Newspaper Coverage of COVID-19 In the Middle East

Final Project Report

Department of Arabic and Translation Studies, American University of Sharjah

ARA 250: Introduction to Arabic Digital Humanities – Spring 2024

Khalifa Salem Almatrooshi – @00090847

Dr. Mai Zaki

May 20, 2024

Introduction

This project explores the Arabic Newspaper COVID-19 Corpus (AraNPCC), which includes newspaper articles from 12 Arab countries spanning 2019 to 2021. The AraNPCC corpus, a crucial resource for Arabic language research, contains over 2 billion words and 7.2 million texts with metadata, collected from various Arabic newspapers (Al-Thubaity et al., 2022).

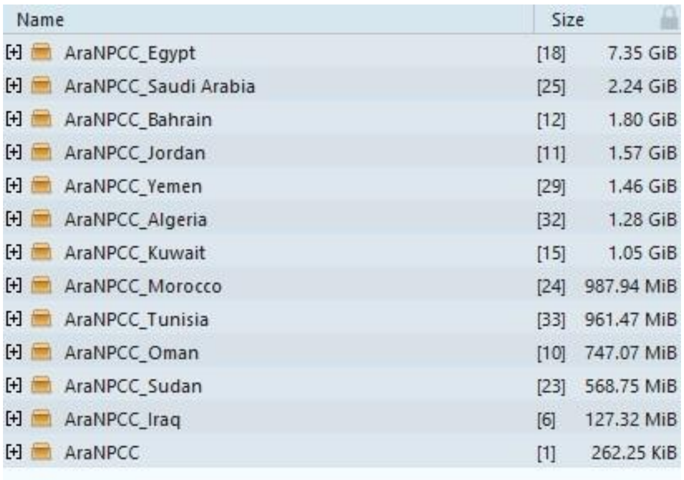
The primary research question is: How has newspaper coverage of COVID-19 in the Middle East evolved over time? This study aims to uncover patterns and trends in pandemic reporting across different Arab countries, providing insights into the media's role in shaping public discourse during the crisis.

The analysis combines textual and temporal methodologies. Keyword frequency analysis identifies prominent terms, while collocation analysis reveals contextual word relationships. Temporal analysis observes how key term frequencies vary over time, identifying spikes corresponding with major pandemic events. The study also includes superimposing COVID-19 statistics to visualize the media's impact. Digital tools like Python and Excel were used, utilizing Pandas for data manipulation, NLTK for text processing, PyArabic for Arabic linguistics, and Excel's Power Query and PivotTables for data transformation and visualization.

This research contributes to digital humanities by examining the media landscape during a critical period. It offers nuanced insights into information dissemination and public perception during COVID-19, highlighting the importance of linguistic and temporal analysis in large datasets. The findings are valuable for researchers, policymakers, and media analysts interested in the intersection of media, language, and public health.

Dataset Description

The AraNPCC corpus can be downloaded for free from https://archive.org/details/AraNPCC, as linked in their report, the files are categorized by country, newspaper, and year. I used Jdownloader2 to web crawl the site, leaving me with 238 csv files with a total size of 20.06 GB.



Name		Size
AraNPCC_Egypt	[18]	7.35 GiB
AraNPCC_Saudi Arabia	[25]	2.24 GiB
AraNPCC_Bahrain	[12]	1.80 GiB
AraNPCC_Jordan	[11]	1.57 GiB
AraNPCC_Yemen	[29]	1.46 GiB
AraNPCC_Algeria	[32]	1.28 GiB
AraNPCC_Kuwait	[15]	1.05 GiB
AraNPCC_Morocco	[24]	987.94 MiB
AraNPCC_Tunisia	[33]	961.47 MiB
AraNPCC_Oman	[10]	747.07 MiB
AraNPCC_Sudan	[23]	568.75 MiB
AraNPCC_Iraq	[6]	127.32 MiB
AraNPCC	[1]	262.25 KiB

Figure 1: Web crawl results.

Country	Newspapers	Texts	Tokens
Algeria	11	439,204	133,040,389
Bahrain	4	571,162	201,409,392
Egypt	6	2,926,693	747,884,209
Iraq	4	48,178	12,879,456
Jordan	5	538,461	161,970,053
Kuwait	8	368,574	107,936,207
Morocco	4	268,827	101,124,149
Oman	7	203,542	76,634,312
Saudi Arabia	8	826,323	214,865,053
Sudan	11	178,461	58,500,490
Tunisia	10	509,427	92,404,722
Yemen	10	398,673	125,990,973
Total	88	7,277,525	2,034,639,405

Table 1: Number of newspapers, number of texts, and the total number of words for each Arab country in AraNPCC. Taken from AraNPCC article.

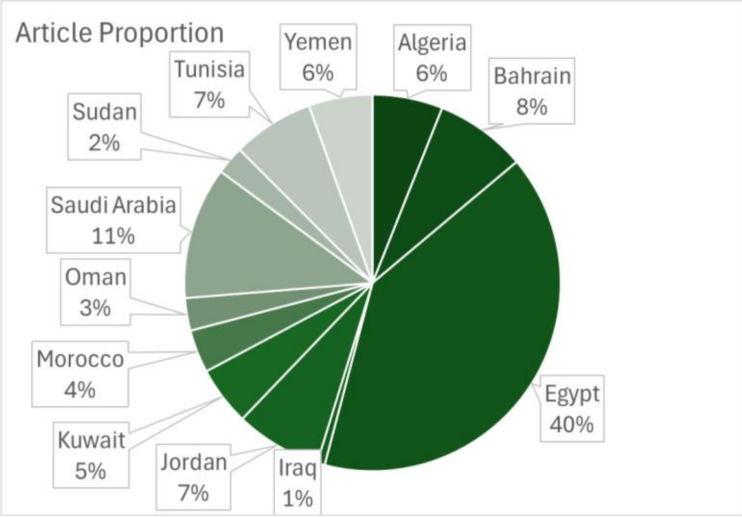


Figure 2: Article Proportion.

We can see from Figure 2 that it is an unbalanced dataset due to Egypt containing 40% of the total article count. Therefore, the distinction between countries in analysis is important to consider.

Figure 3: Token Count of Countries

To simplify data cleaning and visualization I focus on three countries: Egypt, Morocco, and Yemen. One reason is that the AraNPCC article has already done analysis between Saudi Arabia and Algeria. Another reason is that the selected countries represent distinct geopolitical regions and cultural contexts within the Arab world. Egypt being one of the most populous and influential countries means that it's response to COVID-19 would have significant regional implications. Morocco, along with Egypt, offers a unique view on how North African countries handles the pandemic. Yemen's situation is particularly complex due to its ongoing civil conflict, meaning that the pandemic's impact is intertwined with humanitarian crises; understanding these compounded challenges is crucial so as to learn from them.

The articles are not filtered to be COVID-19 related therefore I used Python to filter through each csv file for each country and save them in csv files categorised by country and newspaper. Resulting in 24 csv files with a total size of 1.7 GB. I used the search terms كورونا and كوفيد (Code C).

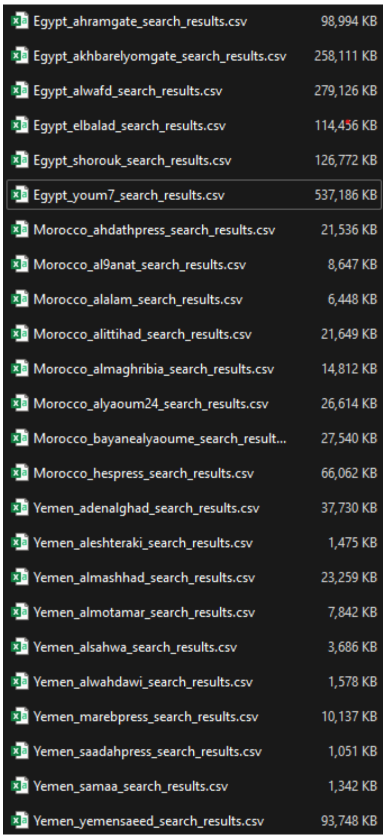


Figure 4: Filtered Articles by related to COVID-19

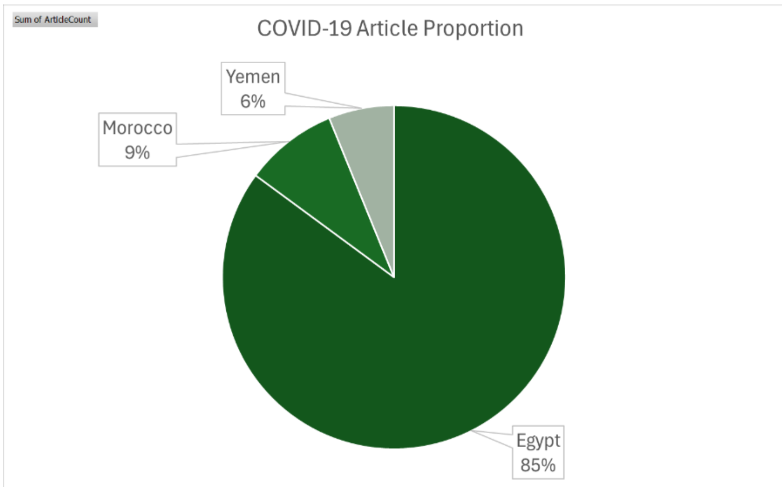


Figure 5: Article Proportion by country related to COVID-19

Row Labels	Sum of ArticleCount
Egypt	457,662
Morocco	47,057
Yemen	33,109
Grand Total	**537,828**

Table 2: Article Count by country related to COVID-19

Text	Full text of Article	**Category**	Categories defined by AraNPCC
Title	Title of Article	**Newspaper**	Newspaper name
URL	Web address to article	**Filename**	Origin file name
Date	Published date	**Term**	Search term found in article

Table 3: Csv file header names

Methodology

The primary goal of this analysis was to examine the evolution of newspaper coverage of COVID-19 in Egypt, Morocco, and Yemen. To achieve this, I employed a combination of textual and temporal analysis methods. The textual analysis involved keyword frequency and collocation analysis to identify significant terms and their contextual relationships. Temporal analysis was used to observe changes in term frequency over time, helping to identify trends and patterns in media coverage.

The analysis utilized various digital tools, including Python for data processing and Excel for data transformation and analysis. Key libraries used in Python included Pandas for data manipulation, NLTK for text processing, and Pyarabic for handling Arabic text (Code A). Excel's Power Query and PivotTables were instrumental in data transformation and visualization.

