A3_ODL1_Data_Analysis_Project

October 25, 2021

0.1 P4DS (ODL1) Assignment 3

1 Data Analysis Project

Notebook template design: Brandon Bennett (2020.11.03, revised 2021.03.02)

2 Sentiment analysis applied to the Videogame Industry: The audience reaction to Mario+Rabbids: Sparks of Hope's Ubisoft announcement

Student: Gianmarco Picarella

Email: od21gp@leeds.ac.uk

3 Project Plan

3.1 The Data (10 marks)

3.1.1 Where the data comes from

For the purposes of this project, we need a data source that represents the overall audience interested in a particular videogame. There are multiple online platforms at our disposal: Twitch, Reddit, and YouTube are the top choices when asking for information related to the videogame industry. Among all, YouTube has by far the most diversified and reliable videogames audience data available online. Another advantage is that almost every video game announcement trailer is published on Youtube, making it a gathering and reference point for gamers.

3.1.2 How the data is organized

The data used to carry out the analyses is composed of a series of CSV formatted text files containing a list of comments referred to a specific youtube video. For our purposes, we have two different CSV files, one for each announcement trailer. Each row in the file has five columns: the ***row index****,* ***author name****,* ***comment text****,* ***like counter****, and the* ***publication date and time****. Each entry represents the previously mentioned columns using an* ***integer****,* ***string****,* ***string****,* ***integer****, and* ***UTC formatted string****.*

3.1.3 How the data is collected

The data is retrieved using ***Google's YouTube API wrapper****, available through a specific python package. Starting from the JSON obtained through the API, we copy only the fields mentioned above in two Pandas Dataframes* ***DATA_SOH*** *and* ***DATA_RKB****, where* ***SOH*** *and* ***RKB*** *stand for* ***Sparks of Hope*** *and* ***Rabbids Kingdom Battle****.* ***DATA_SOH*** *and* ***DATA_RKB*** *contains 9592 and 3707 comments, respectively.*

3.1.4 Additional supporting data

In addition to the mentioned data, the project uses other two datasets that are necessary during the transformation phase. The first dataset -called ***ENG_BAG_OF_WORDS*** *in code- contains the most frequently used English words online. The dataset is available on GitHub and comprises 466k words from the WordNet[2] Corpus. The best result has been achieved by considering only the first 80.000 entries. The second dataset is a machine learning model that* ***FastText****[2] uses to recognise languages.*

3.2 Project Aim and Objectives (5 marks)

My data analysis project aims to understand how the youtube audience reacted and perceived ***Mario+Rabbids: Sparks of Hope****, the Ubisoft videogame sequel announced during the* ***Ubisoft Forward E3 2021 event****. Being the sequel of the highly acclaimed* ***Mario+Rabbids: Kingdom Battle****, the project compares the results of the new Ubisoft's videogame with its predecessor, revealed in 2017, to depict the critical differences between the two announcements in terms of appreciation and virality by the public.*

I will consider the worldwide announcement trailers published by ***Nintendo*** *as data sources to scrape valuable information for our specific purposes.*

The project is based on a series of analyses, each focusing on answering a specific question by exploiting a subset of appropriate data starting from the set of features extrapolated from the over mentioned videos. The first objective is to determine the overall game appreciation among youtube comments through a Sentiment Analysis algorithm. As a second interesting goal, I will analyse the sentiment distribution within the first 24 hours from the announcement trailer publication. In conclusion, I will apply the over mentioned analyses to Mario+Rabbids: Kingdom Battle's announcement trailer and compare the results with those obtained for M+R: Sparks of Hope.

3.2.1 Specific Objective(s)

- **Objective 1:** *Analyse the overall sentiment distribution of Mario+Rabbids: Sparks of Hope's announcement trailer.*
- **Objective 2:** *Analyse the sentiment distribution within the first 24 hours from the announcement of Mario+Rabbids: Sparks of Hope.*
- **Objective 3:** *Compare the results with the analysis of Mario+Rabbids: Kingdom Battle's announcement trailer.*

3.3 System Design (5 marks)

3.3.1 Architecture

The project comprises four macro stages: data retrieval, transformation, analysis, and visualization.

The data retrieval phase is responsible for retrieving youtube comments from a particular video. The transformation phase carries out a classification procedure to classify comments as English or not. After that, a filtering procedure is used to remove any non-English comment from the pandas data frame. The analysis phase conducts a sentiment and frequency analysis on the transformed data frame.

At any given stage, there are multiple functions, each of them solving a specific problem. A final python function will then represent the actual stage using all the previously implemented functions as building blocks. In my opinion, this structure fits best the idea behind notebooks and prevents long code cells that would have been otherwise necessary using Python classes.

3.3.2 Processing Modules and Algorithms

- **Retrieve the comments list and build a new pandas dataframe** *We achieve this by abstracting one of the available youtube API wrappers for Python so that from a video id, the function returns a Pandas Dataframe representing the comments list.*

- **Identify and filter out non-English comments** *Each comment is classified by querying the FastText[2] library to retrieve its language. As **FastText**[2] uses a probabilistic algorithm based on machine learning, it is still possible for false-positive to occur. For this reason, we also derive and consider the bag of English words. If the number of letters belonging to the bag of words is at least 80% of the total text length, then the string is marked as English. Based on the previous results, each non-English comment is removed.*

- **Identify and filter out comments published out of a given timeframe** *Each comment has a UTC string representing the publication date and time. We will apply a pandas filtering routine to select only those comment published within a specific time frame.*

- **Apply a Sentiment Analysis algorithm on the filtered dataframe** *We carry out a Sentiment analysis using the **VaderSentiment**[3] Python library. The library is capable of assigning a normalised sentiment score to a given text. It does that by using advanced text parsers and machine learning pre-trained models. The resulting overall score will be a dictionary containing the number of positive, neutral and negative comments. We will also consider another scoring method that considers for each comment the number of likes it received.*

4 Program Code (15 marks)

4.1 Section 1: Retrieve the comments list and build a new pandas DataFrame

4.1.1 *Importing libraries*

As a first step, the following cell imports all the necessary libraries used throughout the project.

[1]:
```
# pip install vaderSentiment google-api-client fasttext wordcloud

import googleapiclient.discovery
import os
import pandas
import fasttext
import re
import datetime
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
from wordcloud import WordCloud
import requests
import numpy as np
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
```

4.1.2 *Retrieving Video IDs.*

When querying the YouTube API for specific video information, we must provide the string ID that uniquely identifies the YouTube video. In this case, we are interested in the IDs for Mario+Rabbids: Kingdom Battle and Mario+Rabbids: Sparks of Hope YouTube videos. Retrieving the ID from a URL is straightforward: it is identified by the substring starting right after "v=" -In case of multiple URL arguments, the ID string ends right before the first "?".

The following cell defines both IDs using two Python global variables.

[2]:
```
RKB_VIDEO_ID = "DqH-iwA0ZmU"
SOH_VIDEO_ID = "VHIuHMv3t88"
```

4.1.3 *Accessing the YouTube API Gateway*

*Almost every Google Online Service can be queried using different libraries and programming languages. The GoogleApiClient library is the Pythonic way to query Google Services. As with every online service, Google requires an **API Key** for each service used. Fortunately, Google allows the creation of free API Keys as long as the amount of data required monthly stays under a defined limitation. Again, creating a new API Key is straightforward: it is sufficient to log in to the Google Cloud Platform and go to the the credential page.*

Then click on "Create new credentials"

And finally "Create API Key". In the next cell, I defined a dictionary containing all the initialization data -Including my API Key- needed by the GoogleApiClient library to work correctly.

[3]:
```
API_SETTINGS = {"API_KEY": "AIzaSyDRgRzPYPdlkTC4wRCfU2l1TkSlsNYZKlI",
                "API_VERSION": "v3",
                "API_SERVICE_NAME": "youtube"}
```

4.1.4 *Initialise the GoogleApiClient's API Gateway Object*

Having all the necessary data, the next cell initialises the YouTube API object used to query the YouTube online services.

[4]:
```
YOUTUBE_API = googleapiclient.discovery.build(API_SETTINGS["API_SERVICE_NAME"],
                                               API_SETTINGS["API_VERSION"],
                                              ␣
 ↪developerKey=API_SETTINGS["API_KEY"])
```

4.1.5 *Starting making queries to the YouTube API*

We can now start making queries to the YouTube API. Unfortunately, one of the limitations imposed by the free API Key is that each response can return a maximum of 500 entries. This limitation slows down the process when parsing a video with thousands of comments.

In the next cell, we define a function called json_comments_by_video_id that, starting from a video ID and page token, returns the JSON object representing the list of comments. A page token is a bookmark used by the Youtube API, indicating the following comment's page to request. By default, providing an empty page token returns the starting page of comments.

In addition, an interesting note is that using the "snippet" settings returns only the comments which are not answers to other comments. This consideration is crucial if we consider the case in which a negative comment receives disapproval with other negative comments. To simplify the process, I decided to avoid these comments that represent a relatively small subset in most cases.

