

AD-GPT System Documentation






Table of Contents

1. [Overview](#)
 2. [What the System Does](#)
 3. [Installation Guide](#)
 4. [Getting API Keys](#)
 5. [How to Run the System](#)
 6. [Understanding the Results](#)
 7. [Function Reference](#)
 8. [File Structure](#)
 9. [Troubleshooting](#)
 10. [Advanced Usage](#)
-

Overview

AD-GPT is an automated system that helps researchers, healthcare professionals, and families stay updated on Alzheimer's disease news and research. It searches the internet for recent articles, analyzes them using AI, and creates easy-to-understand summaries and visualizations.

Key Features

-  **Automatic News Search:** Finds the latest Alzheimer's research articles
 -  **AI-Powered Analysis:** Uses OpenAI GPT or Google Gemini to summarize complex articles
 -  **Visual Reports:** Creates charts and graphs showing trends in the research
 -  **Comprehensive Reports:** Generates detailed summaries of key findings
 -  **Model Comparison:** Compare results from different AI models
-

What the System Does

Step-by-Step Process

1. **Search for News:** The system searches DuckDuckGo for recent Alzheimer's disease articles
2. **Collect Articles:** Downloads and processes the full text of each article
3. **AI Analysis:** Uses OpenAI GPT or Google Gemini to create summaries of complex medical content

4. **Sentiment Analysis:** Determines if articles contain positive, negative, or neutral news
5. **Extract Keywords:** Identifies the most important terms and topics
6. **Create Visualizations:** Makes charts showing:
 - How positive/negative the news is overall
 - Most common keywords in a word cloud
 - Which news sources are covering the topic most
7. **Generate Report:** Creates a comprehensive summary highlighting key trends and breakthroughs

What You Get

- **Individual article summaries** in plain language
 - **Visual charts** showing patterns in the research
 - **A comprehensive report** highlighting major trends and breakthroughs
 - **All data saved** to your computer for future reference
-

Installation Guide

Prerequisites

- Python 3.8 or newer installed on your computer
- Internet connection
- At least one AI API key (OpenAI or Google Gemini)

Step 1: Download the Files

Save these two files to a folder on your computer:

- `ad_gpt_system.py` (the main program)
- `requirements.txt` (list of needed software)

Step 2: Install Required Software

Open your command line/terminal and navigate to the folder with the files, then run:

```
bash
```

```
pip install -r requirements.txt
```

This will automatically install all the needed software libraries.

Step 3: Test Installation

Try running this command to make sure everything installed correctly:

```
bash  
  
python ad_gpt_system.py
```

If you see the AD-GPT welcome message, you're ready to go!

Getting API Keys

You need at least one AI service to run the system. You can use either or both:

Option 1: OpenAI (GPT)

1. Go to [OpenAI's website](#)
2. Create an account or sign in
3. Navigate to "API Keys" in your dashboard
4. Click "Create new secret key"
5. Copy the key (it starts with "sk-")
6. **Important:** Keep this key private and secure!

Cost: Pay-per-use, typically \$0.01-0.10 per analysis depending on article count

Option 2: Google Gemini

1. Go to [Google AI Studio](#)
2. Sign in with your Google account
3. Click "Create API Key"
4. Copy the key
5. **Important:** Keep this key private and secure!

Cost: Free tier available, then pay-per-use

Which Should You Choose?

- **OpenAI GPT:** Generally produces more detailed medical analysis
 - **Google Gemini:** Often faster and has a free tier
 - **Both:** Get the most comprehensive analysis by comparing results
-

How to Run the System

Interactive Mode (Recommended for Beginners)

1. Start the Program:

```
bash
```

```
python ad_gpt_system.py
```

2. Enter Your API Key(s):

- When prompted, paste your OpenAI key (or press Enter to skip)
- When prompted, paste your Gemini key (or press Enter to skip)
- You need at least one key to continue

3. Choose Your Default AI Model:

- If you have both keys, choose which AI to use by default
- You can switch later or compare both

4. Use the Menu System: The program will show you a menu with these options:

- **search** - Find Alzheimer's articles (doesn't use AI, free)
- **analyze** - Full analysis with AI summaries (uses API credits)
- **report** - Generate comprehensive report from existing data
- **compare** - Compare OpenAI vs Gemini results (requires both keys)
- **switch** - Change which AI model to use
- **status** - Check system status
- **quit** - Exit the program

Quick Start Example

```
Enter your command: analyze
```

```
Enter analysis query (or press Enter for default): [Press Enter]
```

```
Choose model (openai/gemini) or press Enter for default: [Press Enter]
```

The system will then:

1. Search for Alzheimer's articles (takes 30-60 seconds)
2. Download and analyze each article (2-5 minutes for 20 articles)
3. Create visualizations and reports

4. Show you a summary of results

Understanding the Results

Where Files Are Saved

The system creates a folder called `ad_gpt_workspace` with these subfolders:

```
ad_gpt_workspace/
├── data/                # Raw data files
│   ├── search_results.json    # Original search results
│   └── processed_articles.json # Analyzed articles with summaries
├── summaries/           # AI-generated reports
│   ├── comprehensive_report_openai.txt
│   ├── comprehensive_report_gemini.txt
│   └── model_comparison.json
└── visualizations/      # Charts and graphs
    ├── sentiment_distribution.png
    ├── keywords_wordcloud.png
    ├── sources_distribution.png
    └── model_usage_distribution.png
```

Understanding the Charts

1. Sentiment Distribution (`sentiment_distribution.png`)

- **What it shows:** How positive or negative the Alzheimer's news is overall
- **How to read:**
 - Left side (negative numbers): More concerning/negative news
 - Right side (positive numbers): More hopeful/positive news
 - Center (around 0): Neutral news
- **Example:** If most articles cluster around +0.2, the news is generally slightly positive

2. Keywords Word Cloud (`keywords_wordcloud.png`)

- **What it shows:** Most frequently mentioned terms across all articles
- **How to read:** Bigger words = mentioned more often
- **Look for:** Treatment names, researcher names, breakthrough terms

3. News Sources (`sources_distribution.png`)

- **What it shows:** Which news outlets are covering Alzheimer's research most
- **How to read:** Taller bars = more articles from that source
- **Useful for:** Understanding which sources to follow for regular updates

4. Model Usage (`model_usage_distribution.png`)

- **What it shows:** Which AI model analyzed which articles
- **Only appears:** When you use both OpenAI and Gemini

Reading the Comprehensive Report

The AI-generated report includes:

1. **Key Trends:** What directions research is moving in
 2. **Major Breakthroughs:** Significant new discoveries
 3. **Patient Implications:** What this means for people with Alzheimer's
 4. **Future Research:** What scientists plan to study next
 5. **Overall Outlook:** Whether the news is generally encouraging or concerning
-

Function Reference

Core Functions (What Each Part Does)

`search_news(query, max_results)`

What it does: Searches the internet for Alzheimer's news articles **Input:** Search terms (like "Alzheimer's treatment breakthrough") **Output:** List of article titles, URLs, and basic info **Uses AI:** No - this is free **Time:** 30-60 seconds

`scrape_article_content(url)`

What it does: Downloads the full text of a news article **Input:** Article web address **Output:** Article title and content text **Uses AI:** No - this is free **Time:** 2-5 seconds per article

`summarize_with_llm(text, model)`

What it does: Creates a short, easy-to-understand summary of complex articles **Input:** Full article text **Output:** 2-3 sentence summary in plain language **Uses AI:** Yes - costs API credits **Time:** 5-15 seconds per article

`analyze_sentiment(text)`

What it does: Determines if an article contains positive, negative, or neutral news **Input:** Article text

Output: Numbers showing positivity (-1 to +1) and objectivity (0 to 1) **Uses AI:** No - uses Python

TextBlob library **Time:** Under 1 second

`extract_keywords(text)`

What it does: Finds the most important words and terms in articles **Input:** Article text **Output:** List of key

terms (like "clinical trial", "memory loss", "treatment") **Uses AI:** No - uses word frequency analysis **Time:**

Under 1 second

`generate_comprehensive_report(articles, model)`

What it does: Creates a detailed analysis combining all articles **Input:** All processed articles **Output:**

Multi-paragraph report highlighting trends and breakthroughs **Uses AI:** Yes - costs API credits **Time:** 30-

60 seconds

`compare_models(articles)`

What it does: Shows how OpenAI and Gemini analyze the same content differently **Input:** Processed

articles **Output:** Side-by-side comparison of summaries and reports **Uses AI:** Yes - uses both APIs **Time:**

1-2 minutes

Workflow Functions

`process_news_articles(articles, model)`

What it does: Complete analysis pipeline for all articles **Steps:**

1. Downloads full article text
2. Creates AI summary
3. Analyzes sentiment
4. Extracts keywords
5. Saves everything

`create_visualizations(articles)`

What it does: Creates all the charts and graphs **Creates:**

- Sentiment histogram
- Keywords word cloud
- News sources bar chart

- Model usage pie chart

`run_full_analysis(query, model)`

What it does: Complete end-to-end analysis **Steps:**

1. Search for articles
 2. Process each article
 3. Create visualizations
 4. Generate comprehensive report
 5. Return summary statistics
-

File Structure

Input Files

- `ad_gpt_system.py` - Main program code
- `requirements.txt` - List of required software

Generated Files

- `search_results.json` - Raw search results from DuckDuckGo
- `processed_articles.json` - Articles with AI summaries and analysis
- `comprehensive_report_[model].txt` - Final analysis report
- `model_comparison.json` - Comparison between AI models
- Various `.png` files - Charts and visualizations

File Formats

`processed_articles.json` **structure:**

Each article contains:

json

```
{
  "title": "Article headline",
  "url": "Web address",
  "published": "Publication date",
  "source": "News outlet name",
  "summary": "AI-generated summary",
  "sentiment": {
    "polarity": 0.1,      // -1 (negative) to +1 (positive)
    "subjectivity": 0.5  // 0 (objective) to 1 (subjective)
  },
  "keywords": ["alzheimer", "treatment", "clinical", "trial"],
  "full_content": "First 1000 characters of article...",
  "processed_by": "openai" // Which AI model created the summary
}
```

Troubleshooting

Common Issues

"No articles found"

Problem: Search returned no results **Solutions:**

- Check your internet connection
- Try a broader search term
- Try searching manually on DuckDuckGo to verify the topic exists

"API key invalid" or "Authentication failed"

Problem: AI service can't verify your key **Solutions:**

- Double-check you copied the entire key correctly
- Make sure there are no extra spaces
- Verify your API account has credit/isn't expired
- Try creating a new API key

"Rate limit exceeded"

Problem: Making too many API calls too quickly **Solutions:**

- Wait 5-10 minutes before trying again

- Reduce the number of articles (use `max_results=10` instead of 20)
- Check your API usage limits

"Failed to scrape [URL]"

Problem: Can't download article content **Solutions:**

- This is normal - some websites block automated access
- The system will continue with other articles
- Check that specific URLs work in your web browser

Missing visualizations

Problem: Chart files not created **Solutions:**

- Make sure matplotlib is installed: `pip install matplotlib`
- Check that the `visualizations` folder exists
- Verify you have processed articles data

Performance Tips

Speed Up Analysis

- Use fewer articles: search with `max_results=10`
- Use Gemini (often faster than OpenAI)
- Skip model comparison unless needed

Reduce API Costs

- Use the free search function first to see if articles are relevant
- Use Gemini's free tier when available
- Process articles in batches rather than all at once

Improve Results Quality

- Use specific search terms: "Alzheimer's clinical trial 2024" vs "Alzheimer's"
- Run analysis on recent timeframes for more relevant results
- Compare both AI models for important research

Advanced Usage

Running Programmatically

Instead of using interactive mode, you can use the system in your own Python scripts:

```
python

from ad_gpt_system import ADGPTSystem

# Initialize with your API keys
ad_gpt = ADGPTSystem(
    openai_api_key="your-key-here",
    gemini_api_key="your-key-here",
    default_model="gemini"
)

# Run analysis
results = ad_gpt.run_full_analysis(
    query="Alzheimer's breakthrough 2024",
    model="openai"
)

# Access results
print(f"Analyzed {results['total_articles']} articles")
print(f"Average sentiment: {results['average_sentiment']}")
print(f"Top keywords: {results['top_keywords'][:5]}")
```

Customizing Search Queries

Effective Search Terms:

- **Specific treatments:** "Alzheimer's aducanumab clinical trial"
- **Recent research:** "Alzheimer's breakthrough 2024"
- **Specific aspects:** "Alzheimer's early detection biomarkers"
- **Geographic focus:** "Alzheimer's research Mayo Clinic"

Search Query Tips:

- Use quotes for exact phrases: "early onset Alzheimer's"
- Include timeframes: Alzheimer's treatment 2024
- Combine terms: Alzheimer's AND clinical trial AND FDA approval

Batch Processing Multiple Topics

```
python
```

```
topics = [  
    "Alzheimer's tau protein research",  
    "Alzheimer's prevention lifestyle",  
    "Alzheimer's drug trials 2024"  
]  
  
for topic in topics:  
    print(f"Analyzing: {topic}")  
    results = ad_gpt.run_full_analysis(topic)  
    # Process results...
```

Scheduling Regular Updates

You can set up the system to run automatically using your computer's task scheduler or cron jobs to get regular updates on Alzheimer's research.

Integration with Other Tools

The JSON output files can be easily imported into:

- Excel or Google Sheets for further analysis
- Jupyter notebooks for data science workflows
- Other Python programs for additional processing
- Database systems for long-term storage

Support and Contribution

Getting Help

If you encounter issues not covered in this documentation:

1. Check the console output for specific error messages
2. Verify your API keys are working with simple test calls
3. Make sure all required libraries are installed correctly

Extending the System

The code is designed to be modular and extensible. You can:

- Add new AI models by following the existing pattern

- Create additional visualization types
- Implement new analysis methods
- Add web interfaces or mobile apps

Best Practices

- **API Usage:** Monitor your usage to avoid unexpected charges
- **Data Storage:** Regularly backup your analysis results
- **Updates:** Keep the required libraries updated for security
- **Privacy:** Never share your API keys publicly

This documentation covers AD-GPT System v2.0 with multi-LLM support. For questions or suggestions, refer to the code comments or extend the system for your specific needs.