wiki-research

March 18, 2024

Lang Links (links btwn pages): https://www.mediawiki.org/wiki/API:Langlinks https://en.wikipedia.org/w/api.php?action=query&titles=Albert%20Einstein&prop=langlinks&format=json&llli

Rest Api: https://wikimedia.org/api/rest_v1/

Page views: https://wikitech.wikimedia.org/wiki/Analytics/AQS/Pageviews

By Article: https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/en.wikipedia/all-access/all-agents/Albert_Einstein/monthly/20220101/20220131

By Page: https://wikimedia.org/api/rest_v1/metrics/edits/per-page/en.wikipedia/Albert%20Einstein/user/daily/20220101/20220131

Metrics that I am using (wiki stat) https://stats.wikimedia.org/#/it.wikipedia.org/reading/total-page-views/normal|table|2-year|agent~user|monthly

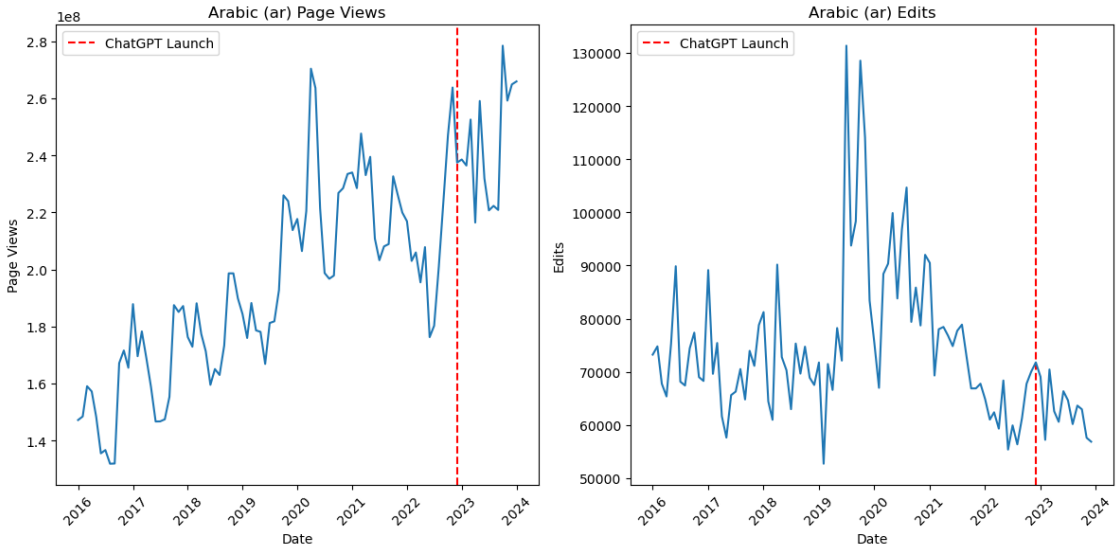
Get edits/contributions https://wikimedia.org/api/rest_v1/metrics/edits/aggregate/en.wikipedia/user/content/d

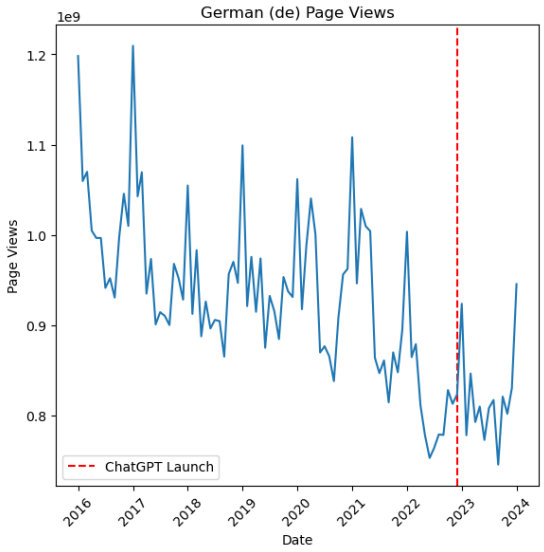
Get page views /metrics/pageviews/per-article/{project}/{access}/{agent}/{article}/{granularity}/{start}/{end} https://wikimedia.org/api/rest_v1/metrics/pageviews/aggregate/en.wikipedia.org/all-access/user/hourly/2015100100/2015100123