[5]:
```
def json_comments_by_video_id(video_id, page_token="", max_results=500):
    return YOUTUBE_API.commentThreads().list(videoId=video_id,
                                             part='snippet',
                                             pageToken=page_token,
                                             textFormat="plainText",
                                             maxResults=max_results).execute()
```

Let's query the youtube API and see the first 2 result.

[6]:
```
json_comments_by_video_id(RKB_VIDEO_ID, "", 2)
```

[6]:
```
{'kind': 'youtube#commentThreadListResponse',
 'etag': '2MJLgK2SZybSU-tUQVIzCoOuy9A',
 'nextPageToken': 'QURTSl9pM3dZc2p2a3N4T21tQ2lmaXhiWDcxQnptdmUzeWlNc3pVMHlZTG1zN
UotV1pUaUV0cXBCdVB0bHdOTmdrdFBnZHhXUGE4MFNOUQ==',
 'pageInfo': {'totalResults': 2, 'resultsPerPage': 2},
 'items': [{'kind': 'youtube#commentThread',
   'etag': 'Kj7IzDXWIXF2bhK_1lhCvdgUSgE',
   'id': 'UgzgEqVVwlkyi-IEtW14AaABAg',
   'snippet': {'videoId': 'DqH-iwA0ZmU',
    'topLevelComment': {'kind': 'youtube#comment',
     'etag': '-5IMbnjt9pVxhPEgcCXF7mywMYg',
     'id': 'UgzgEqVVwlkyi-IEtW14AaABAg',
     'snippet': {'videoId': 'DqH-iwA0ZmU',
      'textDisplay': '   Nintendo eu tiamo   meu Deus da coração pufavo
Nintendo   eu só muito seu fã gemer super mario Bros e agora e super mario
Odysseus   ',
      'textOriginal': '   Nintendo eu tiamo   meu Deus da coração pufavo
Nintendo   eu só muito seu fã gemer super mario Bros e agora e super mario
Odysseus   ',
      'authorDisplayName': 'Santhiago Oliveira',
      'authorProfileImageUrl': 'https://yt3.ggpht.com/ytc/AKedOLTCzKakqcG9JFuV7C
MR_-3QLobDNiiosfI71Q=s48-c-k-c0x00ffffff-no-rj',
      'authorChannelUrl': 'http://www.youtube.com/channel/UCUhi2Fnkoo5prTGW-
xySVvg',
      'authorChannelId': {'value': 'UCUhi2Fnkoo5prTGW-xySVvg'},
      'canRate': True,
      'viewerRating': 'none',
      'likeCount': 0,
      'publishedAt': '2021-10-23T01:50:50Z',
      'updatedAt': '2021-10-23T01:50:50Z'}},
    'canReply': True,
    'totalReplyCount': 0,
    'isPublic': True}},
  {'kind': 'youtube#commentThread',
   'etag': 'ryYiJl2bCO6r2y04qYEZRq7Xi0U',
   'id': 'UgwwPv7_2O3E0waj8Ad4AaABAg',
   'snippet': {'videoId': 'DqH-iwA0ZmU',
    'topLevelComment': {'kind': 'youtube#comment',
     'etag': '45sPiEMWuoQeKEFQYBZZA_4yiO8',
     'id': 'UgwwPv7_2O3E0waj8Ad4AaABAg',
     'snippet': {'videoId': 'DqH-iwA0ZmU',
      'textDisplay': "Person at Nintendo: let's make Mario use a
gun?\n\nEveryone: alright",
      'textOriginal': "Person at Nintendo: let's make Mario use a
gun?\n\nEveryone: alright",
      'authorDisplayName': 'Roopnarine Deonarine',
      'authorProfileImageUrl': 'https://yt3.ggpht.com/ytc/AKedOLRz1x07lbutpi-
BkQdoH4kGKO-bFuyB0s9c1CMgo28=s48-c-k-c0x00ffffff-no-rj',
      'authorChannelUrl':
'http://www.youtube.com/channel/UCr0EnYJf7qNed9yBTlGEg9g',
      'authorChannelId': {'value': 'UCr0EnYJf7qNed9yBTlGEg9g'},
      'canRate': True,
      'viewerRating': 'none',
      'likeCount': 1,
      'publishedAt': '2021-10-17T17:05:04Z',
      'updatedAt': '2021-10-17T17:05:04Z'}},
    'canReply': True,
    'totalReplyCount': 0,
    'isPublic': True}}]}
```

*As we can see, the result is represented as a JSON document. To exploit all the Pandas functionalities, In the next cell, I defined a function called **json_comments_to_list** that, given a JSON partial result, returns the list of dictionaries with the subset of information needed for the analysis.*

For each entry, the function checks if the keys exist and then access them. This check is mandatory to avoid runtime errors when some entries don't have a specific field available.

[7]:
```python
def json_comments_to_list(json):
    comments = []
    for row in json["items"]:
        if "topLevelComment" in row["snippet"] \
                and "snippet" in row["snippet"]["topLevelComment"]:

                comment = row["snippet"]["topLevelComment"]["snippet"]
                comment_info = {"author": comment["authorDisplayName"],
                               "text": comment["textDisplay"],
                               "likes": comment["likeCount"],
                               "date": comment["updatedAt"]
                               }
                comments.append(comment_info)
    return comments
```

*The last function defined in this section uses the above-defined functions and is called **get_video_comments_dataframe**. From a given video ID, the function returns a Pandas DataFrame containing all the comments for that particular video.*

[8]:
```python
def get_video_comments_dataframe(video_id):
    comments, ask_for_new, next_page_token = [], True, ""

    while ask_for_new:
        response = json_comments_by_video_id(video_id, next_page_token)
        comments += json_comments_to_list(response)
        ask_for_new = "nextPageToken" in response
        next_page_token = response["nextPageToken"] if ask_for_new else ""

    return pandas.DataFrame.from_dict(comments)
```

Finally, let's use the video IDs to populate the DataFrames with the comments and display them.

[9]:
```python
# ETA: Less than one minutes

RKB_DT = get_video_comments_dataframe(RKB_VIDEO_ID)

RKB_DT
```

[9]:
```
                                         author  \
0                             Santhiago Oliveira
1                            Roopnarine Deonarine
2                                     viktowire
3                            Tamara Bermudez Diaz
4                                   Dilmi AOUKLI
…                                           …
2953                               Eiiro L'Toxico
2954  Darío García-Franco Gómez de Villavedón
2955                         Chalish Fadhilah Amin
2956                                     Liam Arra
2957                               F l a p s o nn i

                                                   text  likes  \
0        Nintendo eu tiamo    meu Deus da coração …      0
1     Person at Nintendo: let's make Mario use a gun…      1
2          Truly one of the best games I've ever played.      1
3                               uymjhgyyuuooooolllopt Lu      0
4                      even after 4 years… how?      3
…                                                   …    …
2953                                    where is smash      0
2954                                               omg      0
2955                                              why?      2
2956                                                No      1
2957                                               lol      0

                      date
0     2021-10-23T01:50:50Z
1     2021-10-17T17:05:04Z
2     2021-09-27T17:23:22Z
3     2021-09-25T17:16:43Z
4     2021-09-22T08:16:05Z
…                      …
2953  2017-06-13T21:30:31Z
2954  2017-06-13T21:30:27Z
2955  2017-06-13T21:30:18Z
2956  2017-06-13T21:30:17Z
2957  2017-06-13T21:30:15Z

[2958 rows x 4 columns]
```

[10]:
```
# ETA: Less than two minutes

SOH_DT = get_video_comments_dataframe(SOH_VIDEO_ID)

SOH_DT
```

```
[10]:                          author  \
      0                       Polsity
      1                 Dakarai Green
      2          Koteswara Rao Samsani
      3                 Helem Barrios
      4       Juamarques Van den Berg
      …                             …
      7664            Ricardo Purnell
      7665                  Kalicosu
      7666                  IkerSito
      7667                     Mixer
      7668               Master Nick

                                                       text  likes  \
      0     I really we get more than 4 worlds  like last …      0
      1     The rabbids should be in alot of mario spinoff…      0
      2                                            Y mmmmhl      0
      3     Hola Mis Amigos Me Gusta Todas las Estrellas q…      0
      4     SMG is my friend Mario game of all time. I hop…      0
      …                                                 …    …
      7664                                          Polopel      0
      7665                                             Bruh      0
      7666                                            First      1
      7667                                                v      1
      7668                                             cool     19

                            date
      0     2021-10-24T15:01:18Z
      1     2021-10-22T18:55:11Z
      2     2021-10-21T23:47:25Z
      3     2021-10-21T19:10:12Z
      4     2021-10-20T16:42:05Z
      …                        …
      7664  2021-06-12T20:14:50Z
      7665  2021-06-12T20:13:22Z
      7666  2021-06-12T20:13:22Z
      7667  2021-06-12T20:13:21Z
      7668  2021-06-12T20:13:20Z

      [7669 rows x 4 columns]
```

4.2 Section 2: Identify and filter out non-English comments

4.2.1 *Download additional supporting data*

The language recognition algorithm used in the following cells needs some additional data to work. The first file we need is the Wordnet dataset containing the most frequent 466k English words. The second file is a machine learning model used by ***FastText****[2] during the language recognition* *phase. To make the following notebook work even if someone doesn't have the over-mentioned files, I defined a helper function called* ***download_file_locally*** *that downloads and saves the file from an URL and local filename in a path that is relative to the notebook location.*

[11]:
```
def download_file_locally(url, filename):
    r = requests.get(url, allow_redirects=True)
    open(filename, 'wb').write(r.content)
```

The following cell defines the URLs related to the resources needed and a local filename for each of them.

[12]:
```
ENGLISH_WORDS_URL  = "https://raw.githubusercontent.com/dwyl/english-words/
 ↪master/words_alpha.txt"
FASTTEXT_MODEL_URL = "https://dl.fbaipublicfiles.com/fasttext/supervised-models/
 ↪lid.176.bin"

ENGLISH_WORDS_FILENAME  = "english-word-frequency.txt"
FASTTEXT_MODEL_FILENAME = "fast_text_model.bin"
```

Having all the necessary data, the next cell downloads both files and saves them locally.

[13]:
```
# ETA: less than 1 minute for both files

download_file_locally(ENGLISH_WORDS_URL, ENGLISH_WORDS_FILENAME)
download_file_locally(FASTTEXT_MODEL_URL, FASTTEXT_MODEL_FILENAME)
```

4.2.2 *Initialize the additional objects*

The WordNet dataset contains more than 466k words. Some of the words are, in some cases, grammatically incorrect or represent archaic English concepts. As such, to carry out an accurate recognition implies considering only a subset of these words.

The following cell defines a function called ***load_english_bag_of_words*** *that loads the dataset and returns a subset of words based on the parameter.*

[14]:
```
def load_english_bag_of_words(URI, counter = -1):
    words_list = []
    with open(URI) as file:
        words_list = file.readlines()

    if counter > -1:
        words_list = words_list[:counter]

    return set(words_list)
```

Among different configurations, considering the first 80.000 words has shown the best results in terms of accuracy. The following cell defines a global variable containing the set of English words loaded from the dataset.

```
[15]: ENG_BAG_OF_WORDS = load_english_bag_of_words(ENGLISH_WORDS_FILENAME, 80000)
```

At this point, we can define all the remaining variables needed for this section. The next cell defines a regex object that removes every character that is not a letter and space character. With the previously downloaded model, we can also define the FastText[2] classifier used in the following cells.

```
[16]: RE_RULE = re.compile('[^a-zA-Z ]')
      LANG_CLASSIFIER = fasttext.load_model(FASTTEXT_MODEL_FILENAME)
```

```
Warning : `load_model` does not return WordVectorModel or SupervisedModel any
more, but a `FastText` object which is very similar.
```

4.2.3 *Determine if a text is non-English*

*The first concept to introduce is the idea behind the bag of words model. Given a text, a bag of words is a list containing all the words in that text. The Kaggle dataset loaded before is a bag of words representing a text with all the English words, or at least a considerable number of them. If we intersect the English bag of words with the one derived from a text, we can establish with a percentage of false-positive if the text is written in English or not. During the intersection phase it is important to consider repeated words. The next function called **list_set_intersection** returns the intersection between a list and a set.*

```
[17]: def list_set_intersection(lst, st):
          return [el for el in lst if el in st]
```

*We then need a function to remove every character that is not a letter or space. The following cell defines a function called **normalize_text** that uses the regex rule compiled before to filter all the matching characters and returns a lowercase representation.*

```
[18]: def normalize_text(text):
          # lowercase the text
          lower_txt = text.lower()
          # remove any non letter or space character from the text
          filtered_txt = RE_RULE.sub(" ", lower_txt)
          # remove spaces before and after text
          return filtered_txt.strip()
```

*If the percentage of letters shared between the two sets is greater or equal to at least 80% of the text length, we can assume the text is written in English. The following cell defines a function called **bag_of_words_from_string** that, given a string, returns its bag of words.*

```
[19]: def bag_of_words_from_string(text):
          bag_of_words = list()
          norm_text = normalize_text(text)

          for word in norm_text.split(" "):
              word = word.strip()
              if len(word) > 0:
                  bag_of_words.append(word)
```

```
    return bag_of_words

def count_bag_of_words_letters(bag_of_words):
    return sum([len(word) for word in bag_of_words])
```

The previously defined method leaves room for false-positive that can be limited indeed. FastText[2] is a machine learning classifier that, given a text, returns the first K set of languages compatible with it. Although it is robust with long sentences, the library fails even with simple sentences when the text contains less than five words. That's why in the following algorithm, I use both methods together to guarantee a much lower error rate. The algorithm first removes any character that is not a letter and space, then uses FastText[2]: if the English language is recognised, then we return True. Otherwise, we compute the bag of words intersection and consider the conditions as mentioned above.

[20]:
```
def is_text_language_english(text):
    norm_text = normalize_text(text)
    fast_text_result = LANG_CLASSIFIER.predict(norm_text, k=1)

    if "__label__en" in list(fast_text_result[0]):
        return True

    bag_of_words = bag_of_words_from_string(norm_text)

    words_intersection = list_set_intersection(bag_of_words, ENG_BAG_OF_WORDS)

    starting_letters_count = count_bag_of_words_letters(bag_of_words)
    filtered_letters_count = count_bag_of_words_letters(words_intersection)

    letters_percentage = filtered_letters_count / starting_letters_count

    return letters_percentage >= 0.8
```

Let's try the function with a couple of examples.

[21]:
```
is_text_language_english("Hello, this is an English text")
```

[21]:
```
True
```

[22]:
```
is_text_language_english("Ciao, questo è un testo in italiano")
```

[22]:
```
False
```

At this point, we have all the tools to define the function named ***filter_non_english_comments*** *that, given a Pandas DataFrame with all the comments for a particular video, returns the filtered DataFrame.*

```
[23]: def filter_non_english_comments(comments_dataframe):
          booleanSeries = comments_dataframe.apply(lambda r:␣
       ↪is_text_language_english(r["text"]), axis = 1)
          return comments_dataframe[booleanSeries]
```

4.3 Section 3: Identify and remove comments published out of a given time range

We will analyse groups of comments from the same video belonging to different time ranges in the following cells. The first thing we need to do is convert the UTC string into a timestap representation. This representation allows using the comparison operators making the filtering function a lot simpler. The next cell defines a function called ***comments_date_string_to_object*** *that does exactly that.*

```
[24]: def comments_date_string_to_timestamp(comments_dataframe):
          comments_dataframe["timestamp"] = comments_dataframe.apply(lambda r:␣
       ↪datetime.datetime.timestamp(pandas.to_datetime(r["date"])) , axis=1)
          return comments_dataframe
```

We can now define a function called ***is_comment_within_time_range*** *that, given a DataFrame and a range of time, returns the DataFrame containing the entries published within that specific range.*

```
[25]: def comments_within_time_range(comments_dataframe, time_range):
          return comments_dataframe[(comments_dataframe["timestamp"] >=␣
       ↪time_range[0]) \
                                    & (comments_dataframe["timestamp"] <=␣
       ↪time_range[1])]
```

Unfortunately the YouTube API doesn't give the possibility to know the publication date and time of a video. Nonetheless, we can still use as a reference date the first published comment.

```
[26]: def get_first_published_comment_timestamp(comments_dataframe):
          return comments_dataframe.iloc[comments_dataframe["timestamp"].
       ↪idxmin()]["timestamp"]

      def get_time_range(starting_time, duration):
          return (starting_time, starting_time + duration)
```

Let's try the function to see how many comments have been published within an hour since the first comment.

```
[27]: comments_date_string_to_timestamp(RKB_DT)
      comments_date_string_to_timestamp(SOH_DT)
```

```
[27]:                        author  \
      0                     Polsity
      1               Dakarai Green
```

```
2       Koteswara Rao Samsani
3                Helem Barrios
4     Juamarques Van den Berg
…                         …
7664           Ricardo Purnell
7665                  Kalicosu
7666                  IkerSito
7667                     Mixer
7668               Master Nick

                                                   text  likes  \
0     I really we get more than 4 worlds  like last …      0
1     The rabbids should be in alot of mario spinoff…      0
2                                             Y mmmmhl      0
3     Hola Mis Amigos Me Gusta Todas las Estrellas q…      0
4     SMG is my friend Mario game of all time. I hop…      0
…                                                  …    …
7664                                           Polopel      0
7665                                              Bruh      0
7666                                             First      1
7667                                                 v      1
7668                                              cool     19

                      date     timestamp
0     2021-10-24T15:01:18Z  1.635088e+09
1     2021-10-22T18:55:11Z  1.634929e+09
2     2021-10-21T23:47:25Z  1.634860e+09
3     2021-10-21T19:10:12Z  1.634843e+09
4     2021-10-20T16:42:05Z  1.634748e+09
…                      …           …
7664  2021-06-12T20:14:50Z  1.623529e+09
7665  2021-06-12T20:13:22Z  1.623529e+09
7666  2021-06-12T20:13:22Z  1.623529e+09
7667  2021-06-12T20:13:21Z  1.623529e+09
7668  2021-06-12T20:13:20Z  1.623529e+09

[7669 rows x 5 columns]
```

[28]:
```
RKB_first_comment_timestamp = get_first_published_comment_timestamp(RKB_DT)
one_hour_time_range         = get_time_range(RKB_first_comment_timestamp, 60*60)

comments_within_time_range(RKB_DT, one_hour_time_range)
```

[28]:
```
                                    author  \
2757                       Alpha Stoutland
2758                             SmashLiXs
2759                     Alessio Lo Bianco
```

```
2760                                  Nosidda
2761                                GamerOGuy
…                                          …
2953                           Eiiro L'Toxico
2954  Darío García-Franco Gómez de Villavedón
2955                     Chalish Fadhilah Amin
2956                                Liam Arra
2957                          F l a p s o nn i

                                                    text  likes  \
2757               Ok Nintendo & Ubisoft…I'm sold :)      0
2758  dumb people will think rabbids ripped off minions     35
2759  bellissimo!!! non vedo l'ora di acquistarlo.. …      0
2760  Love the part when Rabbid Luigi held the banan…      0
2761  The Rabbids are cringy but this looks like a g…      0
…                                                    …    …
2953                                      where is smash      0
2954                                                 omg      0
2955                                                why?      2
2956                                                  No      1
2957                                                 lol      0

                      date     timestamp
2757  2017-06-13T22:29:35Z  1.497393e+09
2758  2017-06-13T22:28:49Z  1.497393e+09
2759  2017-06-13T22:28:32Z  1.497393e+09
2760  2017-06-13T22:28:24Z  1.497393e+09
2761  2017-06-13T22:27:57Z  1.497393e+09
…                        …             …
2953  2017-06-13T21:30:31Z  1.497389e+09
2954  2017-06-13T21:30:27Z  1.497389e+09
2955  2017-06-13T21:30:18Z  1.497389e+09
2956  2017-06-13T21:30:17Z  1.497389e+09
2957  2017-06-13T21:30:15Z  1.497389e+09

[198 rows x 5 columns]
```

[29]:
```
SOH_first_comment_timestamp = get_first_published_comment_timestamp(SOH_DT)
one_hour_time_range         = get_time_range(SOH_first_comment_timestamp, 60*60)
comments_within_time_range(SOH_DT, one_hour_time_range)
```

[29]:
```
                author                                               text  \
4536  Rafael A. A. Merlo                                   Looks awesome <3
4537           DE23 :]  Mario with gun is still something that seems w…
4538      Shino BayWind                              So he came back to life
4539           Julious  Me: Mom can we get Mario Galaxy 3?\nMom: We ha…
4540        FakkaFalko  Its good, but its not as good as skylanders, f…
```

```
…                 …                                             …
7664    Ricardo Purnell                                    Polopel
7665           Kalicosu                                       Bruh
7666           IkerSito                                      First
7667              Mixer                                          v
7668        Master Nick                                       cool

      likes                  date     timestamp
4536      0  2021-06-12T21:13:17Z  1.623532e+09
4537      0  2021-06-12T21:13:11Z  1.623532e+09
4538      0  2021-06-12T21:13:01Z  1.623532e+09
4539      2  2021-06-12T21:13:01Z  1.623532e+09
4540      1  2021-06-12T21:13:01Z  1.623532e+09
…       …                     …             …
7664      0  2021-06-12T20:14:50Z  1.623529e+09
7665      0  2021-06-12T20:13:22Z  1.623529e+09
7666      1  2021-06-12T20:13:22Z  1.623529e+09
7667      1  2021-06-12T20:13:21Z  1.623529e+09
7668     19  2021-06-12T20:13:20Z  1.623529e+09

[3116 rows x 5 columns]
```

4.4 Section 4: Apply a Sentiment Analysis algorithm

VaderSentiment[3] is the Sentiment Analyser library used in this project. It has the cleanest interface among all the libraries I found and considers many side-cases and interesting details. It can understand slang, emoticons -Text and UTF-8 based-, punctuation, negations, degree modifiers and many more features that will be useful to consider during the analysis. The next cell defines the sentiment analyser object.

[30]:
```
SENTIMENT_ANALYSER = SentimentIntensityAnalyzer()
```

For each sentence given, VaderSentiment[3] returns a dictionary of four values.

[31]:
```
SENTIMENT_ANALYSER.polarity_scores("Hello everyone!")
```

[31]:
```
{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}
```

The cumulative sum of the first three values is 1.

The most interesting value is the compound value. As stated on the VaderSentiment's GitHub page[3]: "The compound score is computed by summing the valence scores of each word in the lexicon, adjusted according to the rules, and then normalised to be between -1 (most extreme negative) and +1 (most extreme positive). This is the most useful metric if you want a single unidimensional measure of sentiment for a given sentence".

In contrast, the author defines the negative, neutral and positive scores as "useful metrics if you want to analyse the context & presentation of how sentiment is conveyed or embedded in rhetoric for a given sentence". In our case, following the author's suggestion, we will use the compound value to carry out the analysis. We consider three different value ranges: negative from -1 to -0.05, neutral *from -0.05 to +0.05 and positive from 0.05 to 1. These values are suggested by the vaderSentiment's author too. The next cell defines a function called* ***comments_sentiment_analysis*** *that, given a DataFrame, returns a dictionary containing the number of comments and the text for each sentiment category. If consider_likes is set to True then each like to a particular comment is considered an additional comment with the same compound value.*

[32]:
```
def comments_sentiment_analysis(comments_dataframe, consider_likes=False):
    sentiment_result = {"count": {"neg": 0, "neu": 0, "pos": 0}, "text": {"neg":
 ↪ "", "neu": "", "pos": ""}}

    for index, row in comments_dataframe.iterrows():
        result = SENTIMENT_ANALYSER.polarity_scores(row["text"])
        compound = result["compound"]

        step = 1 + int(row["likes"]) if consider_likes else 1
        key = "neg"

        if -0.05 < compound < 0.05: key = "neu"
        elif compound >= 0.05: key = "pos"

        sentiment_result["count"][key] += step
        sentiment_result["text"][key]  += row["text"]

    return sentiment_result
```

4.5 Objective 1: Analyse the overall sentiment distribution of Mario+Rabbids: Sparks of Hope's announcement trailer

[33]:
```
E_SOH_DT = filter_non_english_comments(SOH_DT)
SA_SOH   = comments_sentiment_analysis(E_SOH_DT)
SAC_SOH  = comments_sentiment_analysis(E_SOH_DT, True)

print(SA_SOH["count"])
print("Weighted sentiment", SAC_SOH["count"])
```

```
{'neg': 1280, 'neu': 2629, 'pos': 3224}
Weighted sentiment {'neg': 12739, 'neu': 18087, 'pos': 63701}
```

4.6 Objective 2: Analyse the sentiment distribution within the first 24 hours from the announcement of Mario+Rabbids: Sparks of Hope

In the next cell, we sample weekly all the comments posted

[34]:
```
SOH_first_comment_timestamp = get_first_published_comment_timestamp(SOH_DT)

seconds_in_five_minutes = 5*60
sample_count = 24 * 12 # 12 samples per hour
```

```
start_time      = float(SOH_first_comment_timestamp)
final_time      = datetime.datetime.now().timestamp()
time_difference = final_time - start_time

SAMPLED_SOH_RES = []

for week in range(sample_count):
    time_range = (start_time, start_time + seconds_in_five_minutes)
    start_time = time_range[1]

    comments_in_range = comments_within_time_range(E_SOH_DT, time_range)
    SAMPLED_SOH_RES.append(comments_in_range)

SAMPLED_SOH_RES[:min(1, len(SAMPLED_SOH_RES))] # show the first result
```

```
[34]: [               author                                              text  \
       6624            CS YT     They announced this before E3 that's swag yo!
       6625            Jacob    WHAT THE ACTUAL HECK\n\n\n\nWHERE CAN I GET ONE
       6626   GALAXY17 GAMING                                   Mario galaxy 3
       6628       Smol Pantsu  So many morons in the comments thinking ninten…
       6629     Awesome Mario                                 this looks so good!
       …               …                                              …
       7656      Tangy #1541                                               Yes
       7658      Squirtle2005                               Looks pretty cool
       7660       OfficialNFB                                       Ayy firsy
       7666          IkerSito                                           First
       7668       Master Nick                                            cool

             likes                  date     timestamp
       6624      0  2021-06-12T20:18:20Z  1.623529e+09
       6625      0  2021-06-12T20:18:20Z  1.623529e+09
       6626      1  2021-06-12T20:18:17Z  1.623529e+09
       6628      0  2021-06-12T20:18:17Z  1.623529e+09
       6629      1  2021-06-12T20:18:16Z  1.623529e+09
       …       …                    …           …
       7656      0  2021-06-12T20:13:25Z  1.623529e+09
       7658      0  2021-06-12T20:13:34Z  1.623529e+09
       7660      0  2021-06-12T20:13:24Z  1.623529e+09
       7666      1  2021-06-12T20:13:22Z  1.623529e+09
       7668     19  2021-06-12T20:13:20Z  1.623529e+09

       [904 rows x 5 columns]]
```

4.7 Objective 3: Compare the results with the analysis of Mario+Rabbids: Kingdom Battle's announcement trailer.

[35]:
```
seconds_in_five_minutes = 5*60
sample_count = 24 * 12 # 12 samples per hour

RKB_first_comment_timestamp = get_first_published_comment_timestamp(RKB_DT)

start_time    = float(RKB_first_comment_timestamp)

E_RKB_DT = filter_non_english_comments(RKB_DT)
E_RKB_DT = comments_within_time_range(E_RKB_DT, (start_time, start_time +␣
 ↪time_difference))

SA_RKB = comments_sentiment_analysis(E_RKB_DT)
SAC_RKB = comments_sentiment_analysis(E_RKB_DT, True)

SA_RKB["count"]
```

[35]:
```
{'neg': 380, 'neu': 688, 'pos': 856}
```

[36]:
```
SAMPLED_RKB_RES = []

for week in range(sample_count):
    time_range = (start_time, start_time + seconds_in_five_minutes)
    start_time = time_range[1]

    comments_in_range = comments_within_time_range(E_RKB_DT, time_range)
    SAMPLED_RKB_RES.append(comments_in_range)

SAMPLED_RKB_RES[:min(1, len(SAMPLED_RKB_RES))] # show the first result
```

[36]:
```
[                             author  \
 2891                        Lissana
 2892                        Ravioli
 2894                    Jordan Loux
 2895                  Coop The Dawg
 2897                     Rad Master
 2898                 Mr. Splashteen
 2899                       Aaronimo
 2900                        FABULOG
 2901                           Davi
 2902           Everyday Day Coconut
 2903                Super Nerd Liam
 2904                           Ness
 2905                              ª
 2906                          Aneby
```

```
2907                      GlamGlob
2908                     BFG Plays
2910                      BlimpBoi
2911                   Dami Oyelola
2912  donaldthescottishtwin (DTST)
2913                       Shinji 
2915                    Fume-shroom
2916             Gly in the Middle
2917                     FaithyTree
2919                   Creed Bratton
2920                  Ulices linares
2921                   ZeusDahGoose
2923                   ZakdagamerTM
2924                       Ash Hage
2925                ElPapasitoShulo
2927                         Robert
2928                           Nexo
2929            Leonardo Castaneda
2930                SticEfragFRENZY
2931                          alo21
2932                          taiki
2933                      andretits
2934                   Diego Moyano
2935                           Hyde
2937                  LavenderTease
2938            The Turbine Turnip
2939                       Fabrizio
2940                       Shinji 
2942                    Floppy Disk
2943                      andretits
2944                       Pika Boi
2945                     George 727
2946                           Jude
2947                thelonelyfish0
2948                            kev
2949                  Rachus Sendou
2951                   German Parra
2952              Henrique Maverick
2953                Eiiro L'Toxico
2955           Chalish Fadhilah Amin
2957              F l a p s o nn i

                                                  text  likes  \
2891  That feel when this was the best game from yes…      0
2892                           MY WALLET! IT'S DYING!!     17
2894              Bowser must be so confused right now.      0
2895                        I would love to play that game     10
```

```
2897  Well at least Luigi and Yoshi aren't being lef…      0
2898  I judged this too quickly. This is not bad at all      0
2899                                   give it a chance      0
2900  does this mean that the rabbids are around 5 f…   2772
2901  It looks so stupid, but it might be a nice gam…      0
2902                 44 views\n256 likes\nYouTube stop!     24
2903            43 Views, 186 likes.\nooookkaaaay then.      0
2904                            "43 views"\n>178 likes      0
2905                                               hola      0
2906                                         Can’t wait      0
2907   Why Nintendo why did you have to do this to us!?      3
2908                      So, Nintendo is on crack now?      7
2910           LOOK AT THIS AND TELL ME THERE IS A GOD.    728
2911                               it's officially LIT      3
2912                        Better have a Rayman cameo.      0
2913                  i just can't like those things…      4
2915                                    i'm so confused     13
2916  How would you grade Nintendo's E3, good people…    244
2917                    Mario is confused as I am 00:19     26
2919                                       69th comment      0
2920                                            Woooow      0
2921                      *_Ohhh yeah!!! LETS-A-GO!!!_*    109
2923                      *gahhhhh peach looks ugly!!!*      0
2924  Omg xD Nintendo You are the best at making gam…     49
2925                                              ?.???      0
2927                                         Really…?      0
2928  I just want to know how they can make a crosso…   3119
2929                       New Battlefield looks great.   1017
2930  Damn cant wait for Rabbid Peach porn that be p…      1
2931                          wheres the waluigi rabbit     27
2932                                  Day one for me!!!      0
2933              43 views, yet 50 likes, HAKER ALERT!      0
2934                                                 24      0
2935                        god has answered my prayers      0
2937                                           …why??      7
2938                                            lol wut      0
2939  Seriously I think I'm gonna pass about getting…      3
2940                         (insert nice comment here)      0
2942                                          cool game     11
2943                                             first!      0
2944                                              what.      0
2945                                              First      0
2946                                           aaaaaaaa      0
2947             Y'all say first but I got a blue shell    455
2948                                                1st      0
2949                                               Cool      1
2951                                            i loved      1
```

```
2952                                                first       0
2953                                       where is smash       0
2955                                                 why?       2
2957                                                  lol       0

                      date     timestamp
2891  2017-06-13T21:34:40Z  1.497390e+09
2892  2017-06-13T21:34:29Z  1.497390e+09
2894  2017-06-13T21:34:25Z  1.497390e+09
2895  2017-06-13T21:34:19Z  1.497390e+09
2897  2017-06-13T21:34:16Z  1.497390e+09
2898  2017-06-13T21:34:16Z  1.497390e+09
2899  2017-06-13T21:34:10Z  1.497390e+09
2900  2017-06-13T21:34:08Z  1.497390e+09
2901  2017-06-13T21:34:02Z  1.497390e+09
2902  2017-06-13T21:33:58Z  1.497390e+09
2903  2017-06-13T21:33:53Z  1.497390e+09
2904  2017-06-13T21:33:50Z  1.497390e+09
2905  2017-06-13T21:33:21Z  1.497390e+09
2906  2017-06-13T21:33:21Z  1.497390e+09
2907  2017-06-13T21:33:12Z  1.497390e+09
2908  2017-06-13T21:32:56Z  1.497390e+09
2910  2017-06-13T21:32:44Z  1.497390e+09
2911  2017-06-13T21:32:43Z  1.497390e+09
2912  2017-06-13T21:32:42Z  1.497390e+09
2913  2017-06-13T21:32:37Z  1.497390e+09
2915  2017-06-13T21:32:20Z  1.497390e+09
2916  2017-06-13T21:32:19Z  1.497390e+09
2917  2017-06-13T21:33:12Z  1.497390e+09
2919  2017-06-13T21:32:16Z  1.497390e+09
2920  2017-06-13T21:32:06Z  1.497390e+09
2921  2017-06-13T21:32:05Z  1.497390e+09
2923  2017-06-13T21:32:01Z  1.497390e+09
2924  2017-06-13T21:31:59Z  1.497390e+09
2925  2017-06-13T21:31:50Z  1.497390e+09
2927  2017-06-13T21:31:45Z  1.497390e+09
2928  2017-06-13T21:31:44Z  1.497390e+09
2929  2017-06-13T21:31:36Z  1.497389e+09
2930  2017-06-13T21:31:27Z  1.497389e+09
2931  2017-06-13T21:31:24Z  1.497389e+09
2932  2017-06-13T21:31:21Z  1.497389e+09
2933  2017-06-13T21:31:21Z  1.497389e+09
2934  2017-06-13T21:31:18Z  1.497389e+09
2935  2017-06-13T21:31:16Z  1.497389e+09
2937  2017-06-13T21:31:12Z  1.497389e+09
2938  2017-06-13T21:31:07Z  1.497389e+09
2939  2017-06-13T21:31:01Z  1.497389e+09
```

```
2940  2017-06-13T21:31:01Z  1.497389e+09
2942  2017-06-13T21:30:50Z  1.497389e+09
2943  2017-06-13T21:30:49Z  1.497389e+09
2944  2017-06-13T21:30:49Z  1.497389e+09
2945  2017-06-13T21:30:48Z  1.497389e+09
2946  2017-06-13T21:30:46Z  1.497389e+09
2947  2017-06-13T21:30:45Z  1.497389e+09
2948  2017-06-13T21:30:38Z  1.497389e+09
2949  2017-06-13T21:30:38Z  1.497389e+09
2951  2017-06-13T21:30:36Z  1.497389e+09
2952  2017-06-13T21:30:35Z  1.497389e+09
2953  2017-06-13T21:30:31Z  1.497389e+09
2955  2017-06-13T21:30:18Z  1.497389e+09
2957  2017-06-13T21:30:15Z  1.497389e+09  ]
```

5 Project Outcome (10 + 10 marks)

5.1 Overview of Results

The analysis gives a way to measure what is the overall sentiment by the audience. In Objective one, we focus on the new video game announcement and briefly compare the results. Using the WordCloud library, we detect some interesting words that could give some clues about what can be done to achieve a higher appreciation among the public. In Objective three, we compare the results obtained with its predecessor, RKB. This comparison is helpful to see whether or not the new announcement trailer is doing better in terms of appreciation by the public.

