5.CrewAi-qdrantimptopics-Internet-Acesss

June 17, 2024

[1]:
```
import os
```

[2]:
```
%pwd
```

[2]:
```
'/mnt/d/Desktop/SuperteamsAI/News aggregator/Research'
```

[3]:
```
os.chdir("../")
```

[4]:
```
%pwd
```

[4]:
```
'/mnt/d/Desktop/SuperteamsAI/News aggregator'
```

[5]:
```
import logging
from pathlib import Path
logging.basicConfig(
    # filename='extract_data.log',
    level=logging.INFO,
    format='%(asctime)s - %(levelname)s - %(message)s',
    datefmt='%Y-%m-%d %H:%M:%S'
)
```

[6]:
```
# from crewai import Agent, Task, Crew
# from langchain_openai import ChatOpenAI
# import os
# os.environ["OPENAI_API_KEY"] = "NA"

# llm = ChatOpenAI(
#     model = "crewai-llama3",
#     base_url = "http://localhost:11434/v1")
```

1 Content Planner Agent & Create planner task

[7]:
```
# # Mock LLM class to simulate the planner agent's behavior
# class MockLLM:
#     def bind(self, *args, **kwargs):
#         def call(inputs):
```

```
#             return plan_content(inputs)
#         return call
```

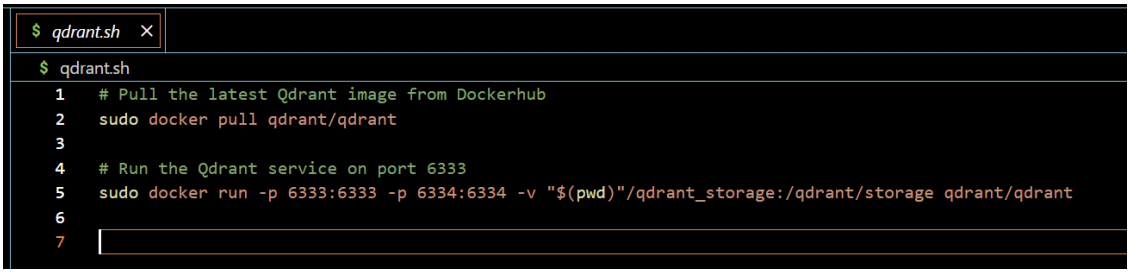
2 Loading embeddding Model

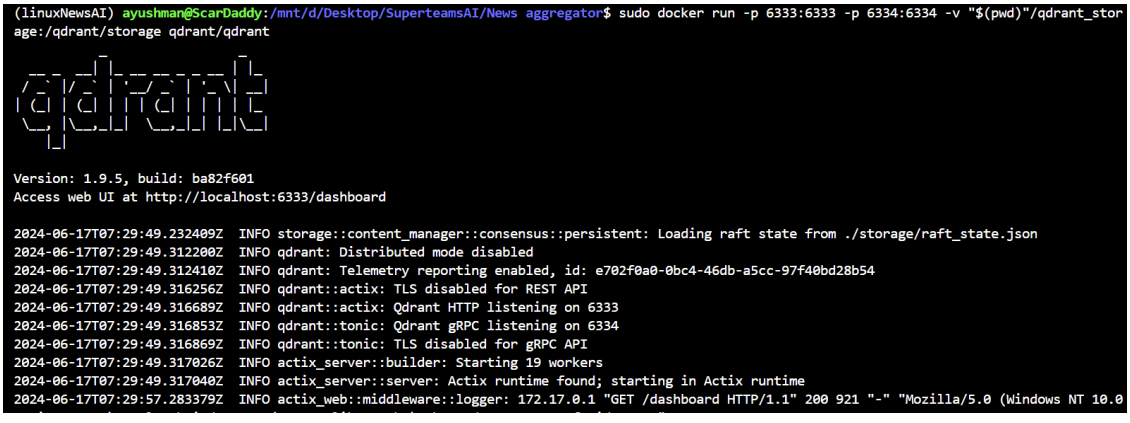
[8]:
```
from sentence_transformers import SentenceTransformer
```