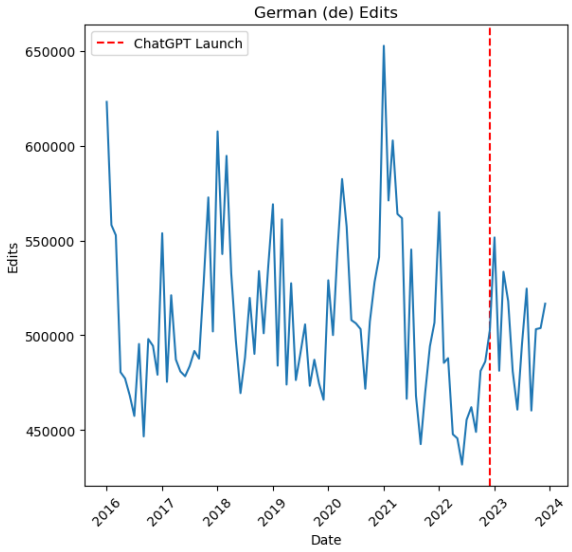
[2]:
```
import requests
import matplotlib.pyplot as plt
from datetime import datetime
import pandas as pd

class WikiDataVisualizer:
    def __init__(self):
        self.codes = self.get_codes()
        self.start_date = '2016010100'
        self.end_date = '2024010100'

    def get_codes(self):
        codes = pd.read_csv("language_codes.csv")
        codes = codes.set_index("country")
        return codes

    def get(self, url):
        headers = {'User-Agent': 'CoolBot/0.0 (https://example.org/coolbot/;␣
 ↪coolbot@example.org) generic-library/0.0 Cool Bot (jesgarbage@gmail.com)'}
```

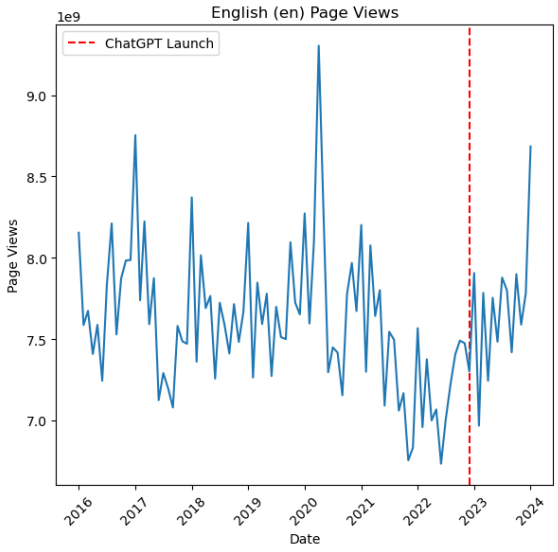
```
        response = requests.get(url, headers=headers)
        if response.status_code == 200:
            return response.json()
        else:
            return None

    def fetch_pageviews(self, country):
        url = f"https://wikimedia.org/api/rest_v1/metrics/pageviews/aggregate/
↪{country}.wikipedia.org/all-access/user/monthly/{self.start_date}/{self.
↪end_date}"
        return self.get(url)

    def fetch_edits(self, country):
        url = f"https://wikimedia.org/api/rest_v1/metrics/edits/aggregate/
↪{country}.wikipedia/user/content/monthly/{self.start_date}/{self.end_date}"
        return self.get(url)

    def add_launch(self, axs):
        chatgpt_launch_date = datetime(2022, 11, 30)
        for ax in axs:
            ax.axvline(chatgpt_launch_date, color='red', linestyle='--',␣
↪label='ChatGPT Launch')
            ax.legend()

    def plot(self, views, edits, country, code):
        fig, axs = plt.subplots(1, 2, figsize=(12, 6))
        self.add_launch(axs)

        if views:
            timestamps_views = [item['timestamp'] for item in views['items']]
            views_data = [item['views'] for item in views['items']]
            dates_views = [datetime.strptime(timestamp, '%Y%m%d%H') for␣
↪timestamp in timestamps_views]

            axs[0].plot(dates_views, views_data)
            axs[0].set_title(f"{country} ({code}) Page Views")
            axs[0].set_xlabel('Date')
            axs[0].set_ylabel('Page Views')
            axs[0].tick_params(axis='x', rotation=45)

        if edits:
            timestamps_edits = [item['timestamp'] for item in␣
↪edits['items'][0]["results"]]
            edits_data = [item['edits'] for item in␣
↪edits['items'][0]["results"]]
```

