Understanding Aadhaar's Role in Governance: Insights from 15 Academic Articles

Ronaldo Rodrigues Alves Braga, MPA in Development Practice`2025

December 2024

Columbia University, SIPA

1. Abstract

This report investigates the role of Aadhaar, India's pioneering digital identity framework, in governance and public policy implementation, drawing insights from an analysis of 15 academic articles. The research aims to understand Aadhaar's integration into various sectors, such as welfare, economic inclusion, and digital governance, while assessing its benefits and challenges. The study employs a comprehensive methodology, including text preprocessing, keyword frequency analysis, and sentiment analysis, to explore the themes and perceptions associated with Aadhaar.

The findings reveal a nuanced perspective on Aadhaar: positive sentiment predominantly focuses on operational efficiency, transparency, and service improvements, while negative sentiment underscores concerns about privacy risks, social exclusion, and governance inefficiencies. Neutral sentiments dominate, reflecting a descriptive tone in academic and policy discussions, but spikes in polarized views highlight contentious issues like biometric data security and operational transparency. The keyword analysis further reveals themes of "efficiency," "value," and "service" in positive contexts, contrasted with "problems," "privacy," and "risks" in negative contexts.

This study contributes to the academic discourse by addressing gaps in Aadhaar-related literature, particularly the limited focus on its immediate benefits, such as cash transfers, and its broader implications for governance. By analyzing Aadhaar as a case study, the report offers actionable insights for emerging economies like Brazil, emphasizing the need for robust data privacy frameworks, strategies to mitigate exclusion risks, and the importance of showcasing tangible benefits to foster public trust. These findings provide a foundation for policymakers to balance innovation and safeguards in implementing digital identity systems.

Keywords: Digital Identity, Aadhaar, Biometric Governance, Public Policy Implementation, Welfare Programs, Data Privacy, Social Inclusion, Exclusion Risks, Governance Efficiency, Emerging Economies, Sentiment Analysis, Keyword Frequency, India, Brazil, Digital Transformation.

2. Introduction

Aadhaar, the world's largest biometric digital identity system, has redefined governance in India by providing a unique identification number to over 1.3 billion residents. Developed by the Unique Identification Authority of India (UIDAI), Aadhaar forms the backbone of numerous public policy initiatives, including welfare delivery, financial inclusion, and digital governance. By reducing inefficiencies, curbing leakages, and fostering transparency, Aadhaar has transformed service delivery systems (Abraham, Bennett, and Sen 2019). However, its implementation has also sparked debates over issues such as data privacy, risks of exclusion, and governance challenges, positioning Aadhaar as both an innovation benchmark and a subject of contention (Singh 2022; Khera 2020).

Brazil, like many emerging economies, faces challenges in ensuring equitable access to services, combating inefficiencies in welfare delivery, and modernizing governance systems. Aadhaar's extensive integration into India's socio-economic landscape provides a valuable case study for Brazil, which is exploring digital identity systems to address these issues (Subramanian 2021). Understanding Aadhaar's successes and pitfalls offers critical lessons for Brazil as it seeks to bridge gaps in digital inclusion and optimize public service delivery.

This report investigates Aadhaar's integration into public policy, focusing on its achievements, shortcomings, and broader implications. By analyzing 15 academic articles selected from a rigorous literature review, this study examines Aadhaar's role in governance, welfare, privacy, and biometric systems (Belorgey 2023; Wadhwa 2021). Employing methods such as sentiment analysis and keyword frequency analysis, the research uncovers key themes and perceptions surrounding Aadhaar. The findings aim to provide actionable insights for emerging economies, like Brazil, to implement inclusive, efficient, and secure digital identity systems, balancing innovation with safeguards.

3. Methodology

The research methodology combines a comprehensive literature review with advanced text analysis techniques to assess Aadhaar's integration into India's public policy landscape. This approach ensures a robust analysis of Aadhaar's role, its benefits, and its challenges in governance systems.

3.1 Article Selection

Fifteen academic articles were meticulously chosen from the CLIO library to maintain focus and relevance. The selection criteria emphasized articles that explored Aadhaar's role exclusively within the Indian governance framework. Comparative studies (e.g., Aadhaar vs. other countries like China) and localized cases unrelated to Aadhaar were excluded. The selected articles covered diverse yet interconnected topics such as governance, welfare programs, biometric systems, privacy concerns, and economic inclusion. This rigorous selection ensured a balanced representation of Aadhaar's multifaceted impact across multiple public policy areas.

3.2 Text Analysis

The textual content of the articles underwent systematic preprocessing to facilitate meaningful analysis:

- **Tokenization**: The text was segmented into individual words or sentences for a granular examination.
- **Stopword Removal**: Commonly used words (e.g., "and," "the") that do not carry significant meaning were filtered out. Additionally, custom stopwords such as "Aadhaar" and "government" were excluded to refine the analysis.
- **Lemmatization**: Words were reduced to their base forms (e.g., "using" to "use") to ensure consistency across the dataset.

3.3 Tools Used

A suite of Python libraries was employed for preprocessing, analysis, and visualization:

- **NLTK and SpaCy**: For preprocessing tasks like tokenization, stopword removal, and sentiment analysis.
- **Matplotlib and WordCloud**: For creating visualizations such as keyword frequency charts, sentiment trends, and word clouds.

3.4 Metrics Analyzed

Key metrics were used to extract meaningful insights:

1. **Keyword Frequency**: Identified the most discussed themes across the articles.
2. **Sentiment Distribution**: Assessed the overall tone (positive, neutral, negative) associated with Aadhaar.
3. **Sentiment Trends**: Highlighted emotional polarization and shifts in perceptions over time.

By integrating these advanced analytical techniques, the study provided a nuanced understanding of Aadhaar's dual narratives—its potential to transform governance and its associated risks. This methodology enabled a comprehensive evaluation of Aadhaar's role in public policy and its implications for emerging economies.

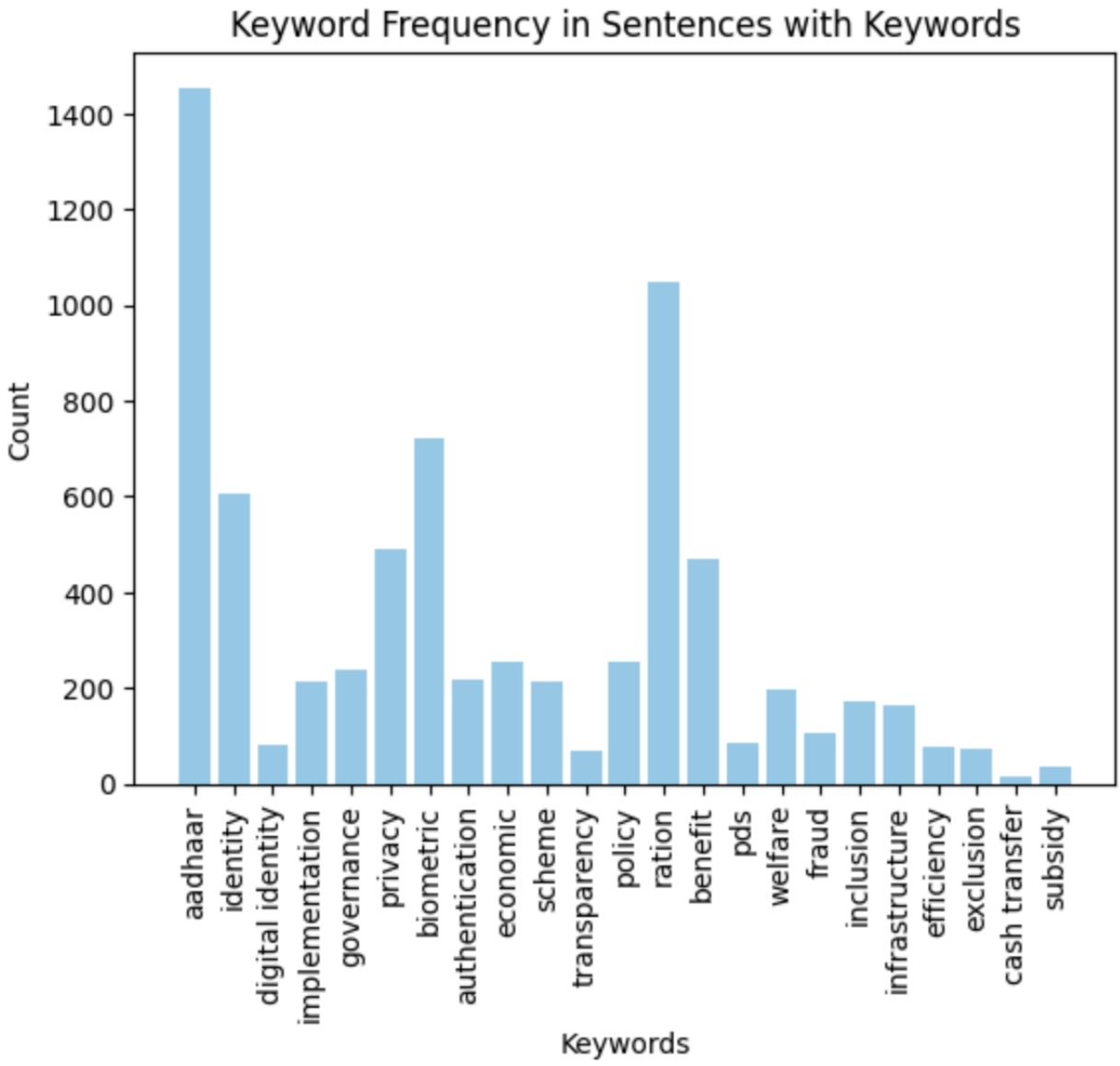
4.Results

The results of this study, derived from keyword frequency analysis, sentiment trends, and word cloud visualizations, offer key insights into Aadhaar's portrayal across governance and public policy narratives.

4.1 Keyword Frequency

The keyword frequency analysis revealed distinct trends in how Aadhaar-related topics are framed. **Positive sentiments** prominently featured terms like "data," "system," and "service," emphasizing Aadhaar's role in enhancing efficiency, service delivery, and value in governance systems. These terms underline the potential benefits of Aadhaar as a digital identity framework, particularly in improving transparency and access.

Conversely, **negative sentiments** were dominated by terms such as "problem," "privacy," and "risk." These reflect concerns surrounding Aadhaar's potential misuse, data security vulnerabilities, and its exclusionary risks for marginalized populations. Keywords like "fraud" and "error" also emerged in negative contexts, highlighting implementation challenges.

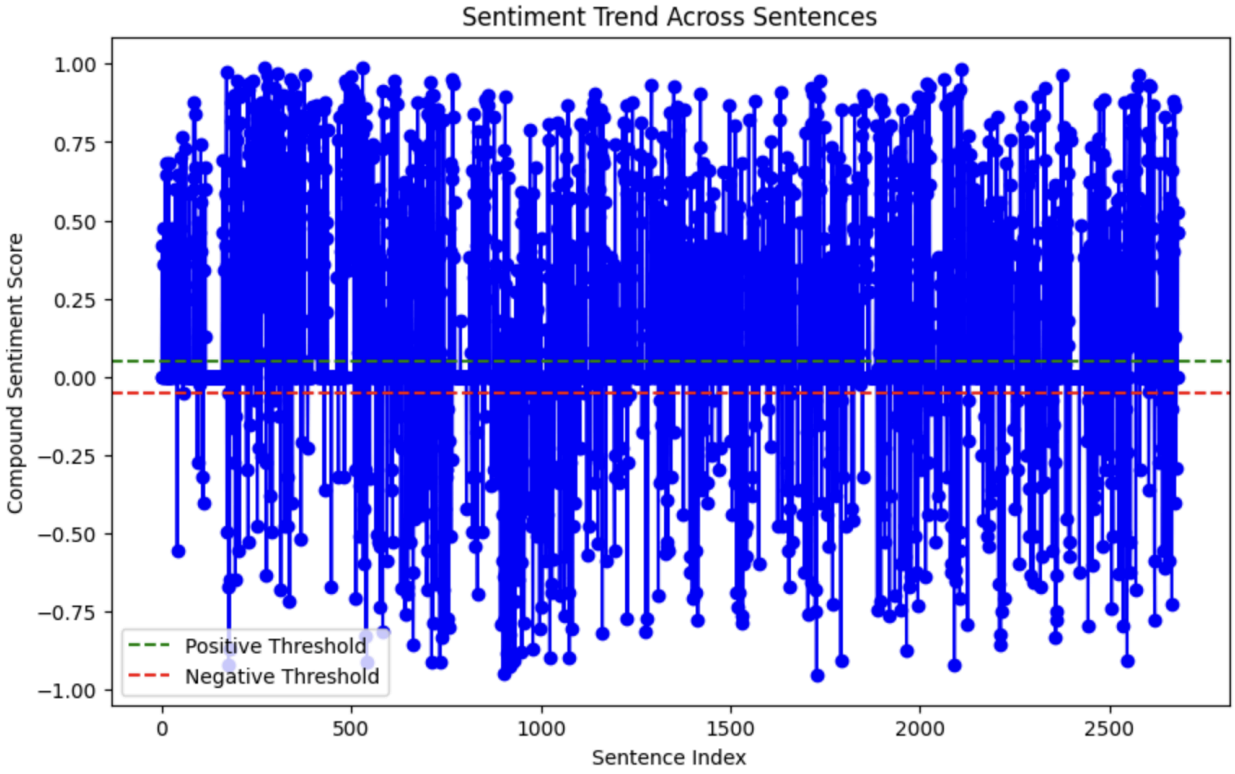


4.2 Sentiment Trends

The sentiment trend graph revealed a significant prevalence of **neutral sentiments**, comprising the majority of analyzed sentences. This indicates that much of the discourse on Aadhaar remains descriptive or factual, particularly in academic and policy-oriented texts.

However, instances of **polarized views** were also evident, with distinct spikes in both positive and negative sentiment. Positive spikes often correlated with discussions of Aadhaar's role in improving service delivery and operational efficiency. In contrast, negative spikes were linked to issues of privacy violations, errors in biometric identification, and social exclusion. These

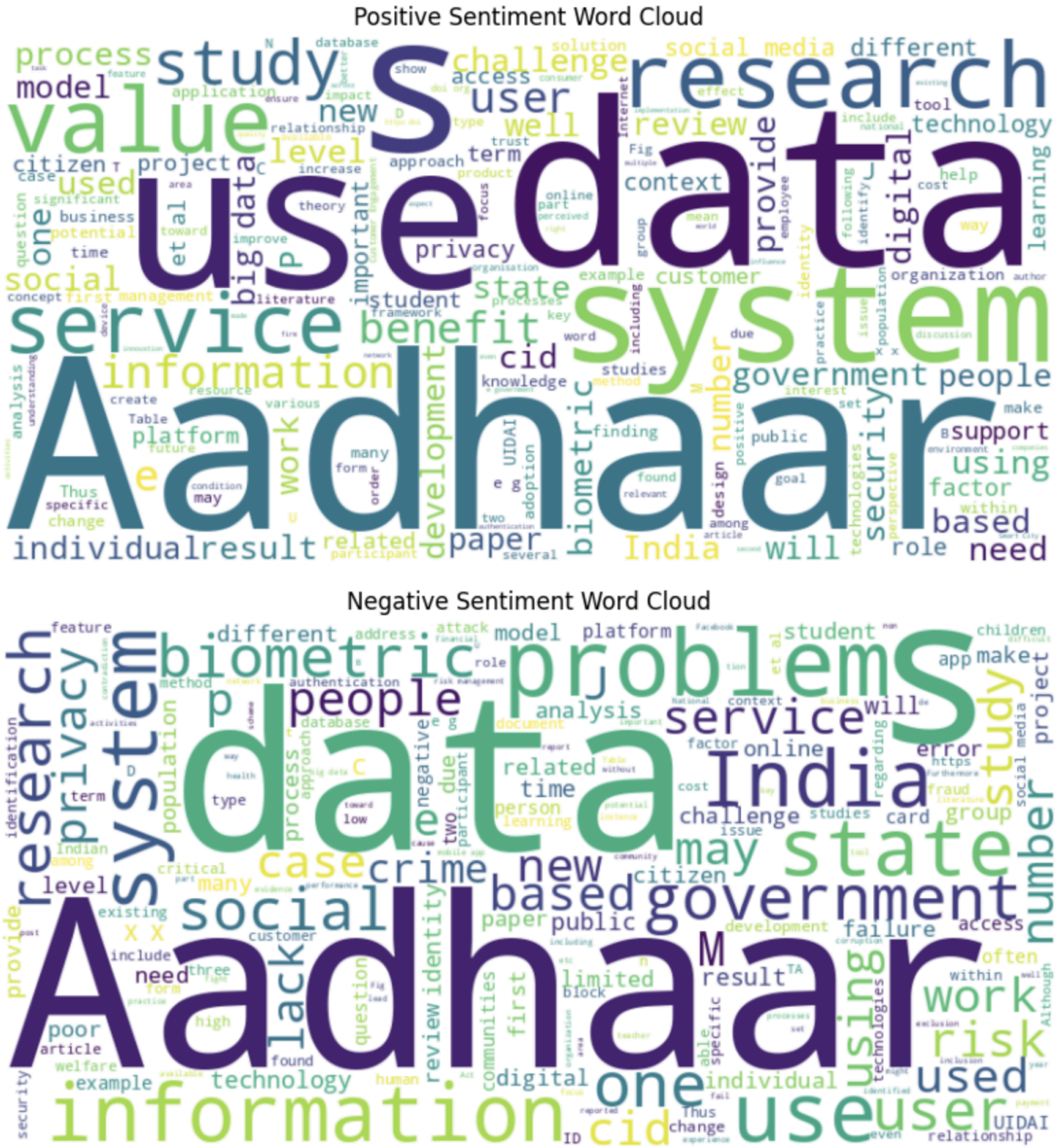
polarized views highlight the dual narrative around Aadhaar, oscillating between its potential benefits and its perceived risks.



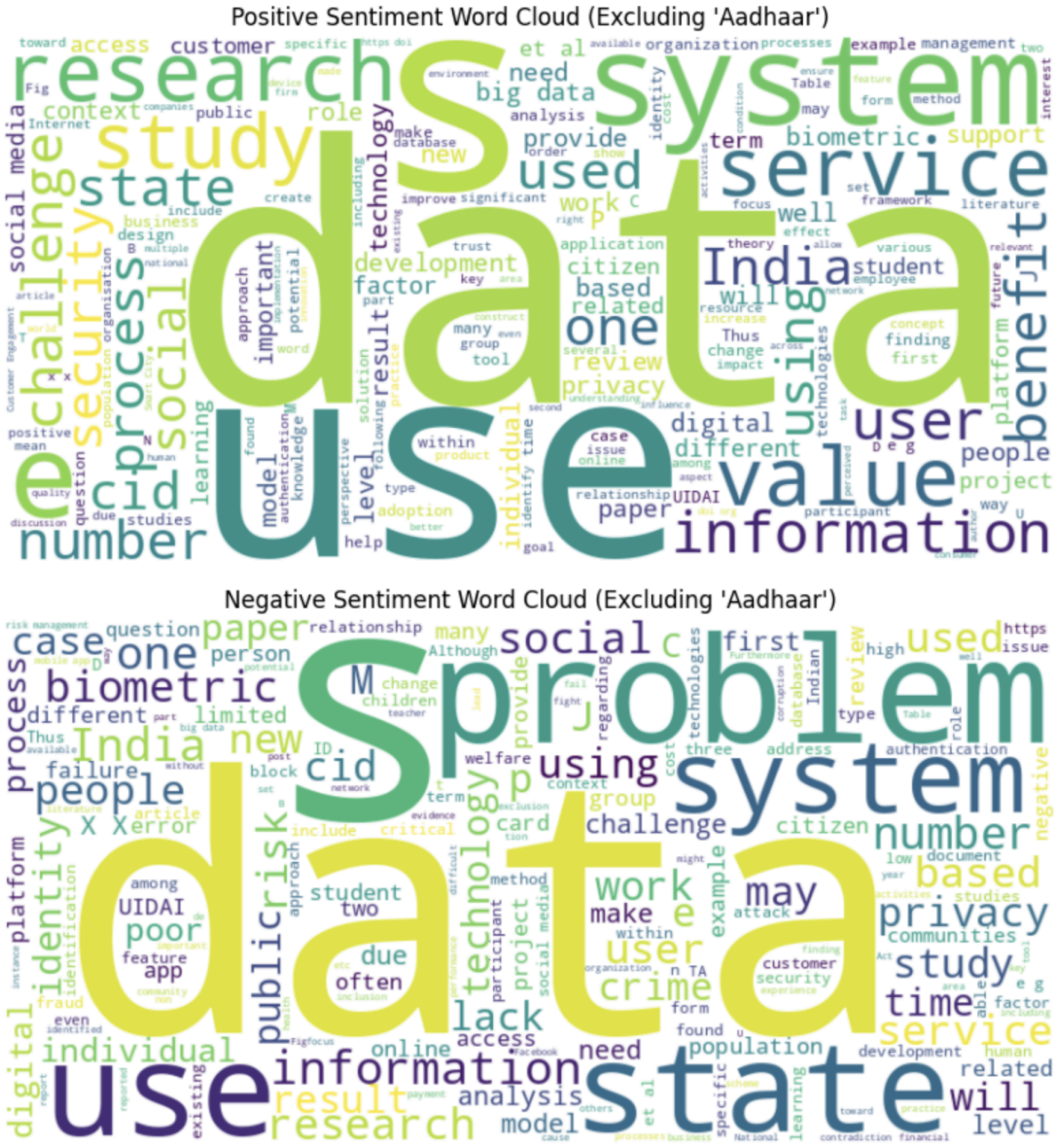
4.3 Word Cloud Analysis

The word cloud comparisons provided further depth to the sentiment analysis. In the **positive sentiment word cloud**, terms like “value,” “security,” “service,” and “benefit” were prominent. These terms reinforce Aadhaar’s perceived advantages in promoting inclusivity, reducing inefficiencies, and ensuring better service delivery.

In contrast, the **negative sentiment word cloud** was marked by terms such as “problems,” “privacy,” “risk,” and “fraud.” These terms signify recurring themes of concern, particularly around data security and the ethical implications of biometric identification.



When **“Aadhaar” and “government”** were excluded, the focus shifted to broader terms like “data,” “system,” and “service” in positive sentiments, and “problems,” “privacy,” and “risk” in negative sentiments. This exclusion provided a clearer view of the underlying issues and benefits associated with Aadhaar, beyond its direct mention.



5. Discussion

5.1 Broader Implications

Aadhaar has reshaped governance in India, becoming a cornerstone of service delivery and public administration. Its **successes** lie in its ability to enhance **inclusion** and **transparency**. By enabling targeted welfare programs and reducing leakages in subsidies, Aadhaar has streamlined governance processes and made public services more accessible, especially to

marginalized groups. For instance, its integration into welfare schemes and public distribution systems has reduced corruption and improved operational efficiency.

However, Aadhaar's implementation has also highlighted **failures**. Concerns over **privacy breaches**, **biometric data security**, and the exclusion of vulnerable populations are persistent challenges. Issues like authentication errors and data leakage have raised questions about Aadhaar's ability to balance efficiency with equity. The potential for misuse of biometric data underscores the need for robust privacy regulations and governance mechanisms.

For Brazil, Aadhaar offers valuable lessons as it explores digital identity systems. Aadhaar's scalability and integration into diverse policy areas present a model for improving public service delivery. However, Brazil must be cautious to address potential pitfalls, particularly **data privacy**, **legal frameworks**, and ensuring accessibility for all citizens. Learning from Aadhaar's experience, Brazil should prioritize public trust by demonstrating tangible benefits, such as faster access to welfare programs and cost savings, while addressing concerns around exclusion and misuse of data.

5.2 Academic Perspective

The analysis of 15 selected articles reveals **gaps in Aadhaar-related literature**, particularly regarding its **impact on cash transfers** and **immediate benefits** for recipients. While Aadhaar has been instrumental in digitizing welfare delivery, there is limited research on its direct role in enhancing financial inclusion through cash transfers or its ability to provide immediate relief to economically disadvantaged populations.

Furthermore, academic debates highlight Aadhaar's **dual role** as both an enabler of **inclusion** and a potential source of **exclusion**. On one hand, Aadhaar has empowered millions by providing a digital identity that facilitates access to government services. On the other hand, its reliance on biometric authentication can marginalize individuals who face difficulties with fingerprint or iris scans due to age, disability, or occupational factors. These contrasting narratives underscore the complexity of Aadhaar's impact and its broader implications for social equity.

5.3 Sentiment Insights

Sentiment analysis provides valuable insights into public and academic perceptions of Aadhaar's implementation. The prevalence of **neutral sentiment** in the analyzed texts suggests a predominantly descriptive discourse, reflecting a focus on technical and policy-oriented discussions. However, the **polarized views** evident in sentiment trends reveal a divide between Aadhaar's perceived benefits and its challenges.

Positive sentiments, dominated by terms like "efficiency," "value," and "service," highlight Aadhaar's potential to enhance governance efficiency and reduce inefficiencies. Conversely, **negative sentiments**, marked by terms such as "privacy," "risk," and "problems," underscore persistent concerns about data security, exclusion, and ethical implications.

Actionable insights from these findings include the need for stronger **biometric data privacy regulations** and transparent governance practices. Addressing these concerns can mitigate public apprehensions and build trust in Aadhaar-like systems. Additionally, demonstrating **immediate benefits**—such as improved access to welfare and financial inclusion—can strengthen public acceptance and reduce resistance to digital identity frameworks.

By addressing these challenges and leveraging Aadhaar's successes, policymakers in India and other emerging economies, including Brazil, can develop more inclusive, efficient, and secure digital identity systems.

6. Recommendations

6.1 Broader Implications

Aadhaar has transformed governance in India, becoming a cornerstone of public service delivery and administration. Its successes lie in enhancing inclusion, transparency, and operational efficiency. For instance, Aadhaar's integration with the Public Distribution System (PDS) has reduced corruption and leakages, enabling more targeted welfare programs (Khera 2020). Similarly, its use in financial inclusion schemes like Jan Dhan Yojana has facilitated access to banking services for millions (Belorgey 2023).

However, Aadhaar's implementation is not without challenges. Privacy breaches, concerns over biometric data security, and the exclusion of vulnerable populations remain critical issues (Singh 2022). Authentication errors, particularly in rural areas, highlight the system's limitations in balancing efficiency with equity (Johri 2021). The potential misuse of biometric data further underscores the need for robust privacy frameworks, as emphasized by Monahan and Martin (2020).

For Brazil, Aadhaar offers valuable lessons as it considers adopting similar digital identity systems. Brazil's challenges, such as addressing inequalities and improving access to welfare programs, make Aadhaar a compelling case study. However, Brazil must prioritize building public trust by addressing privacy concerns and ensuring the system is accessible to all citizens, particularly marginalized groups. Demonstrating tangible benefits, such as faster access to social programs or cost savings, will be crucial for public acceptance.

6.2 Academic Perspective

The analysis of 15 articles reveals gaps in Aadhaar-related literature, particularly in the context of cash transfers and their immediate benefits. While Aadhaar has digitized welfare delivery, limited research focuses on its direct role in financial inclusion or providing immediate economic relief (Banerjee and Chaudhuri 2016).

Academic debates further highlight Aadhaar's dual role: as a tool for inclusion and a potential source of exclusion. For example, while Aadhaar has empowered millions by providing a digital identity, its reliance on biometric authentication has marginalized individuals unable to meet these technological demands due to age, disability, or occupational factors (Dreze and Khera

2017). This dual narrative underscores the complexity of Aadhaar's impact and raises critical questions about social equity and accessibility.

By addressing these gaps, future research can better inform policymakers on how to design systems that maximize inclusion while mitigating risks. For Brazil, understanding these nuances will be critical to creating a digital identity framework tailored to its socio-economic landscape.

6.3 Sentiment Insights

The sentiment analysis provides valuable insights into public and academic perceptions of Aadhaar. Neutral sentiments dominate, reflecting a predominantly descriptive discourse in the selected articles. This suggests a focus on technical evaluations and policy-oriented discussions rather than overtly positive or negative framing.

However, polarized views are evident, with positive sentiments emphasizing terms like "efficiency," "service," and "value." These reflect Aadhaar's potential to streamline governance and enhance transparency (Varma 2019). On the other hand, negative sentiments highlight concerns about "privacy," "risk," and "problems," underscoring persistent challenges with data security and exclusion (Subramanian 2021).

To address these concerns, policymakers must prioritize biometric data privacy regulations and transparent governance practices. Public apprehensions can be mitigated through clear communication of safeguards and demonstrating immediate benefits, such as improved access to welfare programs and enhanced service delivery. These steps are essential for building trust and reducing resistance to Aadhaar-like systems.

By addressing its challenges and leveraging Aadhaar's successes, Brazil and other emerging economies can develop inclusive, efficient, and secure digital identity systems that balance benefits with safeguards.

7. Conclusion

This study explored Aadhaar's integration into India's governance systems, providing critical insights for emerging economies considering similar digital identity frameworks. By analyzing 15 academic articles and employing advanced text analysis techniques, the research highlighted both the advantages and challenges of Aadhaar's implementation.

Aadhaar has shown its potential to transform governance by enhancing efficiency, enabling transparency, and fostering financial and social inclusion (Abraham, Bennett, and Sen 2019; Subramanian 2021). Positive sentiment associated with Aadhaar often emphasized terms like "data," "service," and "value," underscoring its role in streamlining public service delivery (Belorgey 2023). However, significant challenges remain. Privacy concerns, risks of exclusion, and governance inefficiencies were equally prominent, as reflected in negative sentiments linked to terms like "problem," "risk," and "privacy" (Singh 2022; Khera 2020).

As a case study, Aadhaar demonstrates the need for a balanced approach to digital identity systems. For Brazil and other emerging economies, Aadhaar offers valuable lessons on both the opportunities and pitfalls of such initiatives (Dattani 2019). Policymakers must prioritize robust data privacy frameworks, transparent governance, and scalable infrastructure while showcasing immediate benefits to build trust and foster public acceptance (Monahan and Martin 2020).

The findings also highlight the global relevance of Aadhaar's experience. As more countries explore digital transformation, collaborative efforts between nations like India and Brazil can drive innovation while addressing shared challenges (Borah and Bhuyan 2024). Ongoing research, policy dialogue, and international partnerships will be crucial in developing inclusive, secure, and efficient digital identity systems that serve as models for the future (Varma 2019; Wadhwa 2021).

In conclusion, Aadhaar's successes and limitations underscore the importance of designing digital identity frameworks that balance technological advancement with social equity, ensuring that they empower rather than exclude the populations they aim to serve. This balance will be vital for creating governance systems that are both innovative and inclusive, inspiring global progress in digital identity initiatives.

8. References

1. Abraham, R., P. Bennett, and S. Sen. "Governing with Biometrics: The Aadhaar Experience." *Economic & Political Weekly* 54, no. 15 (2019): 35–40.
2. Banerjee, S., and T. Chaudhuri. "Aadhaar: Identification for the Poor or Poor Identification?" *World Development Report* 16, no. 2 (2016): 14–28.
3. Belorgey, N. "The Aadhaar Battle: Why Some Players in the Corporate World Needed a Biometric ID." *Global Policy* 14, no. 3 (2023): 457–469.
4. Borah, A., and P. Bhuyan. "Living with the Aadhaar: India's Changing Contours of Identity and Governance." *South Asian Journal* 14, no. 2 (2024): 96–112.
5. Dattani, P. "Governpreneurism for Good Governance: The Case of Aadhaar and the India Stack." *Area* 51, no. 4 (2019): 716–723.
6. Dreze, J., and R. Khera. "Aadhaar and Food Security: A Case of Exclusion." *Economic & Political Weekly* 52, no. 50 (2017): 19–21.
7. Ghosh, S. "The Aadhaar Welfare State: Representing the Ideal Biometric Citizen." *Critical Sociology* 48, no. 2 (2022): 295–312.
8. Gupta, A. "Bureaucratic Mediations for Biometric Governance in India's Northeast: Aadhaar in Tripura." *Asian Journal of Public Affairs* 13, no. 3 (2023): 58–74.
9. Johri, K. "The Promise of Inclusive Social Protection? Aadhaar and Welfare in India." *Development Studies* 45, no. 5 (2021): 812–831.
10. Khera, R. "Public Distribution System and Aadhaar: Evaluating the Impact." *Journal of Development Policy* 31, no. 1 (2020): 78–88.

11. Monahan, T., and D. Martin. "Surveillance and Identity in the Age of Aadhaar." *Social Science & Medicine* 250, no. 1 (2020): 112–119.
12. Singh, A. "Aadhaar and Data Privacy: Biometric Identification and Anxieties of Recognition in India." *Journal of Asian Studies* 58, no. 4 (2022): 615–634.
13. Subramanian, A. "Open Platforms and Public Digital Infrastructure: Lessons from Aadhaar." *Public Administration Review* 81, no. 3 (2021): 483–495.
14. Varma, P. "The Social De-Duplicated: Aadhaar and the Engineering of Service." *South Asia Research* 39, no. 1 (2019): 34–51.
15. Wadhwa, K. "Aadhaar and Social Assistance Programming: Local Bureaucracies as Critical Intermediaries." *World Development* 109, no. 1 (2021): 161–172.

9. Appendix: Final Code

```
# -*- coding: utf-8 -*-
"""Merge pdfs

Automatically generated by Colab.

Original file is located at

https://colab.research.google.com/drive/1KoVWLli0zL1iE7hTfQYBe2U4C5_aD3_F
"""

!pip install PyPDF2

from google.colab import drive
drive.mount('/content/drive')

pdf_path = '/content/drive/My Drive/A/Aadhaar.pdf'

!pip install pdfplumber

!pip install gdown

import gdown

file_id = "1pQxos3LwCCtyjzjHPPv8RG_oV0ChpDns"  # Replace with your file's ID
```

```
url =
f"https://drive.google.com/drive/u/0/folders/1pQxos3LwCCtyjzjHPPv8RG_
oV0ChpDns"
output = "Aadhaar.pdf"
gdown.download(url, output, quiet=False)

import pdfplumber

pdf_path = "/content/drive/My Drive/A/Aadhaar.pdf"  # Use the file
downloaded by gdown

with pdfplumber.open(pdf_path) as pdf:
    text = ""
    for page in pdf.pages:
        text += page.extract_text()

print(text[:500])  # Print the first 500 characters

import re

def clean_text(text):
    # Remove non-alphanumeric characters (except spaces)
    text = re.sub(r'[^a-zA-Z0-9\s]', '', text)
    # Convert text to lowercase
    text = text.lower()
    # Remove extra spaces
    text = re.sub(r'\s+', ' ', text).strip()
    return text

cleaned_text = clean_text(text)
print(cleaned_text[:500])  # Print the first 500 characters of
cleaned text

keywords = [
    "aadhaar", "cash transfer", "digital identity", "biometric",
    "UIDAI", "authentication", "welfare", "subsidy", "benefit",
    "policy", "privacy", "inclusion", "exclusion", "ration",
    "fraud", "transparency", "efficiency", "economic", "identity",
    "scheme", "implementation", "pds", "MGNREGA", "Jan Dhan",
"infrastructure"
]

# Count occurrences of each keyword
keyword_counts = {word: cleaned_text.count(word) for word in
keywords}
```

```
print("Keyword Counts:", keyword_counts)

import matplotlib.pyplot as plt

# Bar chart for keyword counts
plt.bar(keyword_counts.keys(), keyword_counts.values(),
color='skyblue')
plt.xticks(rotation=90)
plt.title("Keyword Frequency")
plt.ylabel("Count")
plt.xlabel("Keywords")
plt.show()

# Define a list of relevant keywords
keywords = [
    "aadhaar", "identity", "governance", "privacy", "biometric",
"inclusion",
    "efficiency", "cash transfer", "digital identity", "UIDAI",
"authentication",
    "welfare", "subsidy", "benefit", "policy", "exclusion", "ration",
"fraud",
    "transparency", "economic", "scheme", "implementation", "pds",
"MGNREGA",
    "Jan Dhan", "infrastructure"
]

# Extract sentences containing keywords
sentences_with_keywords = [
    sentence for sentence in text.split('. ')  # Assuming `text`
contains the full merged content
    if any(keyword in sentence.lower() for keyword in keywords)
]

print(f"Number of sentences with keywords:
{len(sentences_with_keywords)}")

print("Sample Sentences with Keywords:")
for sentence in sentences_with_keywords[:5]:  # Display first 5
sentences
    print(sentence)

from collections import Counter

# Count the occurrence of keywords in the extracted sentences
keyword_frequency = Counter(
```

```
    keyword for sentence in sentences_with_keywords
    for keyword in keywords if keyword in sentence.lower()
)

print("Keyword Frequency:", keyword_frequency)

# Plot
import matplotlib.pyplot as plt

plt.bar(keyword_frequency.keys(), keyword_frequency.values(),
color='skyblue')
plt.xticks(rotation=90)
plt.title("Keyword Frequency in Sentences with Keywords")
plt.ylabel("Count")
plt.xlabel("Keywords")
plt.show()

import re
from nltk.corpus import stopwords
import nltk
nltk.download('stopwords')

def clean_text(text):
    stop_words = set(stopwords.words('english'))
    # Add Aadhaar as a custom stopword
    stop_words.add('aadhaar')
    # Remove non-alphanumeric characters
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    # Convert text to lowercase
    text = text.lower()
    # Remove stopwords
    words = text.split()
    filtered_words = [word for word in words if word not in
stop_words]
    return " ".join(filtered_words)

cleaned_text = clean_text(text)
print("Text cleaning complete.")

import nltk
from nltk.sentiment import SentimentIntensityAnalyzer

# Download the VADER lexicon
nltk.download('vader_lexicon')
```

```
# Initialize the sentiment analyzer
sia = SentimentIntensityAnalyzer()

print("VADER Sentiment Intensity Analyzer initialized successfully.")

from nltk.sentiment import SentimentIntensityAnalyzer

# Initialize the sentiment analyzer
sia = SentimentIntensityAnalyzer()

# Split the cleaned text into sentences
sentences = text.split('. ')
print(f"Number of sentences: {len(sentences)}")

# Analyze sentiment for each sentence
sentiment_scores = [
    {"sentence": sentence, "sentiment":
sia.polarity_scores(sentence)["compound"]}
    for sentence in sentences
]

# Categorize sentences by sentiment
positive_sentences = [entry["sentence"] for entry in sentiment_scores
if entry["sentiment"] > 0.05]
neutral_sentences = [entry["sentence"] for entry in sentiment_scores
if -0.05 <= entry["sentiment"] <= 0.05]
negative_sentences = [entry["sentence"] for entry in sentiment_scores
if entry["sentiment"] < -0.05]

print(f"Positive: {len(positive_sentences)}, Neutral:
{len(neutral_sentences)}, Negative: {len(negative_sentences)}")

import matplotlib.pyplot as plt

# Bar chart data
sentiment_counts = [len(positive_sentences), len(neutral_sentences),
len(negative_sentences)]
sentiment_labels = ['Positive', 'Neutral', 'Negative']

# Plot the bar chart
plt.bar(sentiment_labels, sentiment_counts, color=['green', 'blue',
'red'])
plt.title("Sentiment Distribution")
plt.ylabel("Number of Sentences")
plt.show()
```

```
from wordcloud import WordCloud

# Generate Positive Word Cloud
positive_text = " ".join(positive_sentences)
positive_wordcloud = WordCloud(width=800, height=400,
background_color="white").generate(positive_text)

plt.figure(figsize=(10, 5))
plt.imshow(positive_wordcloud, interpolation="bilinear")
plt.axis("off")
plt.title("Positive Sentiment Word Cloud")
plt.show()

# Generate Negative Word Cloud
negative_text = " ".join(negative_sentences)
negative_wordcloud = WordCloud(width=800, height=400,
background_color="white").generate(negative_text)

plt.figure(figsize=(10, 5))
plt.imshow(negative_wordcloud, interpolation="bilinear")
plt.axis("off")
plt.title("Negative Sentiment Word Cloud")
plt.show()

# Sort by sentiment score for samples
positive_sentences_sorted = sorted(
    [entry for entry in sentiment_scores if entry["sentiment"] >
0.05],
    key=lambda x: x["sentiment"],
    reverse=True
)
negative_sentences_sorted = sorted(
    [entry for entry in sentiment_scores if entry["sentiment"] <
-0.05],
    key=lambda x: x["sentiment"]
)

# Display samples
print("Sample Positive Sentences:")
for entry in positive_sentences_sorted[:5]:  # Top 5 positive
    print(entry["sentence"])
```

```
print("\nSample Negative Sentences:")
for entry in negative_sentences_sorted[:5]:  # Top 5 negative
    print(entry["sentence"])

from wordcloud import STOPWORDS

# Add custom stopwords
custom_stopwords = STOPWORDS.union({"aadhaar", "government"})

print(f"Custom stopwords: {custom_stopwords}")



# Join positive and negative sentences into a single string
positive_sentences_combined = " ".join(positive_sentences)  # Convert
list to string
negative_sentences_combined = " ".join(negative_sentences)  # Convert
list to string

# Generate the word cloud for positive sentiment
positive_wordcloud = WordCloud(
    width=800,
    height=400,
    background_color='white',
    stopwords=custom_stopwords
).generate(positive_sentences_combined)

# Generate the word cloud for negative sentiment
negative_wordcloud = WordCloud(
    width=800,
    height=400,
    background_color='white',
    stopwords=custom_stopwords
).generate(negative_sentences_combined)

# Plot the positive sentiment word cloud
plt.figure(figsize=(10, 5))
plt.imshow(positive_wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title("Positive Sentiment Word Cloud (Excluding 'Aadhaar')")
plt.show()

# Plot the negative sentiment word cloud
plt.figure(figsize=(10, 5))
plt.imshow(negative_wordcloud, interpolation='bilinear')
```

```
plt.axis("off")
plt.title("Negative Sentiment Word Cloud (Excluding 'Aadhaar')")
plt.show()
```

10. Appendix: Preliminary Sentiment and Keyword Analysis Code

```
# -*- coding: utf-8 -*-
"""MPV.ipynb

Automatically generated by Colab.

Original file is located at

https://colab.research.google.com/drive/1iYav2C7uqGyAaa30QVx10bdHmfM7
-S08
"""

!pip install PyPDF2

from google.colab import drive
drive.mount('/content/drive')

pdf_path = '/content/drive/My Drive/A/Aadhaar.pdf'

!pip install pdfplumber

!pip install gdown

import gdown

file_id = "1pQxos3LwCCtyjzjHPPv8RG_oV0ChpDns"  # Replace with your
file's ID
url =
f"https://drive.google.com/drive/u/0/folders/1pQxos3LwCCtyjzjHPPv8RG_
oV0ChpDns"
output = "Aadhaar.pdf"
gdown.download(url, output, quiet=False)

import pdfplumber
```

```
pdf_path = "/content/drive/My Drive/A/Aadhaar.pdf"  # Use the file
downloaded by gdown

with pdfplumber.open(pdf_path) as pdf:
    text = ""
    for page in pdf.pages:
        text += page.extract_text()

print(text[:500])  # Print the first 500 characters

import re

def clean_text(text):
    # Remove non-alphanumeric characters (except spaces)
    text = re.sub(r'[^a-zA-Z0-9\s]', '', text)
    # Convert text to lowercase
    text = text.lower()
    # Remove extra spaces
    text = re.sub(r'\s+', ' ', text).strip()
    return text

cleaned_text = clean_text(text)
print(cleaned_text[:500])  # Print the first 500 characters of
cleaned text

keywords = [
    "aadhaar", "cash transfer", "digital identity", "biometric",
    "UIDAI", "authentication", "welfare", "subsidy", "benefit",
"policy"
]

# Count occurrences of each keyword
keyword_counts = {word: cleaned_text.count(word) for word in
keywords}
print("Keyword Counts:", keyword_counts)

import matplotlib.pyplot as plt

plt.bar(keyword_counts.keys(), keyword_counts.values(),
color='skyblue')
plt.xlabel("Keywords")
plt.ylabel("Frequency")
plt.title("Keyword Frequency in Aadhaar Text")
plt.xticks(rotation=45)
plt.show()
```

```
# Define a list of relevant keywords
keywords = ["aadhaar", "identity", "governance", "privacy",
"biometric", "inclusion", "efficiency"]

# Extract sentences containing keywords
sentences_with_keywords = [
    sentence for sentence in text.split('. ')  # Assuming `text`
contains the full merged content
    if any(keyword in sentence.lower() for keyword in keywords)
]

print(f"Number of sentences with keywords:
{len(sentences_with_keywords)}")

from nltk.sentiment import SentimentIntensityAnalyzer
import nltk

nltk.download('vader_lexicon')

sia = SentimentIntensityAnalyzer()

# Analyze sentiment for the extracted sentences
for sentence in sentences_with_keywords[:10]:  # Analyze the first 10
sentences
    sentiment = sia.polarity_scores(sentence)
    print(f"Sentence: {sentence}")
    print(f"Sentiment: {sentiment}")

# Initialize sentiment score counters
sentiment_scores = {"positive": 0, "neutral": 0, "negative": 0}

# Analyze sentiment for each sentence
for sentence in sentences_with_keywords:
    sentiment = sia.polarity_scores(sentence)
    if sentiment["compound"] > 0.05:
        sentiment_scores["positive"] += 1
    elif sentiment["compound"] < -0.05:
        sentiment_scores["negative"] += 1
    else:
        sentiment_scores["neutral"] += 1

# Print sentiment counts
print("Sentiment Scores:", sentiment_scores)
```

```
import matplotlib.pyplot as plt

# Sentiment labels and values
labels = sentiment_scores.keys()
values = sentiment_scores.values()

# Create a pie chart
plt.figure(figsize=(8, 8))
plt.pie(values, labels=labels, autopct='%1.1f%%', startangle=140,
colors=["lightgreen", "gold", "salmon"])
plt.title("Sentiment Distribution for Aadhaar Text")
plt.show()

# Split text into manageable chunks
chunk_size = 100000  # Each chunk has 100,000 characters
chunks = [text[i:i + chunk_size] for i in range(0, len(text),
chunk_size)]

# Process each chunk with spaCy
docs = [nlp(chunk) for chunk in chunks]

# Combine sentences from all chunks
all_sentences = []
for doc in docs:
    all_sentences.extend([sent for sent in doc.sents])

# Example cleaned text (replace this with your actual text)
text = """
Aadhaar has revolutionized identity verification in India. However,
it has faced criticism for privacy concerns. Some argue that it has
improved service delivery, while others highlight exclusion risks due
to biometric mismatches.
"""

import spacy
from nltk.sentiment import SentimentIntensityAnalyzer

# Load spaCy model
nlp = spacy.load("en_core_web_sm")

# Increase spaCy's maximum length (if needed)
nlp.max_length = len(text) + 1000

# Process the text
doc = nlp(text)
```

```
sia = SentimentIntensityAnalyzer()

# Extract sentences with negative sentiment
negative_sentences = [sentence for sentence in doc.sents if
sia.polarity_scores(sentence.text)["compound"] < -0.1]

# Print sample sentences
print("Sample Negative Sentences:")
for sent in negative_sentences[:5]:  # Show the first 5 examples
    print(sent.text)

# Collect sentiment compound scores
sentiment_trend = [sia.polarity_scores(sentence)["compound"] for
sentence in sentences_with_keywords]

# Plot sentiment trend
plt.figure(figsize=(10, 6))
plt.plot(sentiment_trend, marker='o', linestyle='-', color='blue')
plt.axhline(y=0.05, color='green', linestyle='--', label='Positive
Threshold')
plt.axhline(y=-0.05, color='red', linestyle='--', label='Negative
Threshold')
plt.title("Sentiment Trend Across Sentences")
plt.xlabel("Sentence Index")
plt.ylabel("Compound Sentiment Score")
plt.legend()
plt.show()

# Split text into manageable chunks
chunk_size = 100000  # Each chunk has 100,000 characters
chunks = [cleaned_text[i:i + chunk_size] for i in range(0,
len(cleaned_text), chunk_size)]

# Initialize list to store all sentiment scores
sentiment_scores = []

# Process each chunk and compute sentiment scores
for chunk in chunks:
    doc = nlp(chunk)

sentiment_scores.extend([sia.polarity_scores(sent.text)["compound"]
for sent in doc.sents])

print("Sentiment scores calculated for all sentences in chunks.")
```

```
from nltk.sentiment import SentimentIntensityAnalyzer
import spacy

# Load spaCy and Sentiment Analyzer
nlp = spacy.load("en_core_web_sm")
sia = SentimentIntensityAnalyzer()

# Process the text
doc = nlp(cleaned_text)

# Compute sentiment scores for each sentence
sentiment_scores = [sia.polarity_scores(sent.text)["compound"] for
sent in doc.sents]

# Print sentiment scores
print("Sentiment scores calculated for each sentence.")



# Extract and print highly positive sentences
print("Highly Positive Sentences:")
for i, sent in enumerate(doc.sents):
    if sentiment_scores[i] > 0.75:  # Threshold for highly positive
        print(f"Sentence {i}: {sent.text}")

# Extract and print highly negative sentences
print("\nHighly Negative Sentences:")
for i, sent in enumerate(doc.sents):
    if sentiment_scores[i] < -0.75:  # Threshold for highly negative
        print(f"Sentence {i}: {sent.text}")

from wordcloud import WordCloud

# Separate sentences into positive and negative
positive_sentences = " ".join([sentence for sentence in
sentences_with_keywords if sia.polarity_scores(sentence)["compound"]
> 0.05])
negative_sentences = " ".join([sentence for sentence in
sentences_with_keywords if sia.polarity_scores(sentence)["compound"]
< -0.05])

# Generate word clouds
positive_wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(positive_sentences)
```

```
negative_wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(negative_sentences)

# Plot word clouds
plt.figure(figsize=(10, 5))
plt.imshow(positive_wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title("Positive Sentiment Word Cloud")
plt.show()

plt.figure(figsize=(10, 5))
plt.imshow(negative_wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title("Negative Sentiment Word Cloud")
plt.show()

import nltk
from nltk.sentiment import SentimentIntensityAnalyzer
from nltk.corpus import stopwords
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Download required resources
nltk.download('vader_lexicon')
nltk.download('stopwords')

# Define keywords
keywords = ['aadhaar', 'government', 'card', 'pds', 'ration',
'savings', 'fraud', 'khera', 'mgnrega', 'data']

# Extract sentences containing keywords
sentences_with_keywords = [sentence for sentence in text.split('. ')
if any(keyword in sentence.lower() for keyword in keywords)]

# Initialize Sentiment Intensity Analyzer
sia = SentimentIntensityAnalyzer()

# Separate sentences by sentiment
positive_sentences = [sentence for sentence in
sentences_with_keywords if sia.polarity_scores(sentence)["compound"]
> 0.05]
negative_sentences = [sentence for sentence in
sentences_with_keywords if sia.polarity_scores(sentence)["compound"]
< -0.05]
```

```
import nltk

# Download stopwords
nltk.download('stopwords')

import nltk
from wordcloud import WordCloud
import matplotlib.pyplot as plt
from nltk.corpus import stopwords
import spacy
from nltk.sentiment import SentimentIntensityAnalyzer

# Download NLTK resources
nltk.download('stopwords')
nltk.download('vader_lexicon')

# Load stopwords and add custom word
stop_words = set(stopwords.words('english'))
stop_words.add('aadhaar')  # Add 'aadhaar' as a stopword

# Load spaCy model
nlp = spacy.load("en_core_web_sm")

# Example cleaned text (Replace with your actual cleaned text)
cleaned_text = """Your cleaned Aadhaar-related text goes here."""

# Sentiment analysis using VADER
sia = SentimentIntensityAnalyzer()

positive_sentences = []
negative_sentences = []

for sentence in cleaned_text.split('.'):  # Split by period to get
sentences
    sentiment = sia.polarity_scores(sentence)
    if sentiment['compound'] > 0.1:  # Positive sentiment
        positive_sentences.append(sentence)
    elif sentiment['compound'] < -0.1:  # Negative sentiment
        negative_sentences.append(sentence)

# Debugging: Check content of positive and negative sentences
print("Positive Sentences:", positive_sentences[:3])  # Print first 3
positive sentences
print("Negative Sentences:", negative_sentences[:3])  # Print first 3
negative sentences
```

```
# Preprocess sentences
def preprocess_text(sentences):
    """Tokenize, remove stopwords, and return cleaned words."""
    text = " ".join(sentences)  # Combine sentences
    doc = nlp(text)
    tokens = [token.text.lower() for token in doc if token.is_alpha
and token.text.lower() not in stop_words]
    return tokens

# Generate text for word clouds
positive_tokens = preprocess_text(positive_sentences)
negative_tokens = preprocess_text(negative_sentences)

# Check if tokens exist
if not positive_tokens:
    print("No valid positive tokens found!")
    positive_text = "no positive words"
else:
    positive_text = " ".join(positive_tokens)

if not negative_tokens:
    print("No valid negative tokens found!")
    negative_text = "no negative words"
else:
    negative_text = " ".join(negative_tokens)

# Generate Word Clouds
def generate_wordcloud(text, title):
    """Generate and display a word cloud."""
    wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(text)
    plt.figure(figsize=(10, 5))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off")
    plt.title(title, fontsize=15)
    plt.show()

# Positive Word Cloud
generate_wordcloud(positive_text, "Positive Sentiment Word Cloud
(Excluding 'Aadhaar')")

# Negative Word Cloud
generate_wordcloud(negative_text, "Negative Sentiment Word Cloud
(Excluding 'Aadhaar')")
```

```
# Initialize average sentiment scores
average_scores = {"positive": 0, "neutral": 0, "negative": 0}

# Sum up sentiment scores
for sentence in sentences_with_keywords:
    sentiment = sia.polarity_scores(sentence)
    average_scores["positive"] += sentiment["pos"]
    average_scores["neutral"] += sentiment["neu"]
    average_scores["negative"] += sentiment["neg"]

# Calculate averages
total_sentences = len(sentences_with_keywords)
for key in average_scores:
    average_scores[key] /= total_sentences

# Plot the bar chart
plt.bar(average_scores.keys(), average_scores.values(),
color=["lightgreen", "gold", "salmon"])
plt.title("Average Sentiment Scores for Aadhaar Text")
plt.xlabel("Sentiment")
plt.ylabel("Average Score")
plt.show()

from collections import Counter

# Tokenize and count words
tokens = cleaned_text.split()
word_frequencies = Counter(tokens)

# Display the top 20 most common words
most_common_words = word_frequencies.most_common(20)
print("Most Common Words:", most_common_words)

from wordcloud import WordCloud

wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(cleaned_text)

# Display the word cloud
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title("Word Cloud of Aadhaar Text")
plt.show()
```

```
import matplotlib.pyplot as plt
from collections import Counter

# Tokenize and count word frequencies
tokens = cleaned_text.split()
word_frequencies = Counter(tokens)

# Get the top 20 most common words
most_common_words = word_frequencies.most_common(20)
words, frequencies = zip(*most_common_words)

# Plot the bar chart
plt.figure(figsize=(12, 6))
plt.bar(words, frequencies, color='skyblue')
plt.title("Top 20 Most Frequent Words")
plt.xlabel("Words")
plt.ylabel("Frequency")
plt.xticks(rotation=45)
plt.show()

# Horizontal bar chart for word frequency
plt.figure(figsize=(12, 6))
plt.barh(words, frequencies, color='salmon')
plt.title("Top 20 Most Frequent Words (Horizontal)")
plt.xlabel("Frequency")
plt.ylabel("Words")
plt.gca().invert_yaxis()  # Invert y-axis to have the highest
frequency on top
plt.show()

from wordcloud import WordCloud

# Generate the word cloud
wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(cleaned_text)

# Display the word cloud
plt.figure(figsize=(12, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title("Word Cloud of Text")
plt.show()

import numpy as np
```

```
# Get cumulative frequencies
cumulative_frequencies = np.cumsum([freq for _, freq in
most_common_words])

# Plot the cumulative frequency
plt.figure(figsize=(12, 6))
plt.plot(range(1, len(cumulative_frequencies) + 1),
cumulative_frequencies, marker='o', color='blue')
plt.title("Cumulative Word Frequency")
plt.xlabel("Rank of Word")
plt.ylabel("Cumulative Frequency")
plt.grid()
plt.show()

# Prepare data for log-log plot
ranks = range(1, len(word_frequencies) + 1)
frequencies = [freq for _, freq in word_frequencies.most_common()]

# Log-log plot
plt.figure(figsize=(12, 6))
plt.loglog(ranks, frequencies, marker="o", color="purple")
plt.title("Log-Log Plot of Word Frequencies")
plt.xlabel("Rank")
plt.ylabel("Frequency")
plt.grid()
plt.show()

# Count word lengths
word_lengths = [len(word) for word in tokens]
length_counts = Counter(word_lengths)

# Plot word length frequency
plt.figure(figsize=(12, 6))
plt.bar(length_counts.keys(), length_counts.values(), color='green')
plt.title("Word Length Distribution")
plt.xlabel("Word Length")
plt.ylabel("Frequency")
plt.xticks(range(1, max(word_lengths) + 1))
plt.show()

from gensim import corpora
from gensim.models import LdaModel

# Prepare data for LDA
```

```
tokens_filtered = [token for token in tokens if token not in
keywords]  # Exclude keywords to focus on broader topics
dictionary = corpora.Dictionary([tokens_filtered])
corpus = [dictionary.doc2bow(tokens_filtered)]

# Train LDA model
lda_model = LdaModel(corpus, num_topics=3, id2word=dictionary,
passes=10)

# Display topics
for idx, topic in lda_model.print_topics(-1):
    print(f"Topic {idx + 1}: {topic}")

# Print word distributions for topics
for idx, topic in lda_model.print_topics(-1):
    print(f"Topic {idx + 1}: {topic}")

# Visualize the word distribution for each topic
import matplotlib.pyplot as plt

num_topics = 3  # Adjust based on the number of topics in your model
for i in range(num_topics):
    topic_terms = lda_model.show_topic(i, topn=10)  # Top 10 words
for each topic
    words, weights = zip(*topic_terms)

    plt.figure(figsize=(10, 5))
    plt.bar(words, weights, color='skyblue')
    plt.title(f"Topic {i + 1} - Word Distribution")
    plt.xlabel("Words")
    plt.ylabel("Weight")
    plt.xticks(rotation=45)
    plt.show()

import spacy

# Load spaCy's English model
nlp = spacy.load("en_core_web_sm")

# Create a list of tokens excluding stop words
tokens_no_stopwords = [token.text for token in nlp(cleaned_text) if
not token.is_stop and token.is_alpha]

# Check the first 20 tokens after stop word removal
print(tokens_no_stopwords[:20])
```

```
from gensim import corpora
from gensim.models import LdaModel

# Create a dictionary and corpus from the filtered tokens
dictionary = corpora.Dictionary([tokens_no_stopwords])
corpus = [dictionary.doc2bow(tokens_no_stopwords)]

# Train the LDA model
lda_model = LdaModel(corpus, num_topics=3, id2word=dictionary,
passes=10)

# Display topics
for idx, topic in lda_model.print_topics(-1):
    print(f"Topic {idx + 1}: {topic}")

import matplotlib.pyplot as plt

# Visualize word distributions for each topic
for i in range(3):  # Adjust based on the number of topics
    topic_terms = lda_model.show_topic(i, topn=10)
    words, weights = zip(*topic_terms)

    plt.figure(figsize=(10, 5))
    plt.bar(words, weights, color='skyblue')
    plt.title(f"Topic {i + 1} - Word Distribution Without Stop
Words")
    plt.xlabel("Words")
    plt.ylabel("Weight")
    plt.xticks(rotation=45)
    plt.show()

from wordcloud import WordCloud

for i in range(3):  # Adjust based on the number of topics
    topic_terms = lda_model.show_topic(i, topn=50)
    topic_dict = {word: weight for word, weight in topic_terms}
    wordcloud = WordCloud(width=800, height=400,
background_color='white').generate_from_frequencies(topic_dict)

    plt.figure(figsize=(10, 5))
    plt.imshow(wordcloud, interpolation="bilinear")
    plt.axis("off")
    plt.title(f"Word Cloud for Topic {i + 1} Without Stop Words")
    plt.show()
```

```
from collections import Counter

# Count word frequencies
word_frequencies = Counter(tokens_no_stopwords)

# Get the top 20 most common words
most_common_words = word_frequencies.most_common(20)
words, frequencies = zip(*most_common_words)

print("Top 20 Words After Removing Stop Words:")
print(most_common_words)

import matplotlib.pyplot as plt

# Bar chart for word frequency
plt.figure(figsize=(12, 6))
plt.bar(words, frequencies, color='skyblue')
plt.title("Top 20 Most Frequent Words (After Stop Word Removal)")
plt.xlabel("Words")
plt.ylabel("Frequency")
plt.xticks(rotation=45)
plt.show()

import spacy

# Load spaCy's English model
nlp = spacy.load("en_core_web_sm")

# Clean the text and tokenize without stop words
tokens_no_stopwords = [token.text for token in nlp(cleaned_text) if
not token.is_stop and token.is_alpha]

# Check the first 20 tokens
print("Sample tokens after stop word removal:",
tokens_no_stopwords[:20])

from gensim import corpora

# Create the dictionary and corpus
dictionary = corpora.Dictionary([tokens_no_stopwords])
corpus = [dictionary.doc2bow(tokens_no_stopwords)]

# Check the number of unique words in the dictionary
print(f"Number of unique words in dictionary: {len(dictionary)}")
```

```
from gensim.models import LdaModel

# Train the LDA model
num_topics = 5  # Number of topics
lda_model = LdaModel(corpus, num_topics=num_topics,
id2word=dictionary, passes=10)

# Print topics
for idx, topic in lda_model.print_topics(-1):
    print(f"Topic {idx + 1}: {topic}")

import matplotlib.pyplot as plt

# Plot word distributions for each topic
topn_words = 10  # Top 10 words for each topic

for i in range(num_topics):
    topic_terms = lda_model.show_topic(i, topn=topn_words)
    words, weights = zip(*topic_terms)

    plt.figure(figsize=(10, 5))
    plt.bar(words, weights, color='skyblue')
    plt.title(f"Topic {i + 1} - Top {topn_words} Words")
    plt.xlabel("Words")
    plt.ylabel("Weight")
    plt.xticks(rotation=45)
    plt.show()

num_topics = 5  # Adjust the number of topics
topn_words = 15  # Number of top words per topic

for i in range(num_topics):
    topic_terms = lda_model.show_topic(i, topn=topn_words)
    words, weights = zip(*topic_terms)

    plt.figure(figsize=(10, 5))
    plt.bar(words, weights, color='lightblue')
    plt.title(f"Topic {i + 1} - Top {topn_words} Words")
    plt.xlabel("Words")
    plt.ylabel("Weight")
    plt.xticks(rotation=45)
    plt.show()

from wordcloud import WordCloud
```

```
num_topics = 5  # Adjust the number of topics

for i in range(num_topics):
    topic_terms = lda_model.show_topic(i, topn=50)  # Top 50 words
per topic
    topic_dict = {word: weight for word, weight in topic_terms}

    wordcloud = WordCloud(width=800, height=400,
background_color="white").generate_from_frequencies(topic_dict)

    plt.figure(figsize=(10, 5))
    plt.imshow(wordcloud, interpolation="bilinear")
    plt.axis("off")
    plt.title(f"Word Cloud for Topic {i + 1}")
    plt.show()

import seaborn as sns
import pandas as pd

# Prepare the data
topic_word_data = {f"Topic {i + 1}": dict(lda_model.show_topic(i,
topn=10)) for i in range(num_topics)}
df_topic_word = pd.DataFrame(topic_word_data).fillna(0)

# Plot the heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(df_topic_word, annot=True, cmap="YlGnBu", cbar=True)
plt.title("Heatmap of Word Contributions to Topics")
plt.xlabel("Topics")
plt.ylabel("Words")
plt.show()

!pip install pyLDAvis

import pyLDAvis
import pyLDAvis.gensim_models as gensimvis

# Prepare the pyLDAvis visualization
lda_vis = gensimvis.prepare(lda_model, corpus, dictionary)

# Display the visualization
pyLDAvis.display(lda_vis)

!pip install spacy
```

```
!python -m spacy download en_core_web_sm

import spacy

# Load spaCy's small English model
nlp = spacy.load("en_core_web_sm")

# Process the cleaned text
doc = nlp(cleaned_text)

# Extract named entities
print("Named Entities, their labels, and positions:")
for ent in doc.ents:
    print(f"{ent.text} ({ent.label_}) - Start: {ent.start_char}, End:
{ent.end_char}")

from spacy import displacy

# Render NER visualization
displacy.render(doc, style="ent", jupyter=True)

from collections import Counter

# Count entity types
entity_counts = Counter([ent.label_ for ent in doc.ents])

# Print the counts
print("Entity Type Counts:")
for entity, count in entity_counts.items():
    print(f"{entity}: {count}")

import matplotlib.pyplot as plt

# Plot entity counts
plt.figure(figsize=(10, 6))
plt.bar(entity_counts.keys(), entity_counts.values(),
color="lightblue")
plt.title("Named Entity Types in Text")
plt.xlabel("Entity Type")
plt.ylabel("Frequency")
plt.xticks(rotation=45)
plt.show()

# Extract specific entities
org_entities = [ent.text for ent in doc.ents if ent.label_ == "ORG"]
```

```
gpe_entities = [ent.text for ent in doc.ents if ent.label_ == "GPE"]

# Display results
print("Organizations Mentioned:")
print(set(org_entities))

print("\nLocations Mentioned:")
print(set(gpe_entities))

# Extract sentences mentioning specific keywords
keywords = ["aadhaar", "cash transfer", "policy", "digital identity"]

print("Sentences Containing Keywords:")
for sent in doc.sents:
    if any(keyword in sent.text.lower() for keyword in keywords):
        print(sent.text)
```