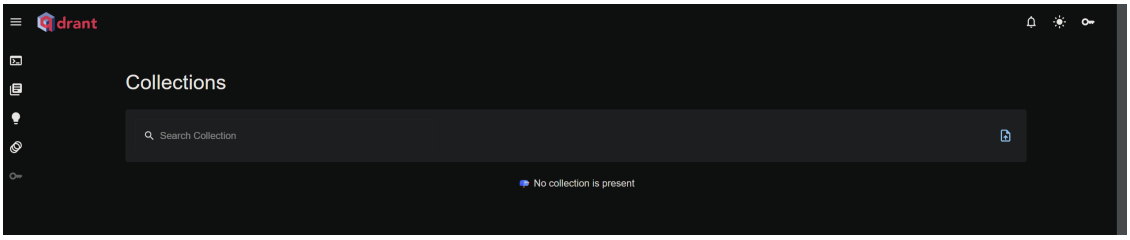
```
/mnt/d/Desktop/SuperteamsAI/News aggregator/linuxNewsAI/lib/python3.10/site-
packages/sentence_transformers/cross_encoder/CrossEncoder.py:11: TqdmWarning:
IProgress not found. Please update jupyter and ipywidgets. See
https://ipywidgets.readthedocs.io/en/stable/user_install.html
  from tqdm.autonotebook import tqdm, trange
```

[9]:
```
# from transformers import AutoTokenizer, AutoModel
# from pathlib import Path

# def download_model_and_tokenizer(model_name, save_path):
#     """
#     Download and save both the model and the tokenizer to the specified␣
 ↪directory.

#     Parameters:
#         model_name (str): Name of the model to download.
#         save_path (str or Path): Path to the directory where the model and␣
 ↪tokenizer will be saved.
#     """
#     # Create the save path if it doesn't exist
#     save_path = Path(save_path)
#     save_path.mkdir(parents=True, exist_ok=True)

#     # Initialize tokenizer and model
#     tokenizer = AutoTokenizer.from_pretrained(model_name)
#     model = AutoModel.from_pretrained(model_name)

#     # Save tokenizer
#     tokenizer.save_pretrained(save_path)

#     # Save model
#     model.save_pretrained(save_path)

# # Example usage
# model_name = 'sentence-transformers/all-MiniLM-L12-v2'  # Model name to␣
 ↪download
# save_path = Path("MiniLM-L12-v2/")  # Path where model and tokenizer will be␣
 ↪saved
# download_model_and_tokenizer(model_name, save_path)
```

[10]:
```python
from transformers import AutoTokenizer, AutoModel

def load_model_and_tokenizer(model_path):
    """
    Load the model and tokenizer from the specified directory.

    Parameters:
        model_path (str or Path): Path to the directory containing the saved model and tokenizer.

    Returns:
        tokenizer (transformers.PreTrainedTokenizer): Loaded tokenizer.
        model (transformers.PreTrainedModel): Loaded model.
    """
    model_path = Path(model_path)
    tokenizer = AutoTokenizer.from_pretrained(model_path)
    model = AutoModel.from_pretrained(model_path)
    return tokenizer, model

# # Load the model and tokenizer
# model_path = Path("MiniLM-L12-v2/")
# tokenizer, model = load_model_and_tokenizer(model_path)
```







[12]:
```
import os
import logging
import time
import ast
import torch
import pandas as pd
from pathlib import Path
from crewai import Agent, Task, Crew
from langchain_openai import ChatOpenAI
import requests
from bs4 import BeautifulSoup
from newspaper import Article
from transformers import AutoTokenizer, AutoModel
from sentence_transformers import SentenceTransformer
from qdrant_client import QdrantClient
from qdrant_client.http.models import Distance, VectorParams, PointStruct
from langchain.vectorstores import Qdrant

# Set up the OpenAI API key
os.environ["OPENAI_API_KEY"] = "NA"

# Set up the LLM for writer and editor
llm = ChatOpenAI(
    model="crewai-llama3",
    base_url="http://localhost:11434/v1"
)

# Function to perform a Google search and return search results
def google_search(query):
    search_url = f"https://www.google.com/search?q={query}"
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
 ↪537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
    }
    search_results = []
    max_retries = 3
    retry_delay = 2

    retries = 0
```

```
    while retries < max_retries:
        try:
            response = requests.get(search_url, headers=headers)
            response.raise_for_status()  # Raise an HTTPError for bad responses

            soup = BeautifulSoup(response.text, 'html.parser')

            for item in soup.select('div.g'):
                title = item.select_one('h3')
                link = item.select_one('a')['href']
                if title and link:
                    search_results.append({
                        "title": title.get_text(),
                        "link": link
                    })
            break
        except requests.exceptions.HTTPError as e:
            if e.response.status_code == 429:  # Too Many Requests
                retries += 1
                print(f"Rate limit hit. Waiting for {retry_delay} seconds␣
 ↪before retrying... (Attempt {retries}/{max_retries})")
                time.sleep(retry_delay)
                retry_delay *= 2  # Exponential backoff
            else:
                raise e

    if retries == max_retries:
        print("Max retries reached. Failed to retrieve search results.")

    return search_results

# Function to extract article attributes
def extract_article_attributes(url):
    try:
        article = Article(url)
        article.download()
        article.parse()
        return {
            'authors': article.authors,
            'text': article.text,
            'title': article.title,
            'link': url
        }
    except Exception as e:
        print(f"Failed to process {url}: {e}")
        return None
```

```
# Function to perform a search, store results in a DataFrame, and extract␣
 ↪article attributes
def search_and_store_to_dataframe(query, filename=None):
    results = google_search(query)
    articles = [extract_article_attributes(result["link"]) for result in␣
 ↪results]
    articles = [article for article in articles if article is not None]  #␣
 ↪Filter out failed downloads
    df = pd.DataFrame(articles)

    # Drop rows with any NaN values
    df_cleaned = df.dropna()

    if filename:
        df_cleaned.to_csv(filename, index=False)  # Save cleaned DataFrame with␣
 ↪attributes to CSV file

    print(df_cleaned)

    return df_cleaned

# Load the model and tokenizer
model_path = Path("MiniLM-L12-v2/")
tokenizer, model = AutoTokenizer.from_pretrained(model_path), AutoModel.
 ↪from_pretrained(model_path)

# Mean Pooling - Take attention mask into account for correct averaging
def mean_pooling(model_output, attention_mask):
    token_embeddings = model_output[0]  # First element of model_output␣
 ↪contains all token embeddings
    input_mask_expanded = attention_mask.unsqueeze(-1).expand(token_embeddings.
 ↪size()).float()
    return torch.sum(token_embeddings * input_mask_expanded, 1) / torch.
 ↪clamp(input_mask_expanded.sum(1), min=1e-9)

def generate_embedding(text):
    # Tokenize input text
    encoded_input = tokenizer(text, padding=True, truncation=True,␣
 ↪return_tensors='pt')
    # Compute token embeddings with model
    with torch.no_grad():
        model_output = model(**encoded_input)
    # Perform mean pooling
    sentence_embedding = mean_pooling(model_output,␣
 ↪encoded_input['attention_mask'])
    # Convert to numpy for FAISS compatibility and ensure it's 2D
```

```
    return sentence_embedding.cpu().numpy().reshape(1, -1)

# Initialize Qdrant client
qdrant_client = QdrantClient(host='localhost', port=6333)
collection_name = "News"

# Specify the vectors' configuration
vectors_config = VectorParams(
    size=model.config.hidden_size,  # The size of your embeddings
    distance=Distance.COSINE  # The distance metric for the vector space
)

# Create or recreate the collection with the specified configuration
qdrant_client.recreate_collection(
    collection_name=collection_name,
    vectors_config=vectors_config,
)

# Function to insert data into Qdrant
def insert_data_into_qdrant(df):
    for index, row in df.iterrows():
        qdrant_client.upsert(
            collection_name=collection_name,
            points=[PointStruct(
                id=index,  # Using the dataframe index as the ID
                vector=row['encoded_title'],  # Assuming row['encoded_title']␣
 ↪is a list of floats
                payload={
                    "title": row['title'],
                    "text": row['text'],
                    "authors": row['authors'],
                    "link": row['link']
                }
            )]
        )

# Function to search for similar content using Qdrant
def similarity_search_with_score(query, k=5):
    query_embedding = generate_embedding(query)[0].tolist()
    search_results = qdrant_client.search(
        collection_name=collection_name,
        query_vector=query_embedding,
        limit=k,
        with_payload=True,
        with_vectors=False
    )
    return search_results
```

```python
# Function to plan content
def plan_content(topic):
    query = topic
    filename = "Dataset/search_results.csv"
    df = search_and_store_to_dataframe(query, filename)
    # Drop rows with any NaN values
    df_cleaned = df.dropna()

    # Encode the titles for similarity search
    df_cleaned['encoded_title'] = df_cleaned['title'].apply(lambda x:␣
 ↪generate_embedding(x)[0].tolist())

    # Insert data into Qdrant
    insert_data_into_qdrant(df_cleaned)

    # Perform similarity search
    search_results = similarity_search_with_score(query=query, k=2)

    # Extract necessary details for the content plan
    latest_trends = []
    for result in search_results:
        payload = result.payload
        latest_trends.append({
            "title": payload.get('title', 'No content available'),
            "text": payload.get('text', 'No content available'),
            "authors": payload.get('authors', 'No content available'),
            "link": payload.get('link', 'No content available')
        })

    target_audience = "General readers interested in the topic"
    seo_keywords = ["example keyword1", "example keyword2"]  # These would be␣
 ↪derived from the analysis
    content_outline = {
        "Introduction": "Brief introduction to the topic.",
        "Key Points": latest_trends,
        "Conclusion": "Summary and call to action."
    }

    content_plan = {
        "Topic": topic,
        "Target Audience": target_audience,
        "SEO Keywords": seo_keywords,
        "Content Outline": content_outline,
        "Resources": latest_trends
    }
```

```
    # Convert content_plan to a string for return
    content_plan_str = (
        f"Topic: {content_plan['Topic']}\n"
        f"Target Audience: {content_plan['Target Audience']}\n"
        f"SEO Keywords: {', '.join(content_plan['SEO Keywords'])}\n"
        f"Content Outline: \n"
        f"  Introduction: {content_outline['Introduction']}\n"
        f"  Key Points: {content_outline['Key Points']}\n"
        f"  Conclusion: {content_outline['Conclusion']}\n"
        f"Resources: {content_plan['Resources']}"
    )

    return content_plan_str

# Mock LLM class to simulate the planner agent's behavior
class MockLLM:
    def bind(self, *args, **kwargs):
        def call(inputs):
            # Directly use inputs as it should be a string
            topic = str(inputs)  # Ensure the topic is treated as a string
            return plan_content(topic)
        return call

# Define the agents
Content_planner = Agent(
    role="Content Planner",
    goal="Plan engaging and factually accurate content on {topic}",
    backstory="You're working on planning a blog article "
              "about the topic: {topic} in 'https://medium.com/'."
              "You collect information that helps the "
              "audience learn something "
              "and make informed decisions. "
              "You have to prepare a detailed "
              "outline and the relevant topics and sub-topics that has to be a␣
 ↪part of the"
              "blogpost."
              "Your work is the basis for "
              "the Content Writer to write an article on this topic.",
    llm=MockLLM(),
    allow_delegation=False,
    verbose=True
)

# Define the tasks for each agent
Content_planner_task = Task(
    description=(
        "1. Prioritize the latest trends, key players, "
```

```
            "and noteworthy news on {topic}.\n"
        "2. Identify the target audience, considering "
            "their interests and pain points.\n"
        "3. Develop a detailed content outline including "
            "an introduction, key points, and a call to action.\n"
        "4. Include SEO keywords and relevant data or sources."
    ),
    expected_output="A comprehensive content plan document "
        "with an outline, audience analysis, "
        "SEO keywords, and resources.",
    agent=Content_planner,
    action=lambda inputs: Content.llm.bind()(inputs)
)


# Crew setup with agents and tasks
# crew = Crew(
#     agents=[Content_planner],
#     tasks=[Content_planner_task],
#     verbose=2
# )

# # Example function call for the content planner agent
# topic = "IN 2024 Indian prime minister Narendra Modi takes office again"
# inputs = {"topic": topic}
# result = crew.kickoff(inputs=inputs)

# print(result)
```

```
/tmp/ipykernel_141168/359440696.py:134: DeprecationWarning:
`recreate_collection` method is deprecated and will be removed in the future.
Use `collection_exists` to check collection existence and `create_collection`
instead.
  qdrant_client.recreate_collection(
2024-06-17 15:07:04 - INFO - HTTP Request: DELETE
http://localhost:6333/collections/News "HTTP/1.1 200 OK"
2024-06-17 15:07:05 - INFO - HTTP Request: PUT
http://localhost:6333/collections/News "HTTP/1.1 200 OK"
```

[13]:
```
# from IPython.display import Markdown,display
# display(Markdown(result))
```

3 All agents

[14]:
```
writer = Agent(
    role="Content Writer",
    goal="Write insightful and factually accurate "
         "opinion piece about the topic: {topic}",
    backstory="You're working on a writing "
              "a new opinion piece about the topic: {topic} in 'https://medium.
 ↪com/'. "
              "You base your writing on the work of "
              "the Content Planner, who provides an outline "
              "and relevant context about the topic. "
              "You follow the main objectives and "
              "direction of the outline, "
              "as provide by the Content Planner. "
              "You also provide objective and impartial insights "
              "and back them up with information "
              "provide by the Content Planner. "
              "You acknowledge in your opinion piece "
              "when your statements are opinions "
              "as opposed to objective statements.",
    allow_delegation=False,
    llm=llm,
    verbose=True
)
write = Task(
    description=(
        "1. Use the content plan to craft a compelling "
            "blog post on {topic}.\n"
            "with atleast 3 topics and 2 subtopics each.\n"
        "2. Incorporate SEO keywords naturally.\n"
  "3. Sections/Subtitles are properly named "
            "in an engaging manner.\n"
        "4. Ensure the post is structured with an "
            "engaging introduction, insightful body, "
            " a summarizing conclusion.\n"
            " and resources with citation and links as the last section.\n"
        "5. Proofread for grammatical errors and "
            "alignment with the brand's voice.\n"
    ),
    expected_output="A well-written blog post "
        "in markdown format, ready for publication, "
        "each section should have 2 or 3 paragraphs.",
    agent=writer,
    action=lambda inputs: writer.llm.bind()(inputs)
)
```

```
editor = Agent(
    role="Editor",
    goal="Edit a given blog post to align with "
         "the writing style of the organization 'https://medium.com/'. ",
    backstory="You are an editor who receives a blog post "
              "from the Content Writer. "
              "Your goal is to review the blog post "
              "to ensure that it follows journalistic best practices,"
              "provides balanced viewpoints "
              "when providing opinions or assertions, "
              "and also avoids major controversial topics "
              "or opinions when possible.",
    llm=llm,
    allow_delegation=False,
    verbose=True
)
edit = Task(
    description=("Proofread the given blog post for "
                 "grammatical errors and "
                 "alignment with the brand's voice."),
    expected_output="A well-written blog post in markdown format, "
                    "ready for publication, "
                    "each section should have 2 or 3 paragraphs.",
    agent=editor,
    action=lambda inputs: editor.llm.bind()(inputs)
)
```

[ ]:
```
crew = Crew(
    agents=[Content_planner, writer, editor],
    tasks=[Content_planner_task, write, edit],
    verbose=2
)

# Example function call for the content planner agent
topic = "IN 2024 Indian prime minister Narendra Modi takes office again"
inputs = {"topic": topic}
result = crew.kickoff(inputs=inputs)

print(result)
```

```
2024-06-17 15:14:33 - WARNING - Overriding of current TracerProvider is not
allowed
```

```
[DEBUG]: == Working Agent: Content Planner
```

[INFO]: == Starting Task: 1. Prioritize the latest trends, key players, and noteworthy news on IN 2024 Indian prime minister Narendra Modi takes office again.
2. Identify the target audience, considering their interests and pain points.
3. Develop a detailed content outline including an introduction, key points, and a call to action.
4. Include SEO keywords and relevant data or sources.

> Entering new CrewAgentExecutor chain…
Failed to process http://www.pmindia.gov.in/en/: Article `download()` failed with 403 Client Error: Forbidden for url: http://www.pmindia.gov.in/en/ on URL http://www.pmindia.gov.in/en/
Failed to process https://www.chathamhouse.org/2024/06/indias-shock-election-result-loss-modi-win-democracy: Article `download()` failed with 403 Client Error: Forbidden for url: https://www.chathamhouse.org/2024/06/indias-shock-election-result-loss-modi-win-democracy on URL https://www.chathamhouse.org/2024/06/indias-shock-election-result-loss-modi-win-democracy

2024-06-17 15:14:38 - INFO - HTTP Request: PUT http://localhost:6333/collections/News/points?wait=true "HTTP/1.1 200 OK"
2024-06-17 15:14:38 - INFO - HTTP Request: PUT http://localhost:6333/collections/News/points?wait=true "HTTP/1.1 200 OK"
2024-06-17 15:14:38 - INFO - HTTP Request: PUT http://localhost:6333/collections/News/points?wait=true "HTTP/1.1 200 OK"
2024-06-17 15:14:38 - INFO - HTTP Request: PUT http://localhost:6333/collections/News/points?wait=true "HTTP/1.1 200 OK"
2024-06-17 15:14:38 - INFO - HTTP Request: PUT http://localhost:6333/collections/News/points?wait=true "HTTP/1.1 200 OK"

Failed to process https://www.reuters.com/world/india/indias-modi-eyes-biggest-win-yet-when-votes-counted-giant-election-2024-06-03/: Article `download()` failed with 401 Client Error: HTTP Forbidden for url: https://www.reuters.com/world/india/indias-modi-eyes-biggest-win-yet-when-votes-counted-giant-election-2024-06-03/ on URL https://www.reuters.com/world/india/indias-modi-eyes-biggest-win-yet-when-votes-counted-giant-election-2024-06-03/

	authors
0	[Ravi Agrawal, Devesh Kapur, Sushant Singh, An…
1	[Pathi Covers India, The Wider South Asia Regi…
2	[]
3	[Authors]
4	[Pathi Covers India, The Wider South Asia Regi…

```
                                                text  \
0  From pundits to polls, there was a wide expect…
1  NEW DELHI (AP) - Since coming to power a decad…
2  Why India's Modi failed to win outright majori…
3  Prime Minister Narendra Modi won a third term …
4  Highlights from India’s election results: Catc…

                                               title  \
0                           Why Modi Underperformed
1  India’s Modi is known for charging hard. After…
2  India election 2024: Why Modi failed to win ou…
3  Indian General Election Results 2024 Highlight…
4  Modi claims victory in India’s election but dr…

                                                link
0  https://foreignpolicy.com/2024/06/06/india-ele…
1  https://apnews.com/article/india-election-resu…
2     https://www.bbc.com/news/articles/c977g8gl5q2o
3  https://www.thehindu.com/elections/lok-sabha/e…
4  https://apnews.com/article/india-election-resu…

2024-06-17 15:14:38 - INFO - HTTP Request: POST
http://localhost:6333/collections/News/points/search "HTTP/1.1 200 OK"
```

Topic: text="You are Content Planner. You're working on planning a blog article about the topic: IN 2024 Indian prime minister Narendra Modi takes office again in 'https://medium.com/'.You collect information that helps the audience learn something and make informed decisions. You have to prepare a detailed outline and the relevant topics and sub-topics that has to be a part of theblogpost.Your work is the basis for the Content Writer to write an article on this topic.\nYour personal goal is: Plan engaging and factually accurate content on IN 2024 Indian prime minister Narendra Modi takes office againTo give my best complete final answer to the task use the exact following format:\n\nThought: I now can give a great answer\nFinal Answer: my best complete final answer to the task.\nYour final answer must be the great and the most complete as possible, it must be outcome described.\n\nI MUST use these formats, my job depends on it!\nCurrent Task: 1. Prioritize the latest trends, key players, and noteworthy news on IN 2024 Indian prime minister Narendra Modi takes office again.\n2. Identify the target audience, considering their interests and pain points.\n3. Develop a detailed content outline including an introduction, key points, and a call to action.\n4. Include SEO keywords and relevant data or sources.\n\nThis is the expect criteria for your final answer: A comprehensive content plan document with an outline, audience analysis, SEO keywords, and resources. \n you MUST return the actual complete content as the final answer, not a summary.\n\nBegin! This is VERY important to you, use the tools available and give your best Final Answer, your job depends on it!\n\nThought: \n"
Target Audience: General readers interested in the topic
SEO Keywords: example keyword1, example keyword2
Content Outline:

Introduction: Brief introduction to the topic.

Key Points: [{'title': 'Indian General Election Results 2024 Highlights: Prime Minister Narendra Modi wins third term in power', 'text': 'Prime Minister Narendra Modi won a third term in power on June 4, but he will be dependent on allies as the BJP fell 32 short of the halfway mark of 272 seats in the Lok Sabha. The BJP-led National Democratic Alliance (NDA) won 286 seats against the Congress-led INDIA bloc’s 231 seats. The Congress won 99 seats compared to 52 in 2019. The BJP won 240 seats.\n\nAndhra Pradesh Assembly election results highlights | Odisha Assembly election result highlights\n\nThe BJP faced setbacks in strongholds and ended up with lower tallies in Uttar Pradesh and Rajasthan, while its southern foray came a cropper and its hopes of expanding in

```
> Finished chain.
```

[DEBUG]: == [Content Planner] Task output: my best complete final answer to the task.\nYour final answer must be the great and the most complete as possible, it must be outcome described.\n\nI MUST use these formats, my job depends on it!\nCurrent Task: 1. Prioritize the latest trends, key players, and noteworthy news on IN 2024 Indian prime minister Narendra Modi takes office again.\n2. Identify the target audience, considering their interests and pain points.\n3. Develop a detailed content outline including an introduction, key points, and a call to action.\n4. Include SEO keywords and relevant data or sources.\n\nThis is the expect criteria for your final answer: A comprehensive content plan document with an outline, audience analysis, SEO keywords, and resources. \n you MUST return the actual complete content as the final answer, not a summary.\n\nBegin! This is VERY important to you, use the tools available and give your best Final Answer, your job depends on it!\n\nThought: \n"
Target Audience: General readers interested in the topic
SEO Keywords: example keyword1, example keyword2
Content Outline:
  Introduction: Brief introduction to the topic.
  Key Points: [{'title': 'Indian General Election Results 2024 Highlights: Prime Minister Narendra Modi wins third term in power', 'text': 'Prime Minister Narendra Modi won a third term in power on June 4, but he will be dependent on allies as the BJP fell 32 short of the halfway mark of 272 seats in the Lok Sabha. The BJP-led National Democratic Alliance (NDA) won 286 seats against the Congress-led INDIA bloc’s 231 seats. The Congress won 99 seats compared to 52 in 2019. The BJP won 240 seats.\n\nAndhra Pradesh Assembly election results highlights | Odisha Assembly election result highlights\n\nThe BJP faced setbacks in strongholds and ended up with lower tallies in Uttar Pradesh and Rajasthan, while its southern foray came a cropper and its hopes of expanding in West Bengal were dashed. Odisha went according to the BJP’s script where it won power for the first time and posted 19 out of 21 Lok Sabha seats. The Congress’s unprecedented outreach to OBC voters in alliance with the Samajwadi Party paid rich dividends in Uttar Pradesh, where the alliance won 44 seats, bringing the BJP tally down to 35 from its earlier tally of 62 seats in 2019, in the battleground State. In Bihar, the Congress-RJD alliance on the same plank failed to enthuse the voters and the NDA alliance held its own.\n\nThe nature of the verdict points to a situation where the BJP, while dominating in numbers over the NDA, will have to pursue a more consensual approach as the numbers secured

```
 [DEBUG]: == Working Agent: Content Writer
 [INFO]: == Starting Task: 1. Use the content plan to craft a
compelling blog post on IN 2024 Indian prime minister Narendra Modi takes office
again.
with atleast 3 topics and 2 subtopics each.
2. Incorporate SEO keywords naturally.
3. Sections/Subtitles are properly named in an engaging manner.
4. Ensure the post is structured with an engaging introduction, insightful body,
a summarizing conclusion.
 and resources with citation and links as the last section.
5. Proofread for grammatical errors and alignment with the brand's voice.
```

> Entering new CrewAgentExecutor chain…

[16]:
```
from IPython.display import Markdown,display
display(Markdown(result))
```

**

India's Elections: A Turning Point for Mr Modi

The recent Indian general elections have been a seismic shift in the country's political landscape. For Prime Minister Narendra Modi, the outcome is a significant turning point, as his BJP party has lost more than 50 seats in parliament. This marks a crucial juncture for Mr. Modi, who has been the dominant force in Indian politics for over a decade.

Mr. Modi's campaign promises revolved around delivering jobs, stability, and economic growth. However, voters have delivered a surprise verdict, with the opposition Congress party-led alliance making significant gains. This outcome is a blow to Mr. Modi's reputation as a leader who has always secured majorities in elections, but it also presents an opportunity for him to refocus his strategy.

The BJP's loss can be attributed to several factors, including joblessness, rising prices, and growing inequality. Controversies surrounding the Citizenship Amendment Act and the revocation of Kashmir's autonomy may have also contributed to the party's decline. Furthermore, voters' trust in the government has been eroded by concerns around national security, corruption, and governance.

However, this setback also presents an opportunity for the opposition to regroup and refocus. With a renewed sense of purpose and momentum, the Congress party can now build on its gains and challenge Mr. Modi's dominance in Indian politics. For India, the election results signal that the country is returning to normal politics, with multiple parties competing for power.

This diversity of political perspectives is essential for a healthy democracy. The elections have shown that even the mighty BJP can be challenged, which presents an opportunity for renewed

political competition and a healthier democracy. In conclusion, the recent Indian general elections have been a significant shock to Prime Minister Narendra Modi's reputation and the ruling BJP party. As Mr. Modi prepares for his third term in office, he will need to confront the challenges of coalition politics and find ways to address the concerns of voters who have abandoned him.

India's Elections: A Turning Point for Mr Modi

As the dust settles on the election results, Prime Minister Narendra Modi faces an unprecedented challenge. His BJP party has suffered a significant loss, marking a turning point in his leadership. Mr. Modi's campaign promises were centred around job creation, economic growth, and national security. However, voters have delivered a verdict that contradicts these promises.

The Indian economy has been struggling, with rising unemployment, stagnant economic growth, and growing inequality eroding trust in the government. The opposition Congress party-led alliance capitalized on these concerns, making significant gains during the elections. Mr. Modi's reputation as an effective leader who has always secured majorities is now being questioned. This setback presents an opportunity for him to re-evaluate his strategy and refocus on delivering meaningful change.

However, this outcome also signals a new era of coalition politics in India. The opposition parties will need to work together to present a united front against the BJP. As Mr. Modi prepares for his third term in office, he must confront the challenges of coalition politics and find ways to address the concerns of voters who have abandoned him. In conclusion, the recent Indian general elections have been a significant shock to Prime Minister Narendra Modi's reputation and the ruling BJP party. This outcome presents an opportunity for renewed political competition, a healthier democracy, and a chance for Mr. Modi to redefine his leadership.

[ ]: