Does this timeframe match the creation timeframe of the data associated with the instances (e.g., recent crawl of old news articles)? If not, please describe the timeframe in which the data associated with the instances was created.*

The code collects the data in real time, which is reflected by the scrapping timestamp within '*dataframe_raw*'. This specific set of 10.000 software data has been collected from March 23 to March 25, 2020.

3.6 *Were any ethical review processes conducted (e.g., by an institutional review board)? If so, please provide a description of these review processes, including the outcomes, as well as a link or other access point to any supporting documentation.*

This question is not applicable, as there were no ethical review processes conducted or necessary to begin with.

3.7 *Does the dataset relate to people? If not, you may skip the remaining questions in this section.*

The dataset does not relate to people. Therefore, questions 3.8 till 3.12 will not be answered and have been removed.

4. Preprocessing, cleaning, labeling

4.1 *Was any preprocessing/cleaning/labeling of the data done (e.g., discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of missing values)? If so, please provide a description. If not, you may skip the remainder of the questions in this section.*

As noticeable in the repository, the code used is split into several different files. The most relevant to take note of in terms of preprocessing is *'src/collection/collect all app id's.py'*. It is essential to collect all *'steam_appid'* available on the Steam store first and foremost. This is done to facilitate the creation of a code that loops through these ids and collects the data.

In terms of cleaning, it is important to note that some of the '*app_id*' responses return '*succes: FALSE*' with an empty object. These responses are not collected and skipped throughout the process. The team has decided to leave the data output for collected *'app_id'* as is to avoid deleting potentially valuable information that users may seek out.

In terms of labelling, all of the variables are as they appear as attributes in Steam API's .json output, as they are relatively self-explanatory when compared with the Steam store displays.

4.2 *Was the "raw" data saved in addition to the preprocessed/cleaned/labeled data (e.g., to support unanticipated future uses)? If so, please provide a link or other access point to the "raw" data.*

For the collection of all data a simple .json list containing only '*steam_appid*' is saved as *'data/app_ids.json'*. The raw datafile containing the full raw data is saved as '*data/raw_data.json'*. In addition, a simple .json list containing all collected app id's is saved as *'data/collected_ids.json'*.

4.3 *Is the software used to preprocess/clean/label the instances available? If so, please provide a link or other access point.*

No other software is used within this project to further facilitate preprocessing, cleaning, and labeling.

5. Uses

5.1 *Has the dataset been used for any tasks already? If so, please provide a description.*

The dataset has not been used for any tasks at this time.

5.2 *Is there a repository that links to any or all papers or systems that use the dataset? If so, please provide a link or other access point.*

As the data output of this repository is very recent, this question is not applicable – there is no repository available yet for papers and systems that use the data scraped in this project.

5.3 *What (other) tasks could the dataset be used for?*

The dataset could be used for several tasks. From a managerial perspective, the dataset can help provide a detailed general landscape of Steam as a store (i.e. general pricing outlooks, genre statistics, DLC availability, etc.). It can help prospective entrants facilitate planning – for instance, how would a game released without a publisher succeed with reviews as reference, what is the general pricing and/or discounting trends in that case? Apart from managerial use, there are academic benefits as well. Researchers can modify and couple this dataset with other datasets that have parameters they would like to analyze. Studies about how the number of articles on a video game interact with the number or content of reviews, or how the genre of a game may showcase differences in reviewing behaviors may now become a possibility as the daunting task of acquiring data is reduced significantly through the availability of this dataset.

Another interesting benefit is the potential improvement of already existing estimation formulas and tools or the creation of new ones. For example, the Boxleiter method allows the estimation of video game units sold (and in turn video game revenue) by using the number of reviews a game has on Steam and multiplying it by a number that differs based on release year, genre, and others (Carless, 2020). This alone is invaluable in an industry where sales figures are hard to acquire for analyses. Nonetheless, making more data available publicly can only serve to improve and correct such methods and our understanding of the phenomena around them.

Suppose that more data availability could even lead to accurate estimates of marketing spending on individual games, downloadable content, and remastered games – this is the ideal towards which we hope to contribute.

5.4 *Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses? For example, is there anything that a future user might need to know to avoid uses that could result in unfair treatment of individuals or groups (e.g., stereotyping, quality of service issues) or other undesirable harms (e.g., financial harms, legal risks) If so, please provide a description. Is there anything a future user could do to mitigate these undesirable harms?*

To our knowledge, there is nothing about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses negatively (be it research or otherwise).

5.5 *Are there tasks for which the dataset should not be used? If so, please provide a description.*

As noticeable within the dataset, several output variables are not only in string format, but contain text from the creators of software meant to describe and entice users into purchasing their product. Just as well, review text from professional review aggregators is provided. This enables potential users of this repository to utilize text-mining techniques to obtain new data and knowledge for subsequent use.

Aside from text, many variable outputs are in link format. Linking either to .jpeg images or videos that are used to showcase and market the software on its store page. Users can potentially download these files and run image or video analysis software or techniques resulting in new data and knowledge for subsequent use.

Lastly, it is important to note a potential caveat: Steam does not allow the usage of the API collected data for commercial purposes. This must be kept in mind to avoid potential legal ramifications with Steam as a company.

6. Sources

Accenture. (2021, April 29). *Global Gaming Industry Value Now Exceeds $300 Billion New Accenture Report Finds*. Accenture Newsroom. Retrieved March 25, 2022, from https://newsroom.accenture.com/news/global-gaming-industry-value-now-exceeds-300-billion-new-accenture-report-finds.htm

Carless, S. (2020, August 3). *How that game sold on Steam, using the "NB number"*. GameDiscoverCo. Retrieved March 25, 2022, from https://newsletter.gamediscover.co/p/how-that-game-sold-on-steam-using?utm_source=url

Dring, C. (2017, February 15). *Why won't the games industry share its digital data?* GamesIndustry.Biz. Retrieved March 25, 2022, from https://www.gamesindustry.biz/articles/2017-02-15-why-is-the-games-industry-still-struggling-with-digital-data

Lanier, L. (2019, May 1). *Video Games Could Be a $300 Billion Industry by 2025 (Report)*. Variety. Retrieved March 25, 2022, from https://variety.com/2019/gaming/news/video-games-300-billion-industry-2025-report-1203202672/

Schreier, J. (2018, May 13). *Some Reasons Why The Games Industry Is So Secretive (And Why Maybe It Shouldn't Be)*. Kotaku. Retrieved March 25, 2022, from https://kotaku.com/some-reasons-why-the-games-industry-is-so-secretive-an-1825929139

Video game industry: Winners in the pandemic. (2021, February 12). TRTWorld. Retrieved March 25, 2022, from https://www.trtworld.com/life/video-game-industry-winners-in-the-pandemic-44119