In all my code I use Arabic stop words sourced from a GitHub repository by Mohamed Taher Alrefaie (Alrefaie, 2016/2019). Stop words are used to eliminate words that are widely used for the purpose of optimizing the data processing, saving space and time. The following is step-by-step description of my workflow:

1. **Data Collection and Preprocessing**: Discussed in the previous section.
2. **Article Distribution Over Time:** Along with filtering the articles, I was able to visualise how the article count changes over time for every country (Code E

8 03 2020	58	Egypt	ahramgate
9 03 2020	68	Egypt	ahramgate
10 03 2020	102	Egypt	ahramgate
11 03 2020	78	Egypt	ahramgate
12 03 2020	54	Egypt	ahramgate
13 03 2020	46	Egypt	ahramgate
14 03 2020	118	Egypt	ahramgate
15 03 2020	168	Egypt	ahramgate
16 03 2020	130	Egypt	ahramgate
17 03 2020	164	Egypt	ahramgate
18 03 2020	145	Egypt	ahramgate

Table 4: Article count over time for each country and newspaper. Columns are Date, ArticleCount, Country, Newspaper

3. **Keyword Frequency**: The objective was to identify the most frequently occurring words in the dataset, along with change over time. Texts were tokenized, and stop words were removed (Code F).

2020-01	شركة	12
2020-01	رحلاتها	13
2020-02	الصحية	122
2020-02	يعد	17
2020-02	الجهاز	12
2020-02	التنفسي	14
2020-02	فيروس	351
2020-02	كورونا	573
2020-02	وأوضح	11
2020-02	العدوى	21

Table 5: Keyword frequency over time for Morocco. Columns are Date, Keyword, Frequency.

4. **Collocations**: The objective was to understand the context in which key terms appear by identifying words that commonly occur together. For each keyword, the script identified neighbouring words within a specified window size and calculated their frequencies, along with change over time (Code G).

2020-01	المستجد	11
2020-01	سارس	9
2020-01	حالة	8
2020-01	انتشر	8
2020-01	جديد	7
2020-02	كورونا	1421
2020-02	انتشار	330
2020-02	الصين	171
2020-02	الجديد	151
2020-02	تفشي	124

Table 6: "فيروس" collocates over time for Yemen with a window size of 5. Columns are Date, Collocate, Frequency.

5. **Temporal trends**: Since the date column was appended every time, it was easy to track changes in article distribution, keyword frequency and collocates over time. Everything was grouped by month, allowing for the identification of significant trends and events.
6. **COVID-19 Statistics:** This section superimposes the official COVID-19 confirmed cases, recovered cases, and deaths on selected keywords to generate insight on what newspaper coverage affects the public (CSSEGISandData, 2020/2024).
7. **Visualization and Analysis**: Visualizations were created using Excel's PivotTables and charts to help facilitate analysis.

One of the major challenges was handling Arabic text, which involves processing text with diacritics, punctuation, and variations in word forms. This was addressed using Pyarabic to strip diacritics and normalize the text. Additionally, stop words were added to remove common but non-informative words as mentioned before.

Another challenge was the large dataset size, which made processing slow and memory intensive. To overcome this, the script was optimized with the help of online forums to process files in chunks, using efficient data structures like Counter for counting frequencies and standard debugging practices to locate where the code went wrong.

Since the dataset is unbalanced, the graphs are biased towards Egypt as shown below.

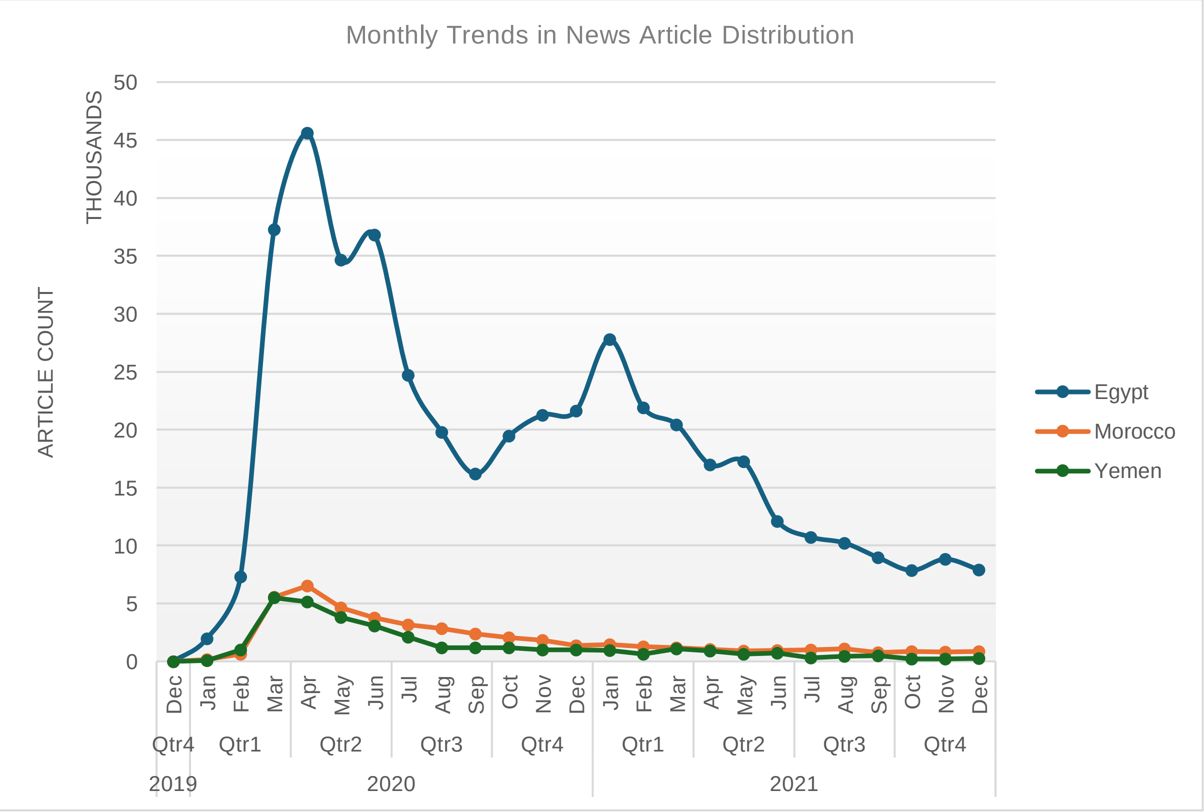


Figure 6: Line Graph of Monthly Trends in News Article Distribution for each country, unnormalized

To solve this problem, I show the values of sum of article count as a percentage of column total, in other words, normalized by total article count.

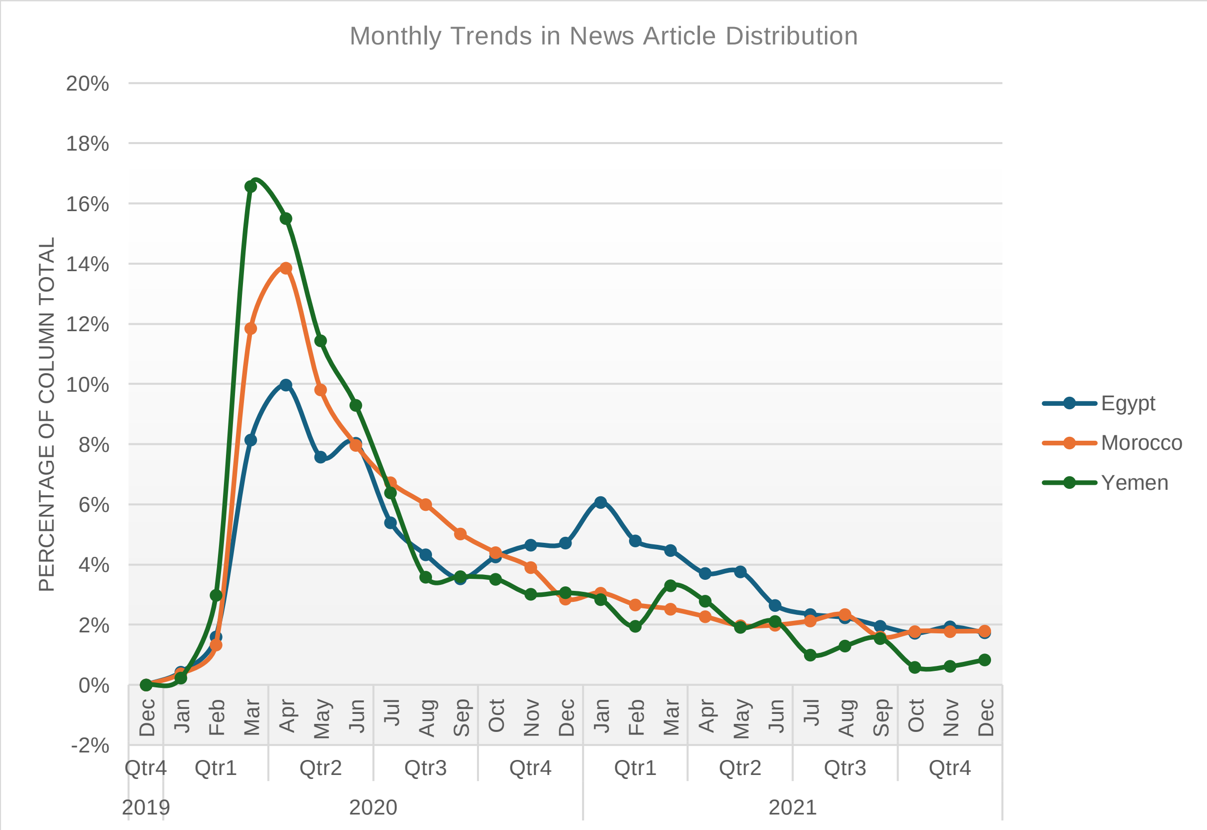


Figure 7: Line Graph of Monthly Trends in News Article Distribution for each country, normalized by total article count.

The temporal analysis also posed challenges, especially with standardizing date formats and grouping data by month. Pandas was used for date parsing and grouping, ensuring consistent temporal analysis. With temporal analysis, I did run into a problem where the date column was not recognized as a date since they were in the form [‘D-M-Y’]. I used python to parse through every csv file and cut the bracket and apostrophe (Code D). The date column in my results was printed as a string which complicated analysis in pivotable, so I had to again parse through them to transform them into a date. Thankfully this is a common problem, and a solution was available in the form of a short excel command.

$$= IF(MID(A1,5,1) = "/", DATEVALUE(TEXT(CONCATENATE(RIGHT(A1,2), "-", LEFT(A1,4)), "YYYY-MM-DD")), DATEVALUE(TEXT(A1, "YYYY-MM-DD")))$$

Equation 1: Excel command to convert date string to datevalue

Results and Interpretation

Throughout this section, I will be referring to events that correlate with an observation by the timeline in the appendix (Global COVID-19 Timeline).

Article Distribution

Referring to Table 2, Figure 5, and Figure 7 We can see that normalizing the data by total article count does help to remove bias. March to April 2020, there is a significant surge in news articles across all countries during this period, corresponding to the initial outbreak of COVID-29 and the subsequent declaration of a global pandemic by the World Health Organization (WHO) on March 11, 2020. This reflects the urgent public health crisis and the need for information dissemination. Post May 2020, there is a noticeable decline, indicating a normalization phase where the media shifts from initial shock to periodic updates with a stable reporting pattern. Late 2020 and Early 2021, Egypt shows secondary peaks possibly correlating with vaccine approvals and the continued waves of COVID-19 cases. Morocco and Yemen display consistent but lower peaks after the initial surge, suggesting less intensive media coverage compared to Egypt. Mid to Late 2021, there is a gradual decline because of the rollout of vaccines and the global adaptation to living with the virus, resulting in fewer breaking news stories.

Keyword Frequency

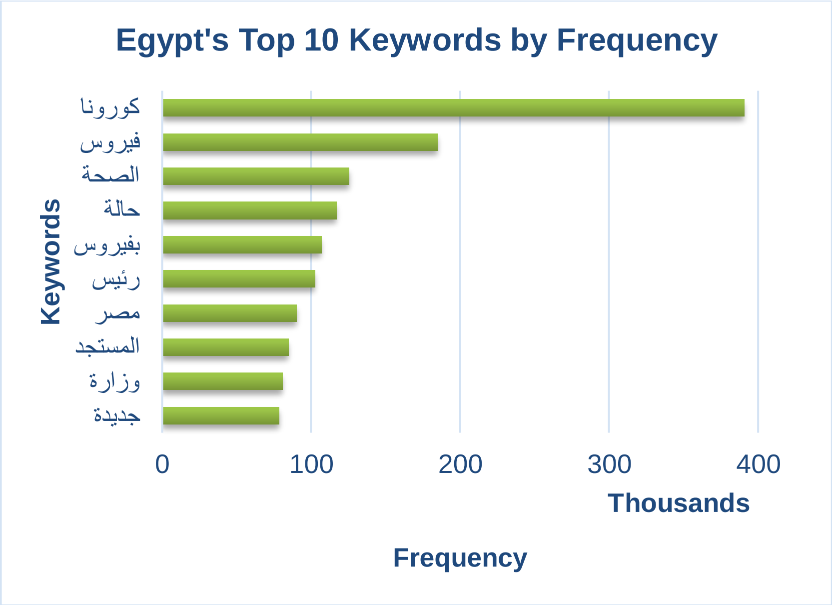


Figure 8: Bar Chart of Egypt's Top 10 Keywords by Frequency

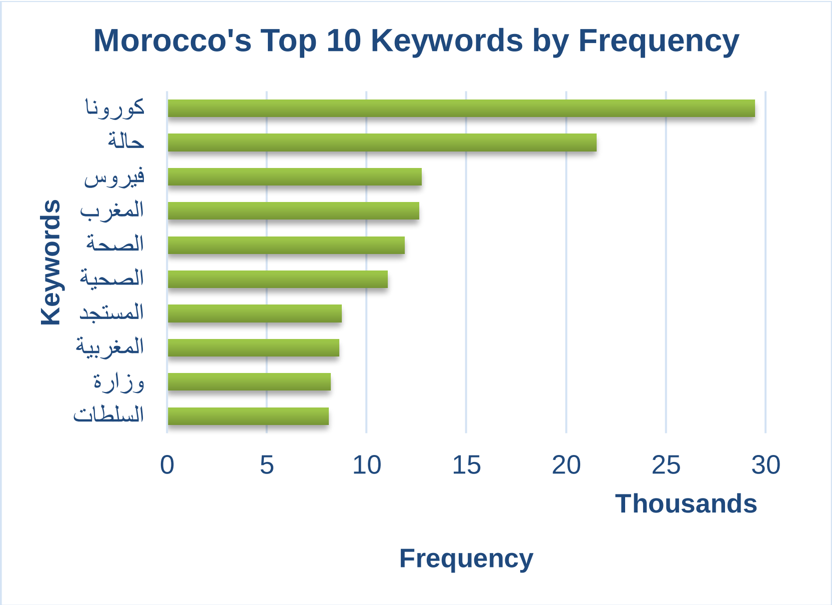


Figure 9: Bar Chart of Morocco's Top 10 Keywords by Frequency

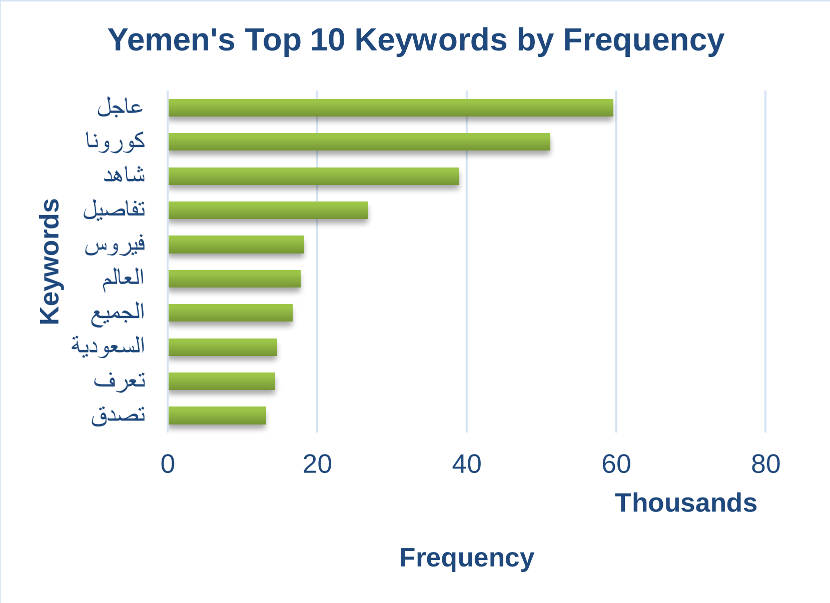


Figure 10: Bar Chart of Yemen's Top 10 Keywords by Frequency

Figure 8, Figure 9, and Figure 10 show the top 10 keywords by frequency for each country. For all countries "كورونا" (corona) is in the top 2, indicating its significant focus in news coverage. In Egypt and Morocco, mentions of "الصحة" (health) and "حالة" (case) indicate extensive reporting on health-related topics and the status of COVID-19 cases.*"*رئيس" (president), "وزارة" (ministry), "السلطات" (authorities) suggest an emphasis on governmental involvement and action, like the Ministry of Health. "مصر" (Egypt), "المغرب" (Morocco), and "المغربية" (Moroccan) indicate national context and identity. In Yemen, the high frequency of "عاجل" (urgent) implies an urgent and dynamic reporting style. The frequent mention of "العالم" (world) suggests the inclusion of global developments, while "السعودية" (Saudi Arabia) highlights the regional context.

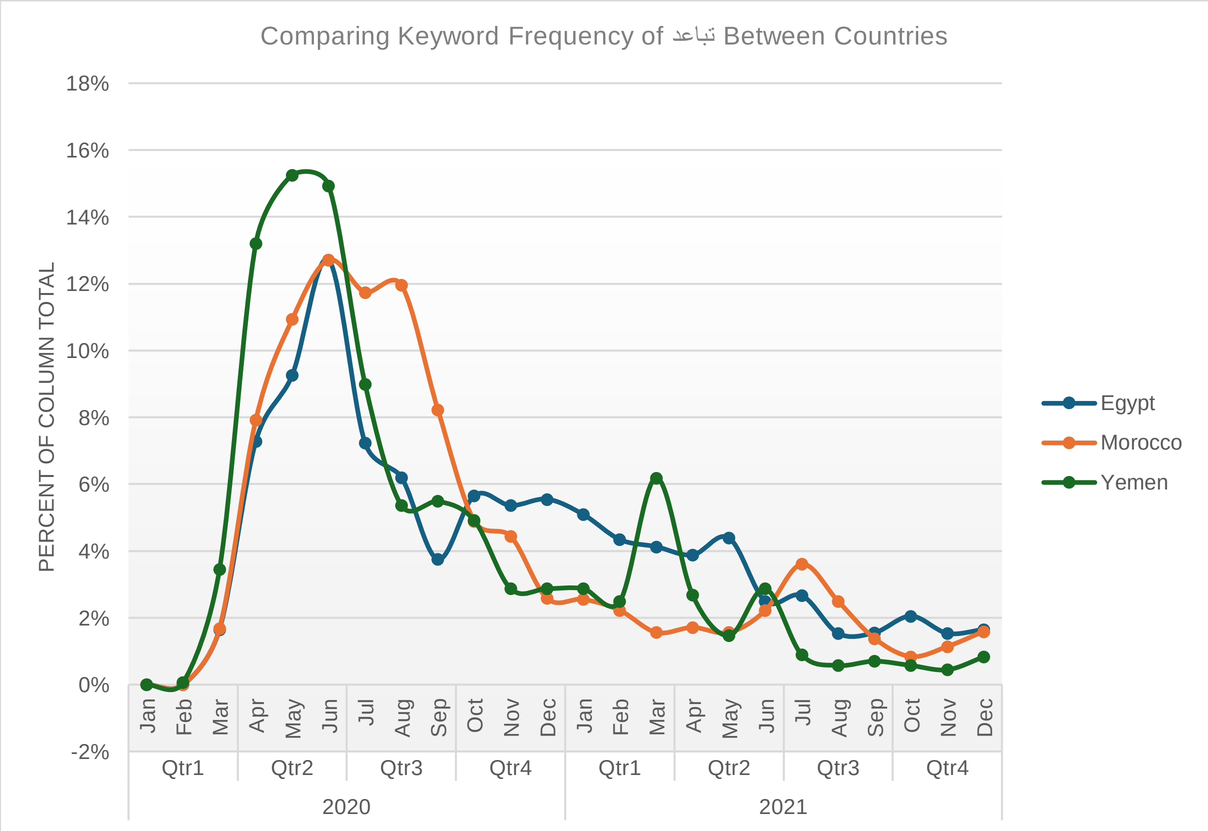


Figure 11: Line Graph of Monthly Trends in Keyword Frequency of "تباعد" Between Egypt, Morocco, and Yemen

Figure 11 shows the frequency of the keyword "تباعد" (distancing) in media coverage across Egypt, Morocco, and Yemen. The peaks in early 2020 align with the initial response to the pandemic and the implementation of social distancing measures. The sharp increase in March 2020 corresponds with the WHO's declaration of COVID-19 as a pandemic on March 11, 2020. Subsequent peaks and troughs reflect the ongoing discussions about social distancing measures as countries experienced various waves of COVID-19.

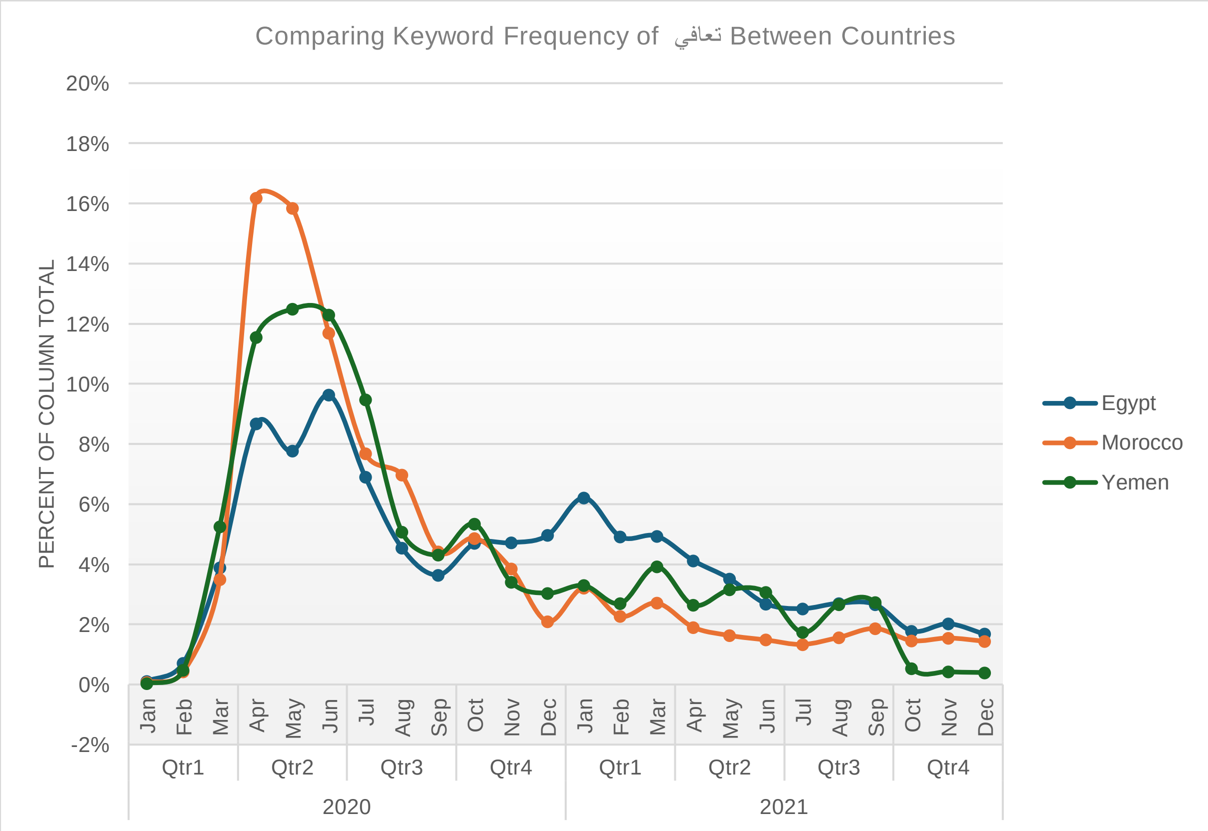


Figure 12: Line Graph of Monthly Trends in Keyword Frequency of "تعافي" Between Egypt, Morocco, and Yemen

Figure 12 illustrates the frequency of the keyword "تعافي" (recovery) in media coverage across Egypt, Morocco, and Yemen. The trends show significant peaks around key events related to recovery efforts, such as the rollout of vaccines. For instance, the increase in December 2020 and early 2021 corresponds with the approval and distribution of vaccines like Pfizer-BioNTech and Moderna. This highlights the media's focus on recovery narratives during critical periods of vaccine availability and administration.

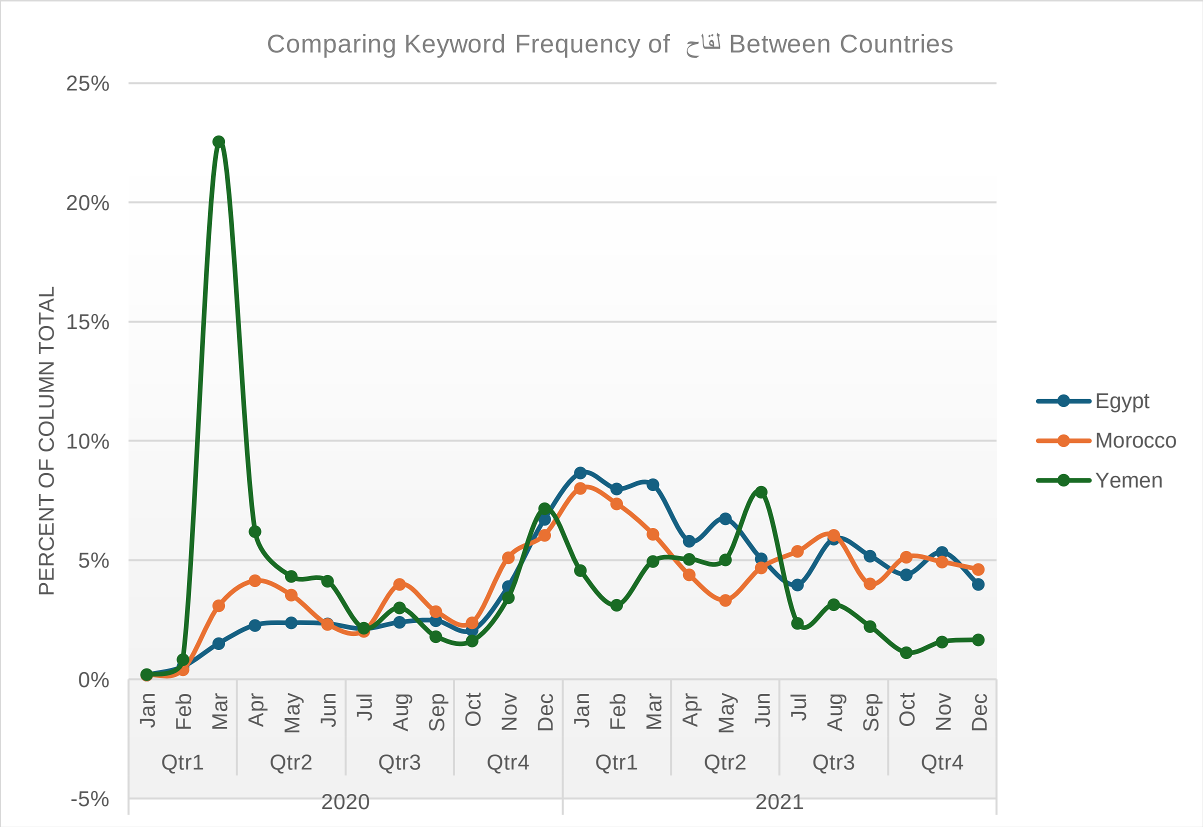


Figure 13: Line Graph of Monthly Trends in Keyword Frequency of "لقاح" Between Egypt, Morocco, and Yemen

Figure 13 depicts the frequency of the keyword "لقاح" (vaccine) in media coverage across Egypt, Morocco, and Yemen. Notable peaks in Yemen, especially in early 2020, underscore the critical importance of vaccination discussions to alleviate the humanitarian crisis. The sharp increase in December 2020 aligns with the UK's approval of the Pfizer/BioNTech vaccine on December 2, 2020, and subsequent approvals and rollouts in other countries. The graph also reflects continued discussions about vaccines as new variants emerged and additional vaccines received approval throughout 2021.

Collocation Frequency

For this section, I decided to focus the collocation analysis on "فيروس" (virus), this way I have a clear theme in the collocations I find. I chose a window size of 5 in both directions so as to get as much information as possible.

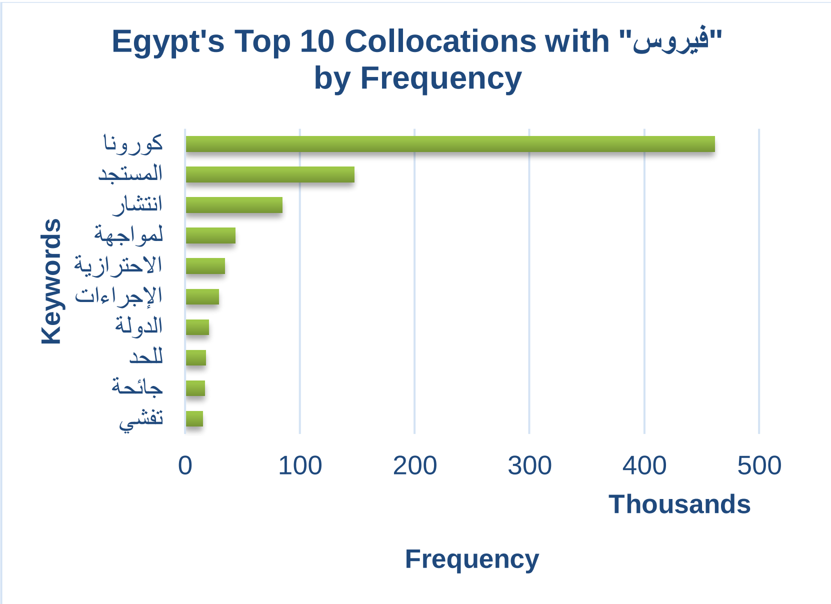


Figure 14: Bar Chart of Egypt's Top 10 Collocations with "فيروس" by Frequency

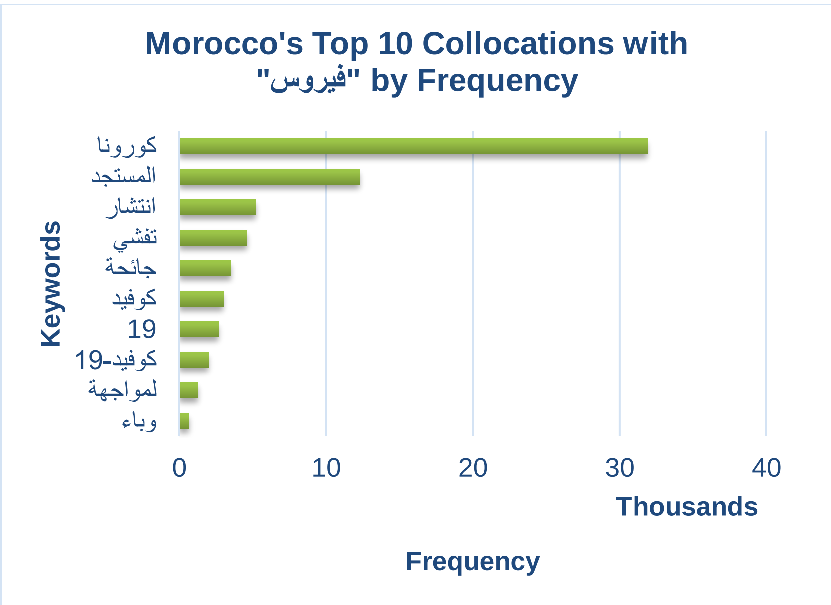


Figure 15: Bar Chart of Morocco's Top 10 Collocations with "فيروس" by Frequency

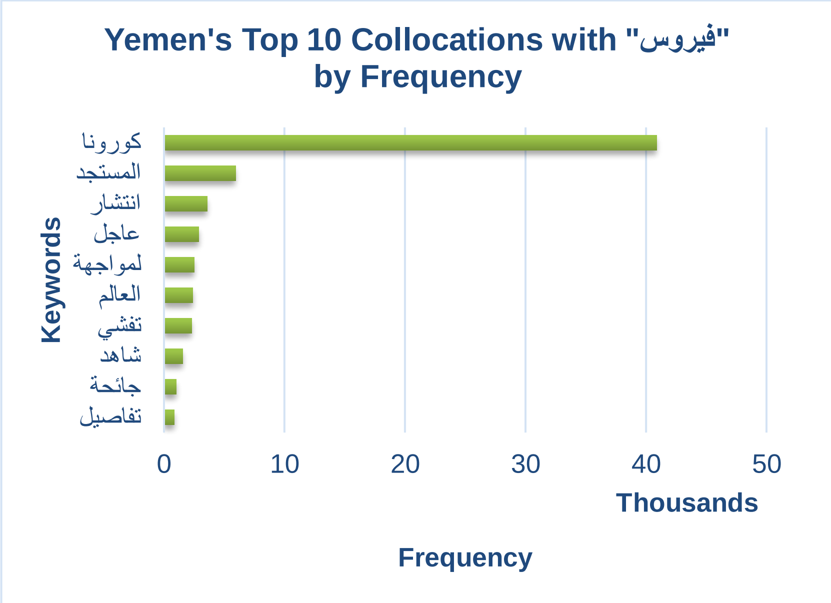


Figure 16: Bar Chart of Yemen's Top 10 Collocations with "فيروس" by Frequency

Figure 14, Figure 15, and Figure 16 show the top 10 collocations with "فيروس" (virus) for each country. For all countries, "كورونا" (corona) is the most frequent collocation, emphasizing its association with the COVID-19 virus. In Egypt, terms like "المستجد" (new), "انتشار" (spread), and "مواجهة" (confrontation) reflect a focus on addressing the virus's spread and impact. We can notice an n-gram of " فيروس كورونا المستجد", which can be added to with "انتشار" or "مواجهة". Additionally, mentions of "الإجراءات" (measures), "الدولة" (state), and "الاحترازية" (precautionary) highlight discussions around government responses and public health measures. Morocco shows a similar pattern with "المستجد" (new) and "انتشار" (spread) being prominent, alongside "تفشي" (outbreak) and "جائحة" (pandemic), which point to the broader context of the global impact. The separate mentions of "كوفيد" and "19" indicate specific discussions about COVID-19. In Yemen, while "كورونا" remains the most prominent collocation, the frequency of other terms is much lower, indicating fewer articles or discussions about these terms. Terms like "انتشار" (spread), "مواجهة" (confrontation), and "تفشي" (outbreak) reflect concerns about the virus's spread and efforts to combat it. The lower frequencies of collocations may suggest less extensive coverage in Yemeni newspapers compared to Egypt and Morocco.

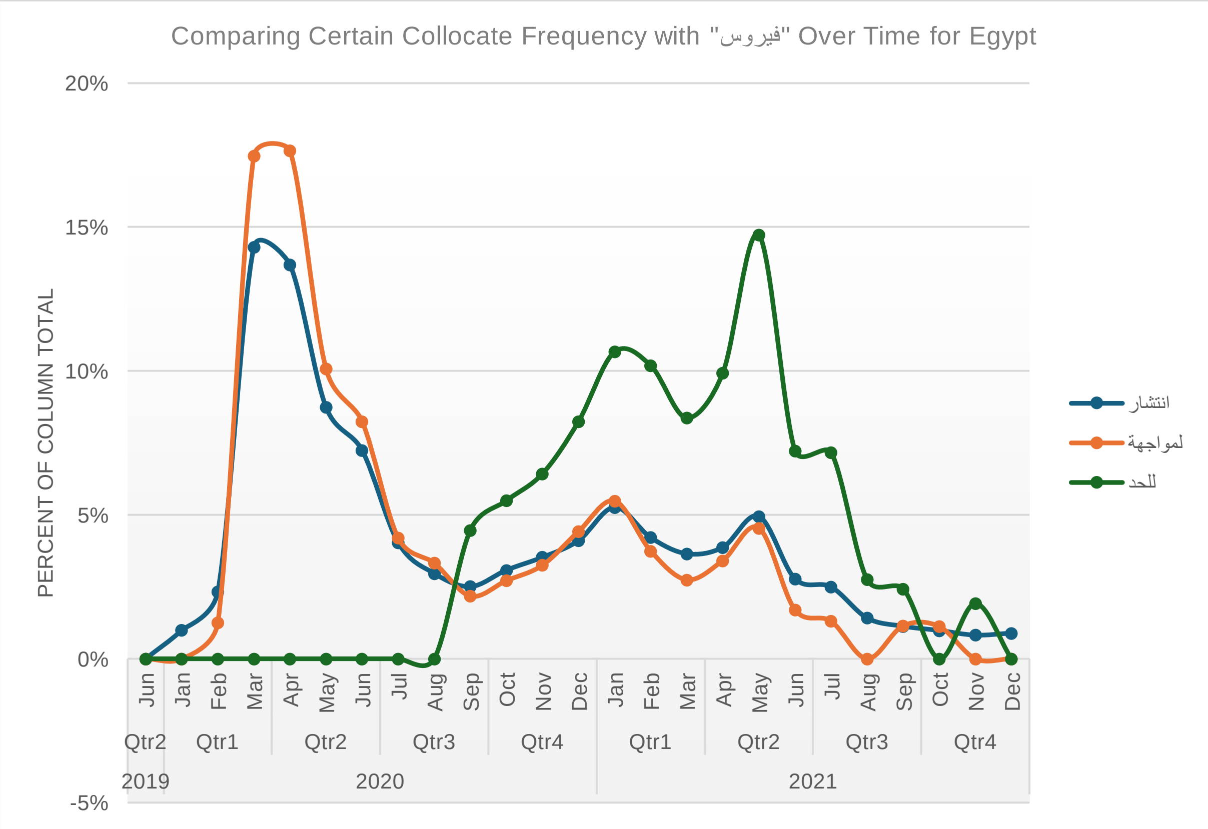


Figure 17: Line Graph of Monthly Trends in Certain Collocate Frequency with "فيروس" for Egypt

Figure 17 compares the frequency of the collocates "انتشار" (spread), "مواجهة" (confrontation), and "الحد" (limit) in Egyptian news articles over time. The term "مواجهة" (confrontation) shows a significant spike in early 2020, peaking around March and April, which coincides with the initial outbreak and rapid escalation of the COVID-19 pandemic. This suggests intense media focus on efforts to confront the virus during this period. The collocate "انتشار" (spread) also shows notable peaks, aligning with key phases of the pandemic's progression, including the initial spread in early 2020 and subsequent waves in late 2020 and mid-2021. The frequency of "الحد" (limit) starts to increase significantly in mid-2020, with a marked peak around June 2021, reflecting ongoing discussions about limiting the virus's spread and the implementation of control measures. Overall, the graph highlights the dynamic nature of media coverage, with fluctuations corresponding to the evolving stages of the pandemic and the associated public health responses in Egypt.

COVID-19 Statistics

For this section only Egypt will be analysed as it has the largest proportion in my dataset, therefore it would give reliable results.

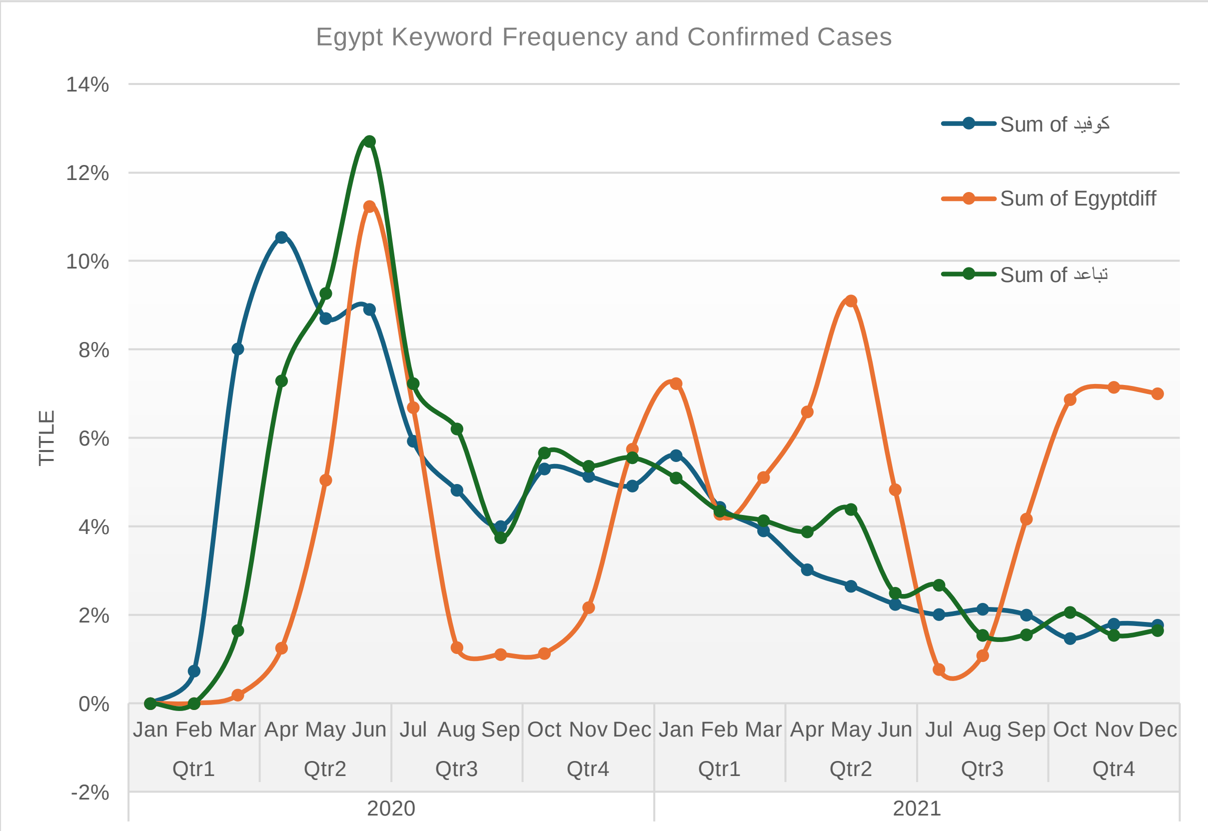


Figure 18: Line Graph of Monthly Trends in Certain Keyword Frequency with Confirmed Cases for Egypt

Figure 18 shows the trends of keyword mentions and confirmed COVID-19 cases (Egyptdiff) over time. The keywords "كوفيد" (COVID), "تباعد" (social distancing), The blue line for "كوفيد" rises sharply from February 2020, peaking around April 2020, warning the public of the imminent pandemic. Then as the cases rise, "تباعد" shows a similar trend, peaking together, indicating increased mentions of social distancing measures as cases rose. The sharp decrease after that implies that it works. For 2020, the keyword frequency closely follows the trends in confirmed cases, reflecting the media's response to the pandemic's progression. However, the subsequent waves with similar peaks do not generate the same response, either the population have learned to live with COVID-19 or the media reduced coverage.

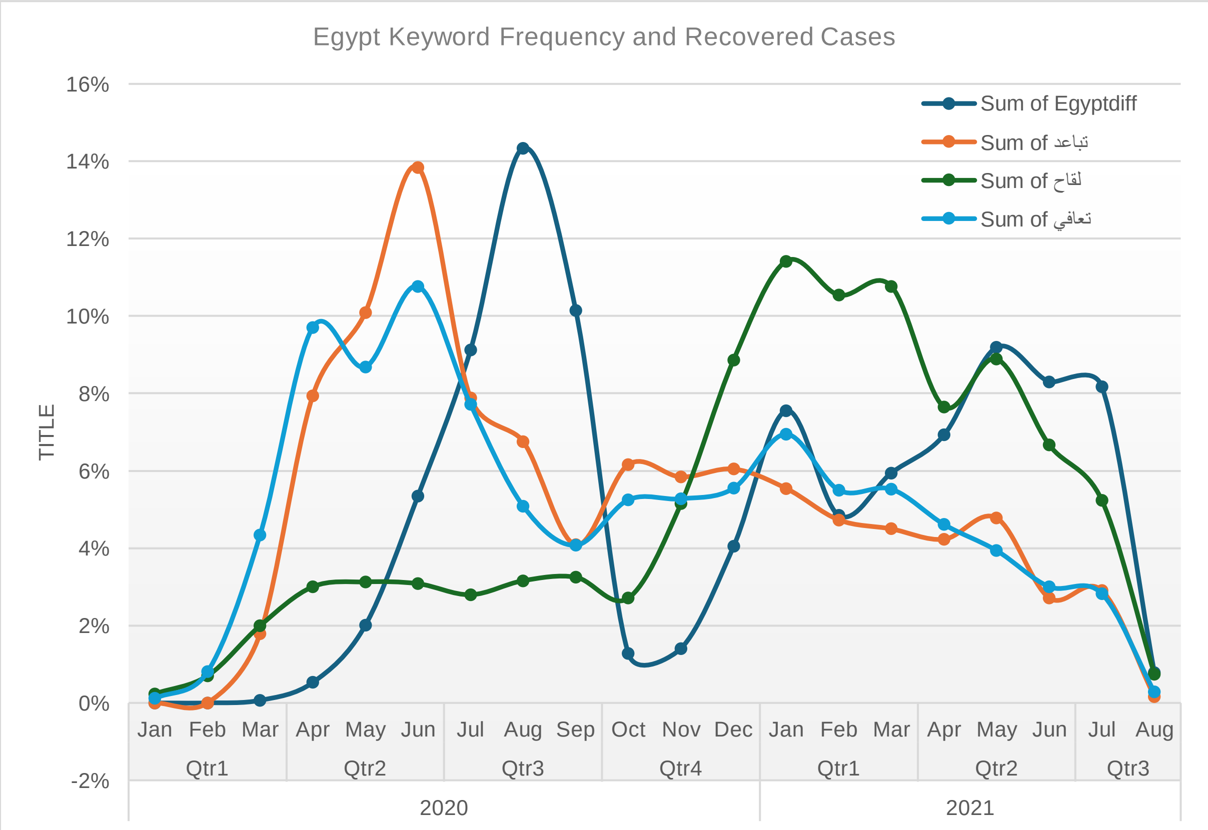


Figure 19: Line Graph of Monthly Trends in Certain Keyword Frequency with Recovered Cases for Egypt

Figure 19 depicts the trends of keyword mentions related to recovered cases (Egyptdiff) over time. The keywords "تباعد" (social distancing), "لقاح" (vaccine), and "تعافي" (recovery) are shown. Recovered cases shows a steady increase and follows by a few months with "تباعد" and "تعافي" during major recovery phases. "لقاح" surges towards the end of 2020 along with recovered cases, showcasing that people have started to take the vaccine. The correlation between keyword frequency and recovered cases highlights the media's focus on recovery and health measures in response to the pandemic's developments.

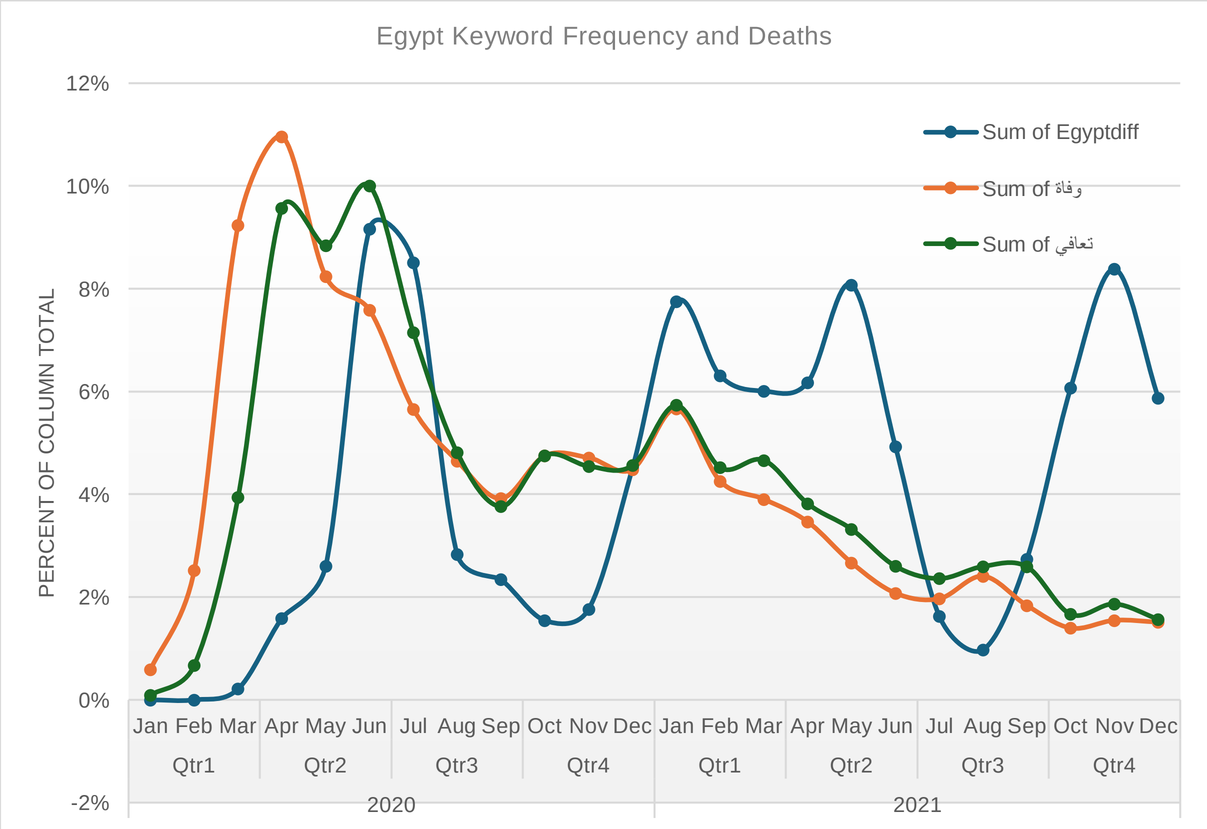


Figure 20: Line Graph of Monthly Trends in Certain Keyword Frequency with Deaths for Egypt

Figure 20 shows the relationship between the frequency of specific keywords and the number of deaths due to COVID-19 (Egyptdiff) in Egypt. the keywords "وفاة" (deaths) and "تعافي" (recovered) show peaks in early 2020, aligning with the start of the pandemic when media coverage was intense due to the virus's novel and impactful nature. They peak together indicating significant media attention to offer positive news amidst the rising concern. As they decrease the deaths in Egypt surge, indicating that the initial talk of deaths was about other countries, or the small amount of deaths in quarter 1. Once again, it’s clear that deaths peaks similarly for all waves, but the keywords do not.

The analysis of media coverage on COVID-19 in the Middle East is strengthened by the comprehensive dataset utilized, the Arabic Newspaper COVID-19 Corpus (AraNPCC). The temporal analysis conducted is another significant strength, as it provides insights into how media coverage evolved throughout the pandemic, helping to identify key moments when media focus shifted. The use of multiple tools, including Python for data processing and analysis, and Excel for visualization, leverages the strengths of both platforms. Python's libraries, such as Pandas and NLTK, efficiently handle large datasets and text processing, while Excel's Power Query and PivotTables facilitate clear and interpretable visual representations. Finally, the study's keyword and collocation analysis provide a nuanced understanding of how specific terms were used in context, shedding light on the framing and focus of media reports.

Despite the strengths, the analysis has several weaknesses and limitations. One significant challenge was data cleaning and preprocessing. Inconsistent date formats and variations in keyword spellings might have affected the results, highlighting the difficulties of working with automatically collected data from diverse sources. An example is in the following table where words were stitched together either from AraNPCC or the python script.

1 10 2021	كورونا	3250	Egypt
1 06 2020	بكورونا	3124	Egypt
1 04 2020	بكورونا	2720	Egypt
1 05 2020	بكورونا	2297	Egypt
1 04 2020	كوروناالموضوعات	2010	Egypt
1 03 2020	كوروناالثلاثاء	1865	Egypt
1 03 2020	كوروناالإثنين	1833	Egypt
1 03 2020	كوروناالموضوعات	1810	Egypt
1 05 2020	كوروناالموضوعات	1801	Egypt
1 03 2020	بكورونا	1759	Egypt

Table 7: Table showing error in tokenization, columns are Date, Keyword, Count, Country.

The stop words list used, although helpful, might not have been exhaustive, potentially including some common words that could skew the results, or even removed some. Moreover, the automatic collection method of the dataset may introduce biases based on the accessibility and availability of online newspaper archives, affecting the generalizability of the findings. The temporal resolution of the data, being monthly, may not capture more nuanced shifts in media coverage, which could be better understood with daily or weekly analysis. There is also the potential for confirmation bias in the selection of keywords and collocations, where the analysis might focus on expected patterns and overlook unexpected ones. Furthermore, the lack of contextual analysis of the surrounding text means that the study misses out on deeper insights into the tone and implications of the media coverage. Lastly, the exclusion of visual and multimedia content, which are significant components of modern news coverage, limits the analysis.

Several potential biases could affect the analysis. Firstly, cultural and political biases inherent in the media sources might reflect the perspectives and priorities of their respective countries, influencing the findings by amplifying or suppressing certain narratives or viewpoints. Secondly, publication bias, driven by factors such as editorial policies, audience interests, and external events, could result in uneven coverage, with some periods or topics receiving disproportionate attention. Thirdly, the diverse nature of the Arabic language, with its numerous dialects, may not be fully accounted for in the analysis, potentially missing out on regional linguistic variations and colloquial expressions. Lastly, the inherent bias in the media's portrayal of events could skew the representation of data, affecting the interpretation and conclusions drawn from the analysis.

Conclusion

This project provided an in-depth analysis of newspaper coverage on COVID-19 in the Middle East using the Arabic Newspaper COVID-19 Corpus (AraNPCC). The key findings revealed significant trends and patterns in media reporting across Egypt, Morocco, and Yemen during the pandemic. The analysis highlighted the peaks in media coverage corresponding to major pandemic events, such as the initial outbreak, the declaration of COVID-19 as a pandemic by the WHO, and the rollout of vaccines. The keyword and collocation analysis identified the most frequently mentioned terms and their contextual relationships, offering insights into the media's focus on public health, government responses, and societal impacts.

The temporal analysis demonstrated how media coverage evolved over time, reflecting the changing nature of the pandemic and the media's role in informing the public. The normalization of data helped to mitigate biases due to the unbalanced dataset. The visualization of keyword frequencies and collocations over time provided a clear representation of media trends, emphasizing the importance of specific keywords during critical periods.

The implications of this project for future research are substantial. Future research could extend the analysis to include more countries or a longer time frame, providing a broader perspective on media reporting. Additionally, incorporating visual and multimedia content could offer a more comprehensive understanding of modern news coverage. Refinements in data cleaning and preprocessing, particularly with handling Arabic text and diverse dialects, may enhance the accuracy of the analysis.

Reflecting on the broader impact of this work on the field of Arabic digital humanities, this project demonstrates the value of combining textual and temporal analysis methods to uncover insights from large datasets. It highlights the potential of digital tools and methodologies in analysing Arabic language corpora, contributing to the growing body of knowledge in this field. By providing a detailed examination of media coverage during the COVID-19 pandemic, this study not only enhances our understanding of the media's role but also offers valuable data for policymakers, researchers, and media analysts. The findings can inform strategies for effective communication during public health crises, ultimately benefiting the wider community by improving information dissemination and public awareness.

References

Alrefaie, M. T. (2019). *Arabic Stop Words* [Computer software]. https://github.com/mohataher/arabic-stop-words (Original work published 2016)

Al-Thubaity, A., Alkhereyf, S., & Bahanshal, A. O. (2022). AraNPCC: The Arabic Newspaper COVID-19 Corpus. In H. Al-Khalifa, T. Elsayed, H. Mubarak, A. Al-Thubaity, W. Magdy, & K. Darwish (Eds.), *Proceedinsg of the 5th Workshop on Open-Source Arabic Corpora and Processing Tools with Shared Tasks on Qur'an QA and Fine-Grained Hate Speech Detection* (pp. 32–40). European Language Resources Association. https://aclanthology.org/2022.osact-1.4

COVID-19 (SARS-CoV-2 Coronavirus) Resources. (n.d.). ASM.Org. https://asm.org:443/Resource-Pages/COVID-19-Resources

CSSEGISandData. (2024). *CSSEGISandData/COVID-19* [Computer software]. https://github.com/CSSEGISandData/COVID-19 (Original work published 2020)

Appendix

Global COVID-19 Timeline

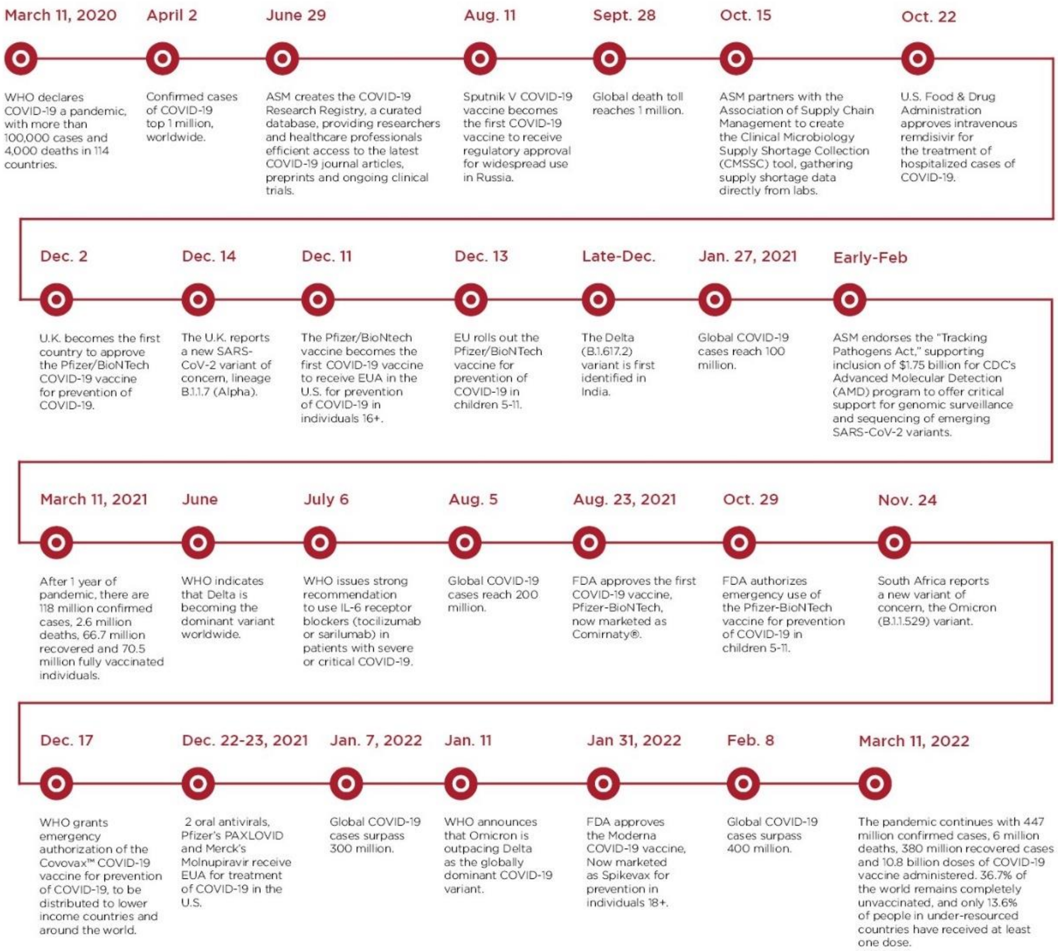


*Figure 21: 2 Year Timeline of COVID-19 (*COVID-19 (SARS-CoV-2 Coronavirus) Resources*, n.d.).*

Code A: Libraries

```
import csv
import os
import pandas as pd
import sys
import string
import nltk
import json
import re
from datetime import datetime
from nltk.tokenize import word_tokenize
from nltk.collocations import BigramCollocationFinder
from nltk.metrics import BigramAssocMeasures
from nltk.util import ngrams
from pyarabic import araby
from pyarabic.araby import strip_tashkeel
from collections import Counter, defaultdict
import xlsxwriter
import arabic_reshaper
from bidi.algorithm import get_display
from openpyxl import Workbook
```

Code B: Make json of files

```python
def generate_detailed_country_newspaper_json(base_path):
    country_newspapers = {}

    for root, dirs, files in os.walk(base_path):
        parts = root.split(os.sep)
        if len(parts) > 1 and parts[-1].startswith("AraNPCC_"):
            country = parts[-1].replace("AraNPCC_", "")
            newspapers = {}

            for file_name in files:
                if file_name.endswith('.csv'):
                    # Extract the newspaper name by removing the year and file extension
                    newspaper_name = '_'.join(file_name.split('_')[1:-1])
                    if newspaper_name not in newspapers:
                        newspapers[newspaper_name] = []
                    newspapers[newspaper_name].append(file_name)

            country_newspapers[country] = newspapers

    json_path = os.path.join(base_path, 'detailed_country_newspapers.json')
    with open(json_path, 'w') as json_file:
        json.dump(country_newspapers, json_file, indent=4, ensure_ascii=False)

    return json_path

base_path = r"C:\Users\khali\OneDrive\AUS\Classes\7 - S24\ARA 250\Project\AraNPCC"
json_path = generate_detailed_country_newspaper_json(base_path)
print("Detailed JSON created at:", json_path)
```

Code C: Filter COVID-19 related articles

```
def set_max_csv_field_size():
    max_int_c_long = 2147483647
    try:
        csv.field_size_limit(max_int_c_long)
        print(f"{datetime.now()}: CSV field size limit set to
{max_int_c_long}")
    except OverflowError as e:
        print(f"{datetime.now()}: OverflowError encountered while
setting field size limit:", e)


def load_json_reference(json_path):
    with open(json_path, 'r', encoding='utf-8') as file:
        return json.load(file)


def tokenize_search_and_context_to_csv(json_reference_path,
output_dir, country_name, search_terms, window=10):
    set_max_csv_field_size()
    search_terms_set = set(search_terms)
    results_header = ["Text", "Title", "URL", "Date", "Category",
"Newspaper", "File_Name", "Term"]

    # Load JSON reference
    country_newspapers = load_json_reference(json_reference_path)
    newspapers = country_newspapers.get(country_name, {})

    for newspaper, files in newspapers.items():
        output_file_name =
f"{country_name}_{newspaper}_search_results.csv"
        output_file_path = os.path.join(output_dir,
output_file_name)
        processed_articles = set()

        with open(output_file_path, mode='w', newline='',
encoding='utf-8') as file:
            writer = csv.DictWriter(file, fieldnames=results_header)
```

```
            writer.writeheader()
            print(f"{datetime.now()}: Creating file for {newspaper}:
{output_file_name}")

            for filename in files:
                file_path = os.path.join(output_dir,
f"AraNPCC_{country_name}", filename)
                print(f"{datetime.now()}: Processing file:
{file_path}")
                if not os.path.exists(file_path):
                    print(f"{datetime.now()}: File not found:
{file_path}")
                    continue

                with open(file_path, mode='r', encoding='utf-8') as
infile:
                    csv_reader = csv.DictReader(infile,
delimiter='\t')
                    for row in csv_reader:
                        text = row.get('Text', '').lower()
                        tokens = word_tokenize(text)
                        found_terms =
search_terms_set.intersection(tokens)

                        article_key = (row.get('Title', ''),
row.get('Date', ''), newspaper)
                        if found_terms and article_key not in
processed_articles:
                            processed_articles.add(article_key)
                            result = {
                                "Text": text,
                                "Title": row.get('Title', ''),
                                "URL": row.get('URL', ''),
                                "Date": row.get('Date', ''),
                                "Category": row.get('Category', ''),
                                "Newspaper": newspaper,
                                "File_Name": filename,
```

```
                                "Term": ", ".join(found_terms)
                            }
                            writer.writerow(result)
            print(f"{datetime.now()}: Results written to 
{output_file_path}")


json_reference_path = r'C:\Users\khali\OneDrive\AUS\Classes\7 - 
S24\ARA 250\Project\AraNPCC\detailed_country_newspapers.json'
output_dir = r"C:\Users\khali\OneDrive\AUS\Classes\7 - S24\ARA 
250\Project\AraNPCC"
country_name = 'Egypt'
search_terms = ['كوفيد', "كورونا"]
tokenize_search_and_context_to_csv(json_reference_path, output_dir, 
country_name, search_terms)
```

Code D: Clean date column

```
def clean_date(date_str):
    # Strip out unwanted characters [' and '] from the date string
    return date_str.strip("[]'")

def process_files(directory):
    # Loop through all files in the directory
    for filename in os.listdir(directory):
        if filename.endswith('.csv'):
            file_path = os.path.join(directory, filename)
            print(f"{datetime.now()}: Processing file: {file_path}")

            # Read the CSV file into a DataFrame
            df = pd.read_csv(file_path)

            # Check if 'Date' column exists in the DataFrame
            if 'Date' in df.columns:
                # Apply the cleaning function to the 'Date' column
                df['Date'] = df['Date'].apply(clean_date)

                # Save the cleaned DataFrame back to CSV
                df.to_csv(file_path, index=False)
                print(f"{datetime.now()}: Cleaned and saved:
{file_path}")
            else:
                print(f"{datetime.now()}: No 'Date' column found in:
{file_path}")

# Specify the directory containing your CSV files
directory = r'C:\Users\khali\OneDrive\AUS\Classes\7 - S24\ARA
250\Project\AraNPCC\COVID_Articles'
process_files(directory)
```

Code E: Count filtered articles by over time and by newspaper

```
def parse_date(date_str):
    """Attempt to parse the date with different expected formats."""
    for fmt in ('%d-%m-%Y', '%m-%d-%Y', '%Y-%m-%d'):
        try:
            return pd.to_datetime(date_str, format=fmt)
        except ValueError:
            continue
    return pd.NaT  # Return Not a Time (NaT) if all formats fail

def process_files(directory):
    results = []
    for file_name in os.listdir(directory):
        if file_name.endswith('.csv'):
            country, newspaper = parse_filename(file_name)
            file_path = os.path.join(directory, file_name)
            print(f"Processing file: {file_path}")

            try:
                df = pd.read_csv(file_path)
                # Apply robust date parsing
                df['Date'] = df['Date'].apply(parse_date)
                df_grouped =
df.groupby(df['Date'].dt.to_period('D')).size().reset_index(name='Ar
ticleCount')
                df_grouped['Country'] = country
                df_grouped['Newspaper'] = newspaper
                results.append(df_grouped)
            except Exception as e:
                print(f"Error processing {file_path}: {e}")

    final_df = pd.concat(results, ignore_index=True)
    return final_df
```

```
def save_results_to_csv(final_df, output_file):
    final_df.to_csv(output_file, index=False)
    print(f"Results written to {output_file}")

directory = r"C:\Users\khali\OneDrive\AUS\Classes\7 - S24\ARA 250\Project\AraNPCC\COVID_Articles"
output_csv_path = r"C:\Users\khali\OneDrive\AUS\Classes\7 - S24\ARA 250\Project\AraNPCC\covid_article_counts_by_date.csv"
final_aggregated_data = process_files(directory)
save_results_to_csv(final_aggregated_data, output_csv_path)
```

Code F: Global keyword frequency by country and month (min 10 count)

```
# Function to clean Arabic text
def clean_arabic_text(text):
    text = re.sub(r'[\u064B-\u065F]', '', text)  # Remove Arabic
diacritics
    text = re.sub(r'[^\w\s]', '', text)  # Remove punctuation
    text = re.sub(r'\d+', '', text)  # Remove digits
    return text

# Load stop words
def load_stop_words(file_path):
    with open(file_path, 'r', encoding='utf-8') as file:
        stop_words = file.read().splitlines()
    return set(stop_words)

# Function to calculate keyword frequencies
def calculate_keyword_frequencies(directory, stop_words):
    keyword_frequencies_by_country = {}

    # Iterate through all files in the specified directory
    for file_name in os.listdir(directory):
        if file_name.endswith('.csv'):
            file_path = os.path.join(directory, file_name)
            country_name = file_name.split('_')[0]  # Extract
country name from the file name
            try:
                df = pd.read_csv(file_path)
                df['Month'] = pd.to_datetime(df['Date'],
errors='coerce').dt.to_period('M')
                if country_name not in
keyword_frequencies_by_country:
                    keyword_frequencies_by_country[country_name] =
{}
                for month, group in df.groupby('Month'):
```

```
                    texts = group['Text'].dropna()  # Drop missing
values
                    monthly_frequencies = Counter()
                    for text in texts:
                        text = clean_arabic_text(text)
                        tokens = word_tokenize(text)
                        filtered_tokens = [token for token in tokens
if token not in stop_words]
                        monthly_frequencies.update(filtered_tokens)
                    keyword_frequencies_by_country[country_name][str
(month)] = Counter({word: count for word, count in
monthly_frequencies.items() if count >= 10})
            except Exception as e:
                print(f"Failed to process {file_path}: {e}")

    return keyword_frequencies_by_country

# Function to save the frequencies to an Excel file
def save_frequencies_to_excel(keyword_frequencies_by_country,
output_file):
    workbook = Workbook()
    for country, monthly_frequencies in
keyword_frequencies_by_country.items():
        sheet = workbook.create_sheet(title=country)
        sheet.append(['Month', 'Keyword', 'Frequency'])
        for month, frequencies in monthly_frequencies.items():
            for keyword, frequency in frequencies.items():
                sheet.append([month, keyword, frequency])

    # Remove the default sheet created by Workbook
    default_sheet = workbook['Sheet']
    workbook.remove(default_sheet)

    workbook.save(output_file)
    print(f"Frequencies saved to {output_file}")
```

```
directory = r"C:\Users\khali\OneDrive\AUS\Classes\7 - S24\ARA
250\Project\AraNPCC\COVID_Articles"
stop_words_path = r"C:\Users\khali\OneDrive\AUS\Classes\7 - S24\ARA
250\Project\AraNPCC\stop_words.txt"
output_excel_path = r"C:\Users\khali\OneDrive\AUS\Classes\7 -
S24\ARA 250\Project\AraNPCC\global_keyword_frequencies.xlsx"


stop_words = load_stop_words(stop_words_path)
keyword_frequencies_by_country =
calculate_keyword_frequencies(directory, stop_words)
save_frequencies_to_excel(keyword_frequencies_by_country,
output_excel_path)
```

Code G: Collocation of selected keyword, can specify window size

```
def parse_filename(file_name):
    """ Extract country from filename. Assumes format 
Country_Newspaper_Date.csv """
    return file_name.split('_')[0]

def clean_token(token, punctuation):
    """ Clean token by removing leading and trailing punctuation. 
"""
    return re.sub(r'^[' + punctuation + ']+|[' + punctuation + 
']+$', '', token)

def load_stop_words(file_path):
    """ Load stop words from a file. """
    with open(file_path, 'r', encoding='utf-8') as file:
        stop_words = set(file.read().splitlines())
    return stop_words

def find_collocations(text, keyword, window_size, stop_words):
    """ Find collocations around a specified keyword within the 
given window size, ignoring punctuation and stop words. """
    text = strip_tashkeel(text)
    tokens = word_tokenize(text)
    collocations = Counter()
    punctuation = string.punctuation + "“”‘’—،–"

    for i, token in enumerate(tokens):
        cleaned_token = clean_token(token, punctuation)
        if cleaned_token == keyword:
            start = max(0, i - window_size)
            end = min(len(tokens), i + window_size + 1)
            # Exclude tokens that are entirely punctuation or stop 
words
            window_tokens = [clean_token(t, punctuation) for t in 
tokens[start:i] + tokens[i+1:end] if not all(char in punctuation for 
char in t) and t not in stop_words]
```

```
            for gram in window_tokens:
                if gram:  # Ensure it's not empty after cleaning
                    collocations[gram] += 1
    return collocations

def process_files(directory, keyword, window_size, stop_words):
    """ Process each file to find collocations for the specified keyword, summed by country and month, ignoring punctuation. """
    country_collocations = defaultdict(lambda: defaultdict(Counter))
    for file_name in os.listdir(directory):
        if file_name.endswith('.csv'):
            country = parse_filename(file_name)
            file_path = os.path.join(directory, file_name)
            print(f"Processing file: {file_path}")

            try:
                df = pd.read_csv(file_path)
                df['Month'] = pd.to_datetime(df['Date'], errors='coerce', dayfirst=True).dt.to_period('M')
                for (month), group in df.groupby('Month'):
                    all_text = ' '.join(group['Text'].dropna())
                    collocations = find_collocations(all_text, keyword, window_size, stop_words)
                    country_collocations[country][month].update(collocations)
            except Exception as e:
                print(f"Error processing {file_path}: {e}")

    return country_collocations

def save_results_to_excel(country_collocations, output_file):
    """ Save the results to an Excel file with each country's collocations on separate sheets, organized by month. """
    with pd.ExcelWriter(output_file, engine='xlsxwriter') as writer:
        for country, months_data in country_collocations.items():
```

```
            rows = []
            for month, collocations in months_data.items():
                for col, freq in collocations.most_common(10):
                    rows.append({
                        'Month': str(month),
                        'Collocation': col,
                        'Frequency': freq
                    })
            if rows:
                df = pd.DataFrame(rows)
                df.sort_values(by=['Month', 'Frequency'],
ascending=[True, False], inplace=True)
                df.to_excel(writer, sheet_name=country, index=False)
                print(f"Results for {country} written to sheet in
{output_file}")
            else:
                print(f"No data for {country}.")

# Directory and parameters setup
directory = r"C:\Users\khali\OneDrive\AUS\Classes\7 - S24\ARA
250\Project\AraNPCC\COVID_Articles"
keyword = 'فيروس'
window_size = 5  # Number of words before and after the keyword
output_excel_path = r"C:\Users\khali\OneDrive\AUS\Classes\7 -
S24\ARA 250\Project\AraNPCC\collocations.xlsx"
stop_words_path = r"C:\Users\khali\OneDrive\AUS\Classes\7 - S24\ARA
250\Project\AraNPCC\stop_words.txt"

# Load stop words
arabic_stop_words = load_stop_words(stop_words_path)

# Process files and save results
country_collocations = process_files(directory, keyword,
window_size, arabic_stop_words)
save_results_to_excel(country_collocations, output_excel_path)
```