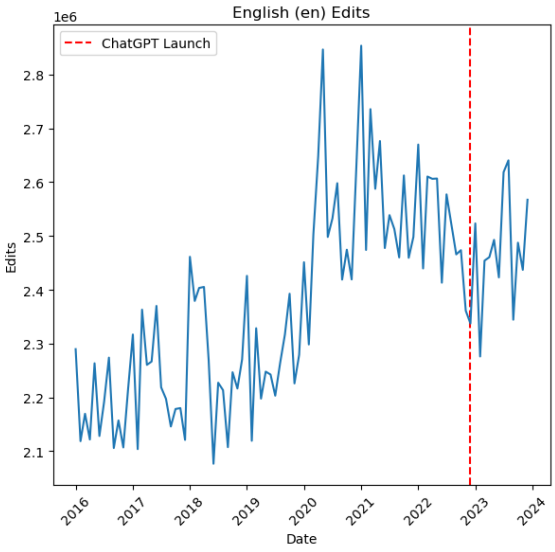
```
            dates_edits = [datetime.fromisoformat(timestamp.replace('Z', '+00:
↪00')) for timestamp in timestamps_edits]

            axs[1].plot(dates_edits, edits_data)
            axs[1].set_title(f"{country} ({code}) Edits")
            axs[1].set_xlabel('Date')
            axs[1].set_ylabel('Edits')
            axs[1].tick_params(axis='x', rotation=45)

        plt.tight_layout()
        plt.show()

    def visualize_data(self):
        for country, row in self.codes.iterrows():
            code = row["code"]
            views = self.fetch_pageviews(code)
            if views is None:
                continue
            views_data = [item['views'] for item in views['items']]
            if any(view < 10**8 for view in views_data):
                continue
            edits = self.fetch_edits(code)
            self.plot(views, edits, country,code)


visualizer = WikiDataVisualizer()
visualizer.visualize_data()
```