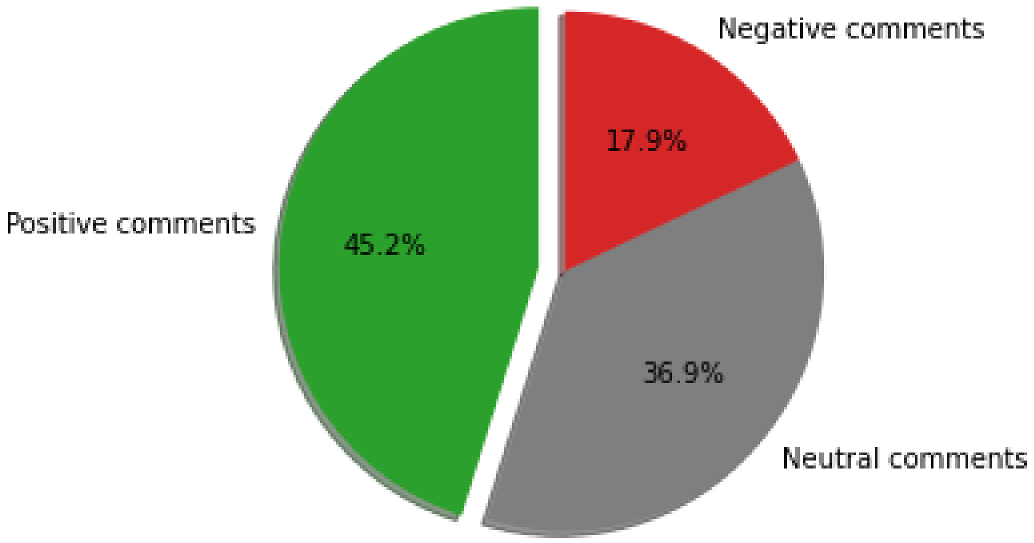
5.2 Objective 1: Analyse the overall sentiment distribution of Mario+Rabbids: Sparks of Hope's announcement trailer

5.2.1 Explanation of Results

The first result I will consider is the one obtained with the DataFrame SA_SOH. Here we don't consider the weight determined by the like counter under each comment. The result shows an overall 45.2% appreciation of the game. This isn't high compared to the other two sentiment classes. Still, considering this is an announcement trailer and many questions and concerns about the actual game remain, we can say that it is a good starting point.

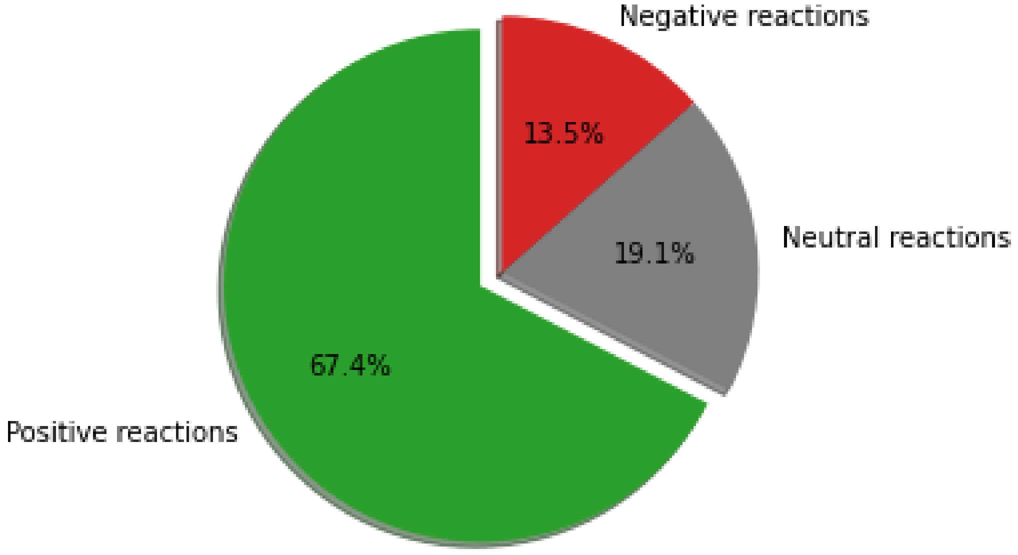
[37]:
```python
def visualize_sentiment_analysis(analysis_result, labels):
    sizes = [analysis_result["count"]["pos"], analysis_result["count"]["neu"],␣
↪analysis_result["count"]["neg"]]
    explode = (0.1, 0, 0)
    colors = [mcolors.TABLEAU_COLORS["tab:green"], mcolors.TABLEAU_COLORS["tab:
↪gray"], mcolors.TABLEAU_COLORS["tab:red"]]
    fig1, ax1 = plt.subplots()
    ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',
            shadow=True, startangle=90, colors=colors)
    ax1.axis('equal')
    plt.show()
```

```
visualize_sentiment_analysis(SA_SOH, ['Positive comments', 'Neutral comments',␣
↪'Negative comments'])
```



Let's now consider the results obtained with the DataFrame SAC_SOH. In this case, each comment is weighted using its number of likes. 67.4% had a positive feeling about the game, and the number of positive and negative feedbacks decreased in percentage. That means the negative and neutral comments didn't find much approval by the public.

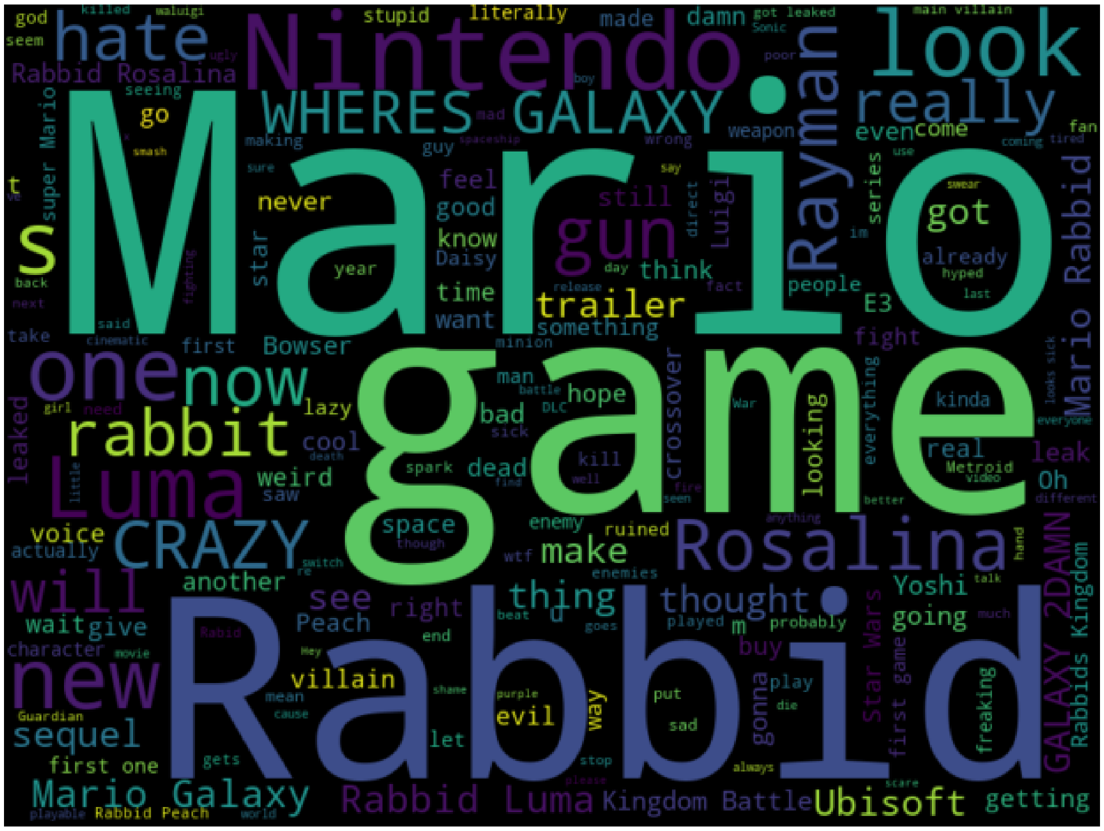
[38]:
```
visualize_sentiment_analysis(SAC_SOH, ['Positive reactions', 'Neutral␣
↪reactions', 'Negative reactions'])
```

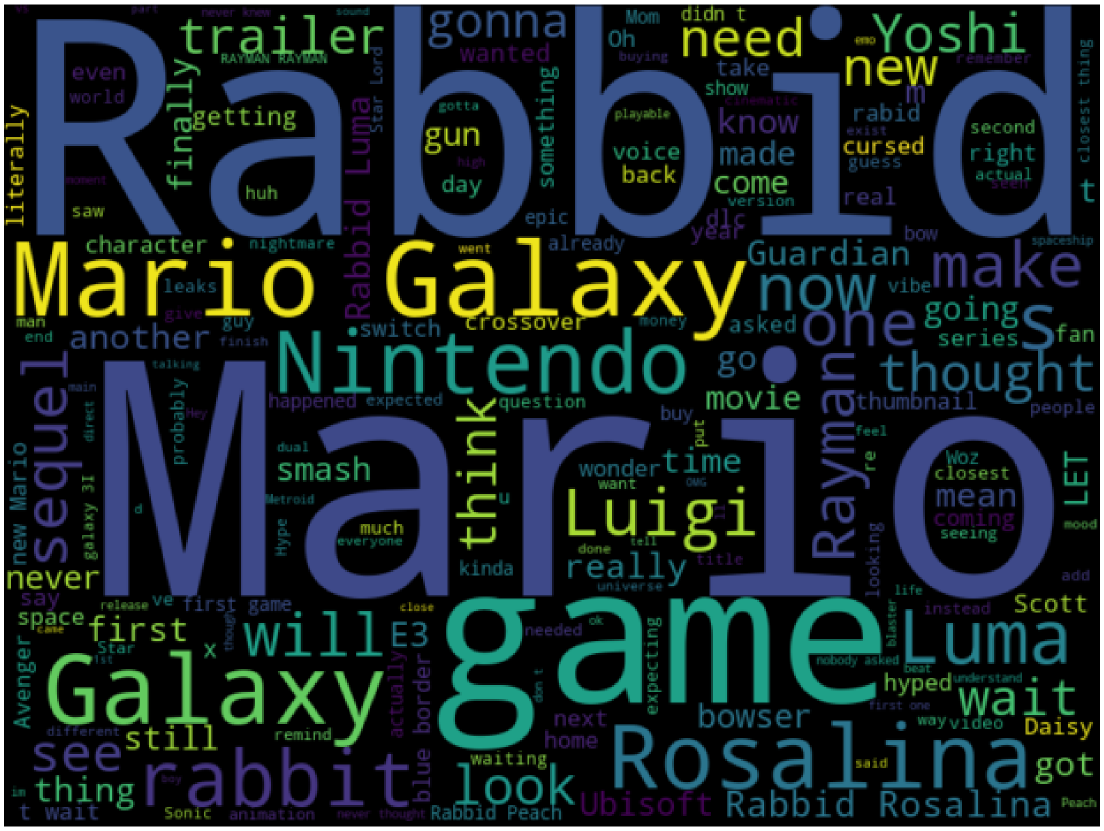


We have 37%, 19.1% and 17.9%, 13.5% of neutral and negative comments with the two data frames, respectively. Even if most of them are not approved by the majority of the audience, it is still essential to know the reason for their reluctance.

If we plot the word cloud for the negative and neutral comments, we see some interesting words. Many comments mentioned the word "crossover", which indicates that most negative comments may consider the crossover between Nintendo and Ubisoft characters a weak point of the game. Another interesting word is "Rayman", which refers to the famous Ubisoft character. Fans were expecting to see Rayman in the trailer, but this didn't happen. The "Rayman" word is interesting because it is mentioned in both neutral and negative word clouds. This suggests that adding Rayman to the game could change the mind of neutral but also reluctant players.

[39]:
```
def visualize_word_clouds(comments_dataset, key):
    plt.figure(figsize = (10,10), dpi=80)
    plt.imshow(WordCloud(width=800, height=600).
 ↪generate(comments_dataset["text"][key]), interpolation='bilinear')
    plt.axis("off")

visualize_word_clouds(SA_SOH, "neg")
visualize_word_clouds(SA_SOH, "neu")
```


Mario
game
Rabbid
Nintendo
look
hate
WHERES GALAXY
really
Rayman
gun
s
one
now
rabbit
Luma
CRAZY
Rosalina
will
new
thought
thing
trailer
sequel
Mario Galaxy
Rabbid Luma
Kingdom Battle
Ubisoft
getting
Rabbid Rosalina
got
Mario Rabbid
GALAXY 2DAMN
Rabbids Kingdom
Star Wars
first game
Yoshi
Peach
Bowser
villain
evil
make
space
voice
another
see
right
going
buy
play
gonna
freaking
leak
real
Oh
kinda
hope
looking
crossover
everything
dead
bad
cool
lazy
weird
saw
feel
good
know
Daisy
time
want
something
think
still
Luigi
series
already
people
E3
fight
never
go
super Mario
star
stupid
literally
made
damn
god
seem
killed
waluigi
main villain
got leaked
Sonic
fan
come
even
weapon
first
take
leaked
actually
wait
give
character
first one
Rabbid Peach
playable
world
gets
stop
ruined
enemy
kill
man
DLC
year
sad
put
way
let
Metroid
different
seeing
making
guy
wrong
smash
hyped
Guardian
mean
cause
shame
purple
die
probably
played
end
beat
goes
wtf
enemies
minion
spark
sick
War
girl
need
little
death
switch
though
Hey
movie
talk
much
hand
better
seen
fire
find
well
video
anything
always
please
scare
fact
release
day
im
use
swear
last
tired
coming
poor
boy
ugly
mad
spaceship
sure
say
direct
back
next
said
fighting
cinematic
battle
looks sick
everyone



5.3 Objective 2: Analyse the sentiment distribution within the first 24 hours from the announcement of Mario+Rabbids: Sparks of Hope

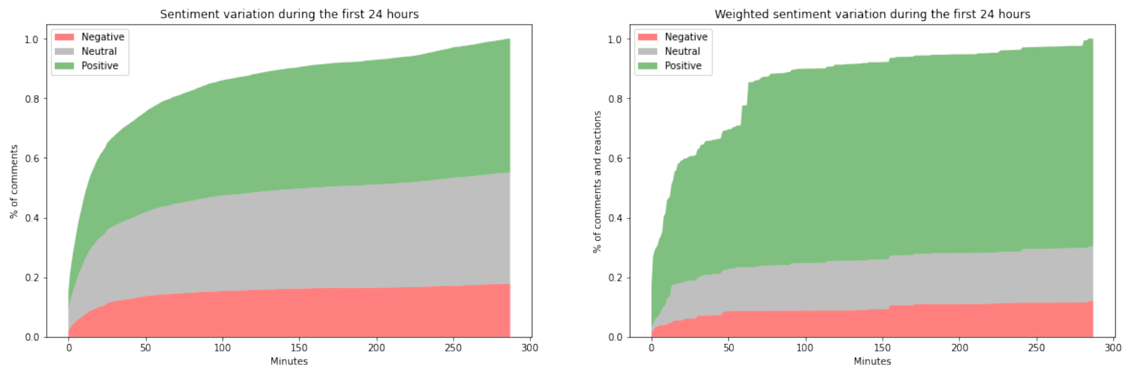
5.3.1 Explanation of Results

The results obtained with the samples stored in SAMPLED_SOH_RES confirm that the sentiment distribution aligns with objective one's results since the first hour after the announcement. It also shows that throughout the 24 hours, the distribution doesn't change much; it is stable among the time frame. As we expected, plotting the weighted sentiment variation shows more robust evidence about the successfulness of the announcement. It's important to note that we don't have any information about the date and time for a specific like. This means that in the 24 hours, we are considering likes that could be arrived after. Given that most of the interactions happened in the first few days after the announcement, this method gives a rough estimate about the appreciation for each comment anyway.

5.3.2 Visualisation

```
[40]: def visualize_sentiment_variation(focus, samples, xlabel, ylabel, title,␣
       ↪consider_likes = False):
          sentiment_neg = np.cumsum([comments_sentiment_analysis(sample,␣
       ↪consider_likes)["count"]["neg"] for sample in samples])
```

```
    sentiment_neu = np.cumsum([comments_sentiment_analysis(sample,␣
↪consider_likes)["count"]["neu"] for sample in samples])
    sentiment_pos = np.cumsum([comments_sentiment_analysis(sample,␣
↪consider_likes)["count"]["pos"] for sample in samples])

    total_comments = sentiment_neg[-1] + sentiment_neu[-1] + sentiment_pos[-1]
    plt.subplot(1, 2, focus)

    plt.stackplot(
        np.arange(0, len(samples)),
        sentiment_neg / total_comments,
        sentiment_neu / total_comments,
        sentiment_pos / total_comments,
        colors=['red', 'gray', 'green'],
        alpha=0.5,
        labels=["Negative", "Neutral", "Positive"]
    )
    plt.xlabel(xlabel)
    plt.ylabel(ylabel)
    plt.title(title)
    plt.legend(loc="upper left")

plt.figure(figsize=(20,6))

visualize_sentiment_variation(1, SAMPLED_SOH_RES, "Minutes", "% of comments",␣
↪"Sentiment variation during the first 24 hours")
visualize_sentiment_variation(2, SAMPLED_SOH_RES, "Minutes", "% of comments and␣
↪reactions", "Weighted sentiment variation during the first 24 hours", True)

plt.show()
```



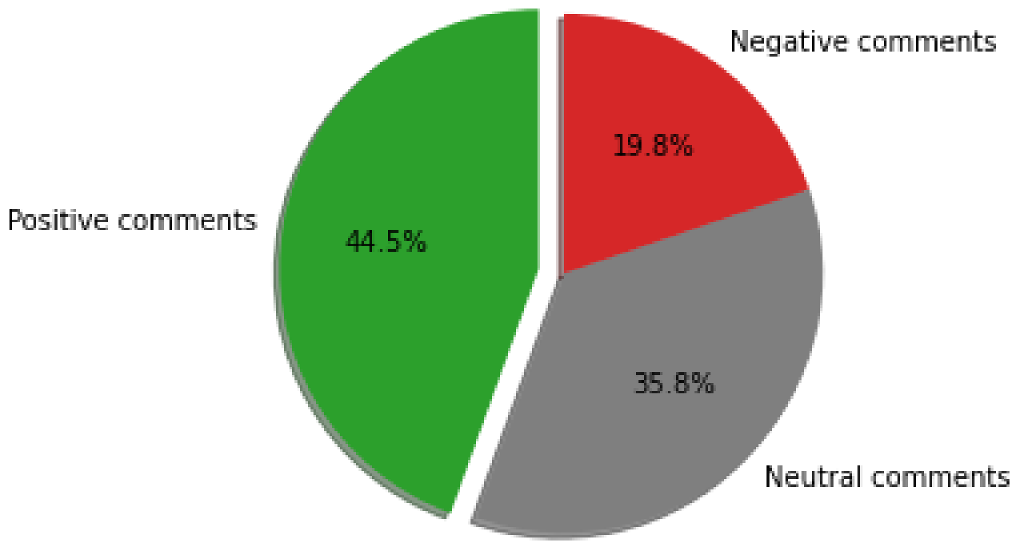
5.4 Objective 3

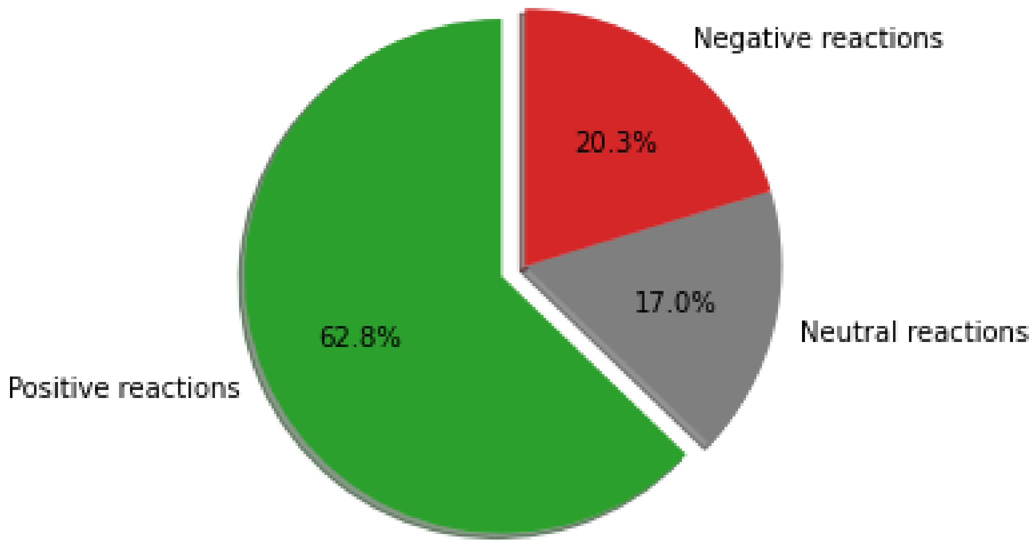
5.4.1 Explanation of Results

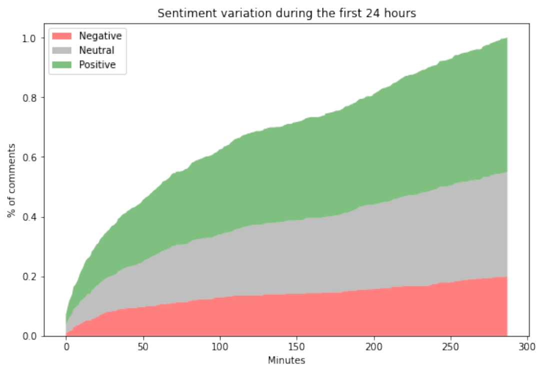
Analysing the "Mario+Rabbids: Kingdom Battle" announcement, we see some similarities with its sequel. Interestingly, even if, at that time, fans didn't know exactly what to expect from the idea of a crossover Ubisoft-Nintendo, in both cases, the sentiment distribution is almost identical to the one seen in objective one. Looking at the sentiment variation, we notice that the positive sentiment converged slower than its sequel in the left graph. This means that the RKB announcement trailer received a lower number of comments per minute. This can be partially explained by the fact that the franchise didn't exist, so there wasn't a strong community waiting for it.

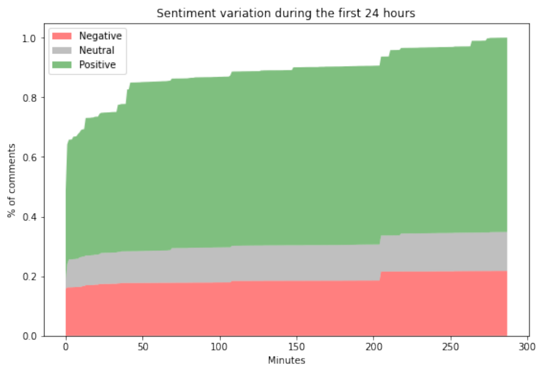
5.4.2 Visualisation

[41]:
```
visualize_sentiment_analysis(SA_RKB, ['Positive comments', 'Neutral comments',␣
 ↪'Negative comments'])
visualize_sentiment_analysis(SAC_RKB, ['Positive reactions', 'Neutral␣
 ↪reactions', 'Negative reactions'])

plt.figure(figsize=(20,6))

visualize_sentiment_variation(1, SAMPLED_RKB_RES, "Minutes", "% of comments",␣
 ↪"Sentiment variation during the first 24 hours")
visualize_sentiment_variation(2, SAMPLED_RKB_RES, "Minutes", "% of comments",␣
 ↪"Sentiment variation during the first 24 hours", True)

plt.show()
```









6 Conclusion (5 marks)

6.0.1 Acheivements

The results show that Mario+Rabbids: Sparks of Hope's announcement trailer is a success in terms of appreciation. The comparison with its predecessor revealed greater attention by the public in the first 24 hours and showed that the sentiment distribution obtained aligns with the previous results. This confirms that the announcement is a success, but still, there are some improvements Ubisoft can make to satisfy a larger portion of the audience.

6.0.2 Limitations

Unfortunately, the analysis has multiple limitations. Firstly, the youtube API doesn't grant access to all the relevant information about a video. For example, having the exact publication time could *have given more precise results when plotting the data. Another interesting information is the time at which a like is added to a comment. This could have been useful to select only the likes added in the first 24 hours from the announcement. Nonetheless, huge limitations are imposed by the sentiment analyser. Even if vaderSentiment[3] is much more precise than most sentiment analysers available for Python, many sentences are still challenging to evaluate correctly. For example, "this game is sick!" would be assessed as a negative sentence while in this context should be considered positive. A way better sentiment analysis can be carried out using SentiStrenght[4]. Unfortunately, the fact that SentiStrenght[4] is only available on windows and requires java to work properly makes it inconvenient for a cross-platform notebook environment.*

6.0.3 Future Work

In future works, we would like to strengthen our pipeline using SentiStrenght[4] to achieve a higher fidelity of the sentiment analyser. Introducing a web scraper could give us new relevant information about a video, such as the number of dislikes for each comment, thus enhancing the dataset and the possible outcome. In conclusion, there is still some room for improvements in performance and precision for what concerns the English filtering. We hypothesise that by using Google Translate, we could get much better results.

7 Bibliography

1. WordNet. Princeton University. Princeton University "About WordNet." WordNet. 2010.
2. FastText. Joulin, Armand and Grave, Edouard and Bojanowski, Piotr and Mikolov, Tomas. Bag of Tricks for Efficient Text Classification. arXiv preprint arXiv:1607.01759. 2016.
3. VaderSentiment. Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014
4. SentiStrength. University of Wolverhampton. Project Website