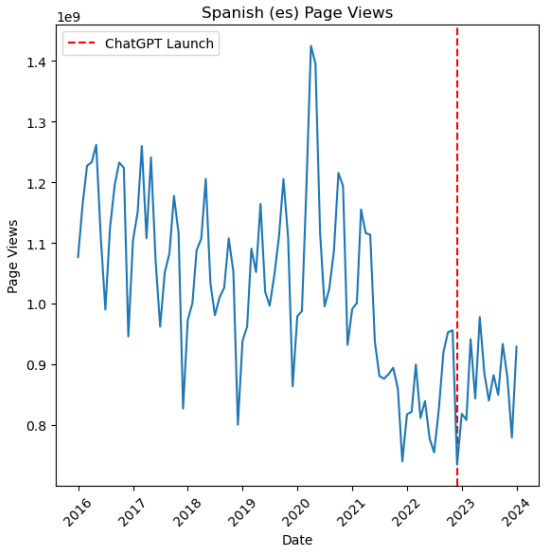


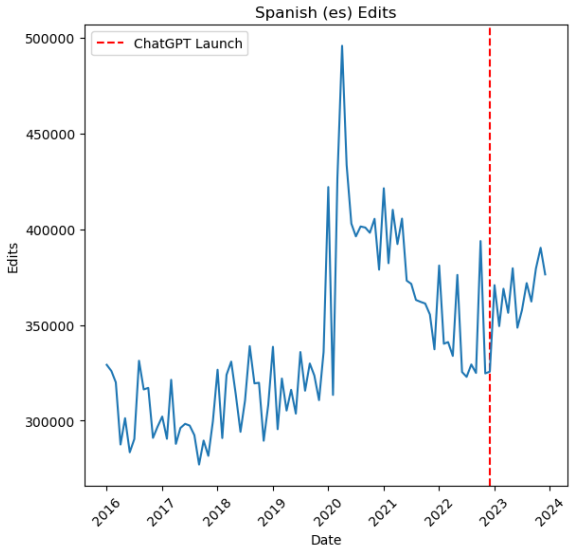

German (de) Page Views
1e9
Page Views
Date
ChatGPT Launch

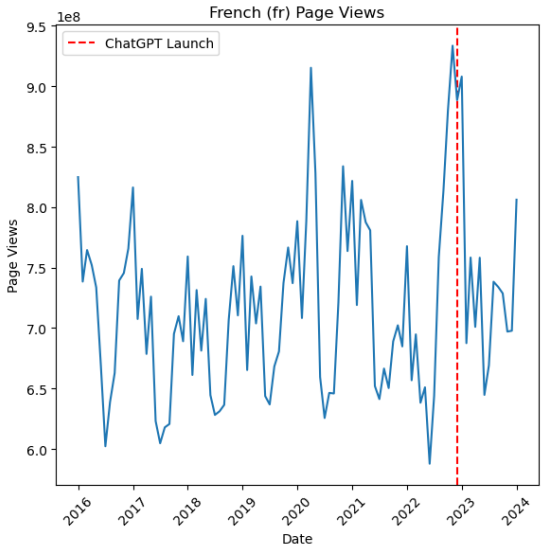

German (de) Edits
Edits
Date
ChatGPT Launch

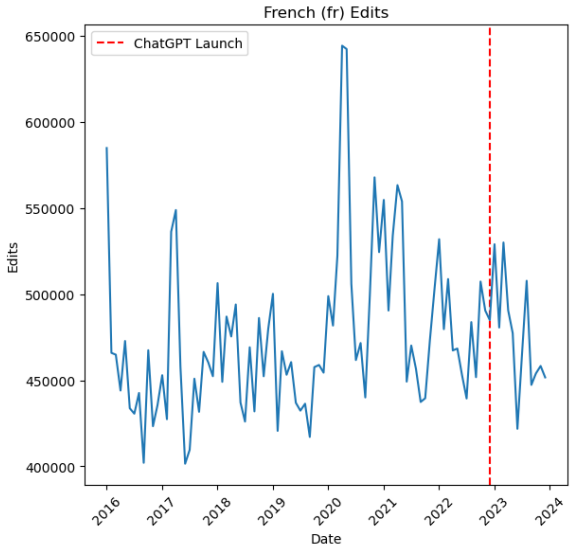

English (en) Page Views
1e9
Page Views
Date
ChatGPT Launch

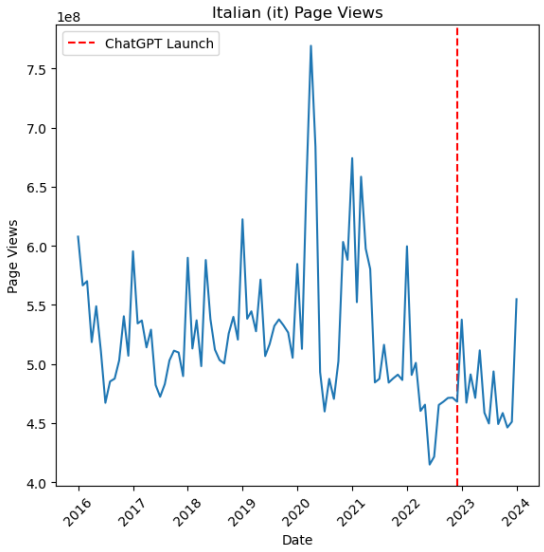

English (en) Edits
1e6
Edits
Date
ChatGPT Launch

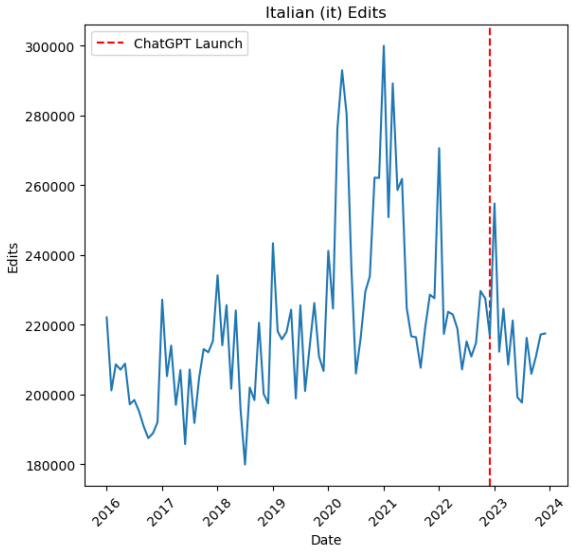

Spanish (es) Page Views
1e9
ChatGPT Launch
Page Views
Date

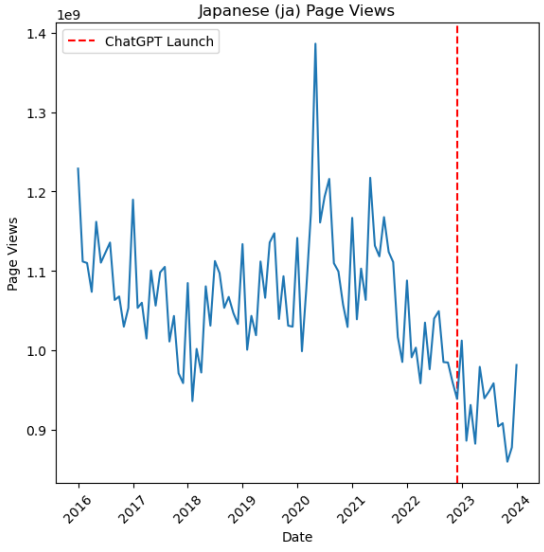

Spanish (es) Edits
ChatGPT Launch
Edits
Date

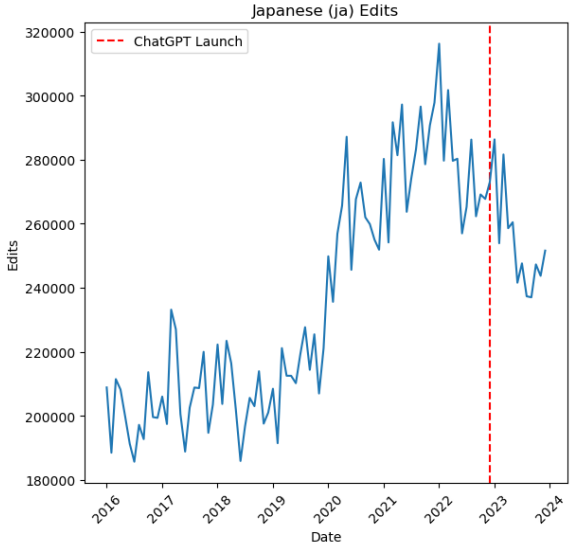

French (fr) Page Views
1e8
ChatGPT Launch
Page Views
Date

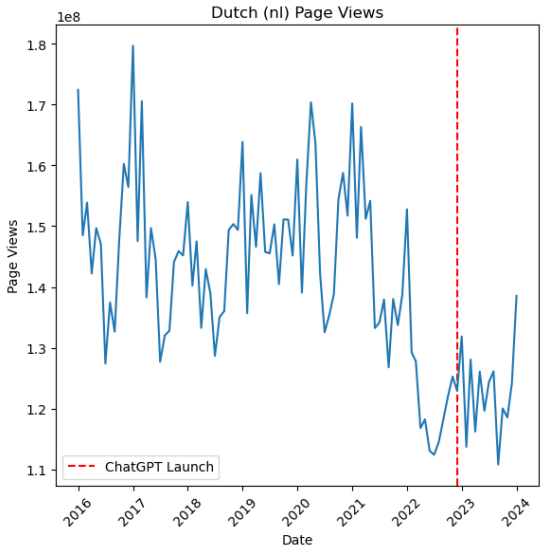

French (fr) Edits
ChatGPT Launch
Edits
Date

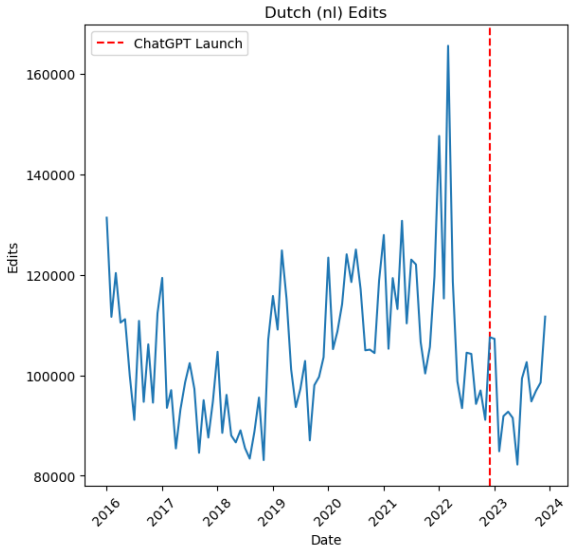

Italian (it) Page Views
1e8
ChatGPT Launch
Page Views
Date

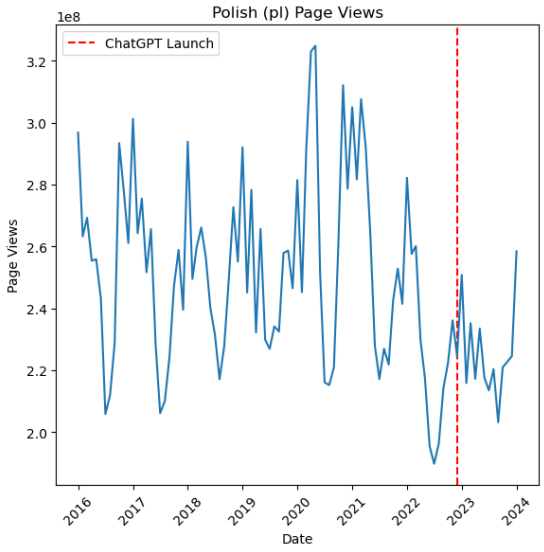

Italian (it) Edits
ChatGPT Launch
Edits
Date

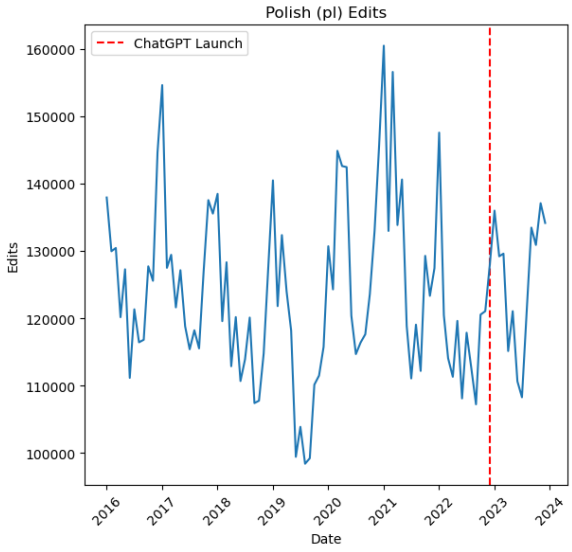

Japanese (ja) Page Views
1e9
ChatGPT Launch
Page Views
Date

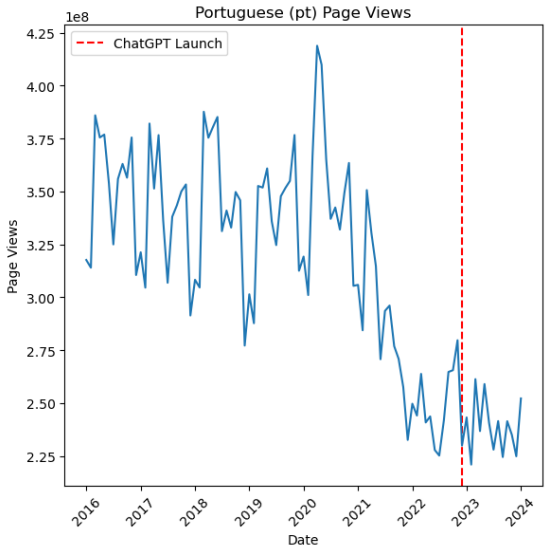

Japanese (ja) Edits
ChatGPT Launch
Edits
Date

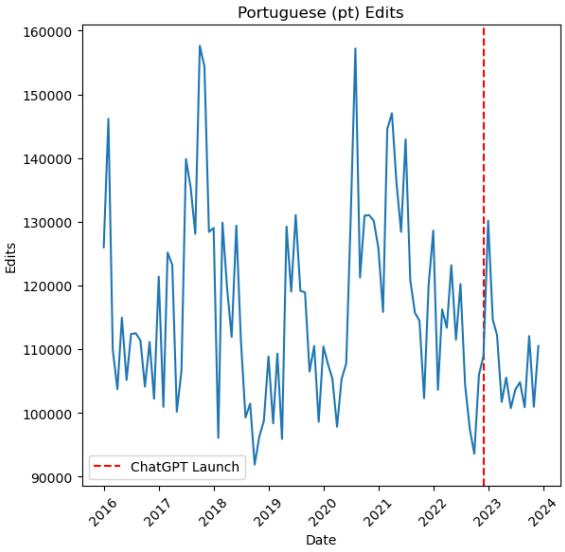

Dutch (nl) Page Views
1e8
Page Views
Date
ChatGPT Launch

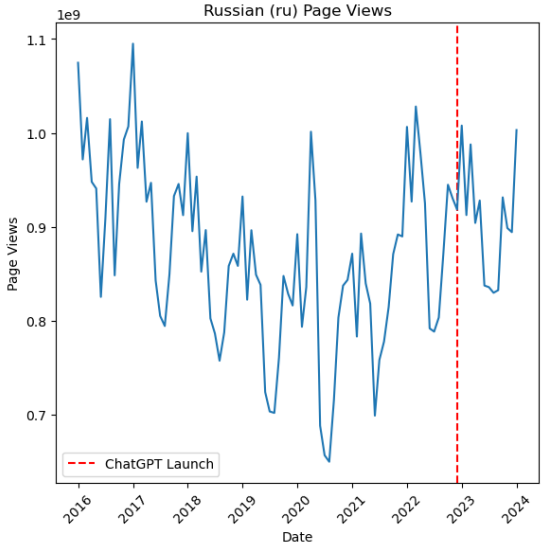

Dutch (nl) Edits
Edits
Date
ChatGPT Launch

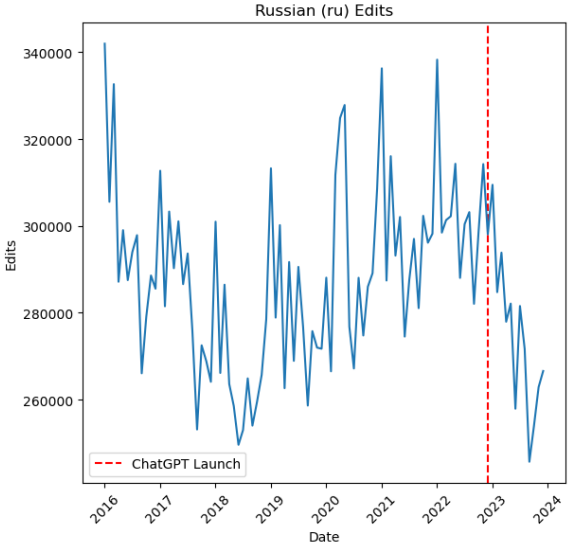

Polish (pl) Page Views
1e8
Page Views
Date
ChatGPT Launch

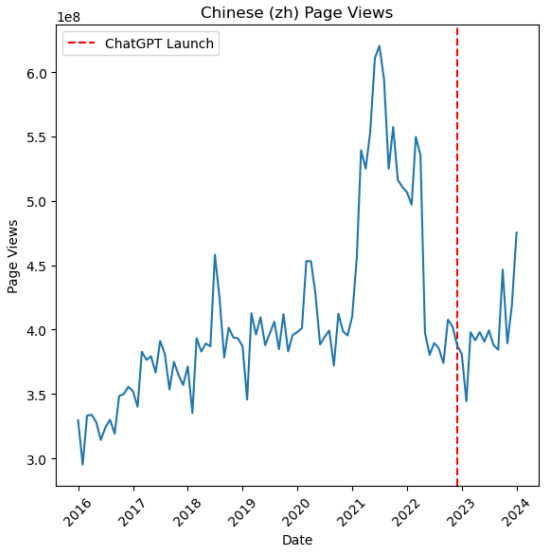

Polish (pl) Edits
Edits
Date
ChatGPT Launch

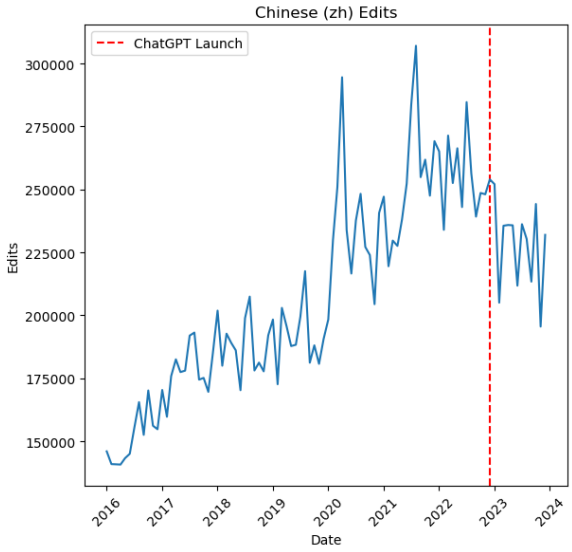

Portuguese (pt) Page Views
1e8
ChatGPT Launch
Page Views
Date


Portuguese (pt) Edits
ChatGPT Launch
Edits
Date


Russian (ru) Page Views
1e9
ChatGPT Launch
Page Views
Date


Russian (ru) Edits
ChatGPT Launch
Edits
Date


Chinese (zh) Page Views
1e8
ChatGPT Launch
Page Views
Date
6.0
5.5
5.0
4.5
4.0
3.5
3.0
2016
2017
2018
2019
2020
2021
2022
2023
2024


Chinese (zh) Edits
ChatGPT Launch
Edits
Date
300000
275000
250000
225000
200000
175000
150000
2016
2017
2018
2019
2020
2021
2022
2023
2024

[ ]: