


Data Ingestion

Document loaders | LangChain

DocumentLoaders load data into the standard LangChain Document format.

 https://python.langchain.com/docs/integrations/document_loaders/



```
%pip install langchain_community
```

Text Loader

`speech.txt`

The world must be made safe for democracy. Its peace must be planted upon the tested foundations of political liberty. We have no selfish ends to serve. We desire no conquest, no dominion. We seek no indemnities for ourselves, no material compensation for the sacrifices we shall freely make. We are but one of the champions of the rights of mankind. We shall be satisfied when those rights have been made as secure as the faith and the freedom of nations can make them.

...

```
## Text Loader
```

```
from langchain_community.document_loaders import TextLoader
```

```
loader=TextLoader('speech.txt')
loader
```

```
<langchain_community.document_loaders.text.TextLoader at 0x1986e2c1510>
```

```
text_documents=loader.load()
text_documents
```

- 📌 `loader.load()` reads the document

Reading a PDf File

```
%pip install pypdf
```

```
## Reading a PDf File
from langchain_community.document_loaders import PyPDFLoader
loader=PyPDFLoader('attention.pdf')
docs=loader.load()
docs
```

[illegible]

Web Base Loader

```
%pip install beautifulsoup4
```

```
from langchain_community.document_loaders import WebBaseLoader
loader=WebBaseLoader(web_path=("https://lilianweng.github.io/posts/2023-06-23-agent/"))
loader.load()
```

Output:

```
[Document(metadata={'source': 'https://lilianweng.github.io/posts/2023-06-23-agent/', 'title': 'LLM Powered Autonomous Agents | Lil'Log', 'description': 'Building agents with LLM (large language mod...
```

- This will display all the content from webpage

Webpage:

The screenshot shows the Lil'Log website with a dark theme. The header includes the site name 'Lil'Log' with a sun icon, and navigation links for 'Posts', 'Archive', 'Search', 'Tags', and 'FAQ'. The main article title is 'LLM Powered Autonomous Agents' in a large, bold font. Below the title, it shows the date 'Date: June 23, 2023', estimated reading time 'Estimated Reading Time: 31 min', and author 'Author: Lilian Weng'. A 'Table of Contents' button is visible. The article text begins with 'Building agents with LLM (large language model) as its core controller is a cool concept. Several proof-of-concepts demos, such as [AutoGPT](#), [GPT-Engineer](#) and [BabyAGI](#), serve as inspiring examples. The potentiality of LLM extends beyond generating well-written copies, stories, essays and programs; it can be framed as a powerful general problem solver.' The section 'Agent System Overview' follows, stating 'In a LLM-powered autonomous agent system, LLM functions as the agent's brain, complemented by several key components:'. A bulleted list under 'Planning' includes 'Subgoal and decomposition: The agent breaks down large tasks into smaller, manageable subgoals, enabling efficient handling of complex tasks.' and 'Reflection and refinement: The agent can do self-criticism and self-reflection over past actions, learn from mistakes and refine them for future steps, thereby improving the quality of final results.'

Lil'Log ☀ | Posts Archive Search Tags FAQ

LLM Powered Autonomous Agents

Date: June 23, 2023 | Estimated Reading Time: 31 min | Author: Lilian Weng

► Table of Contents

Building agents with LLM (large language model) as its core controller is a cool concept. Several proof-of-concepts demos, such as [AutoGPT](#), [GPT-Engineer](#) and [BabyAGI](#), serve as inspiring examples. The potentiality of LLM extends beyond generating well-written copies, stories, essays and programs; it can be framed as a powerful general problem solver.

Agent System Overview

In a LLM-powered autonomous agent system, LLM functions as the agent's brain, complemented by several key components:

- **Planning**
 - Subgoal and decomposition: The agent breaks down large tasks into smaller, manageable subgoals, enabling efficient handling of complex tasks.
 - Reflection and refinement: The agent can do self-criticism and self-reflection over past actions, learn from mistakes and refine them for future steps, thereby improving the quality of final results.

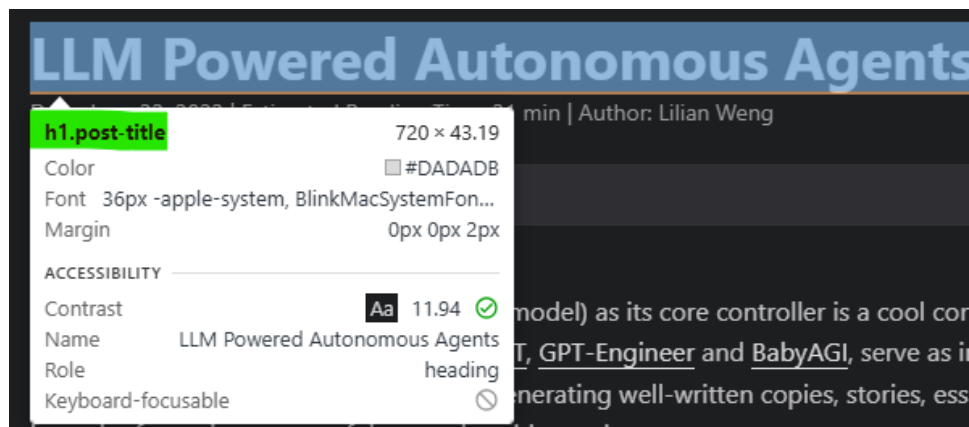
`web_paths` → Multiple URLs (You can give a comma after the URL)

`web_path` → Single URL

If we want specific content from the webpage:

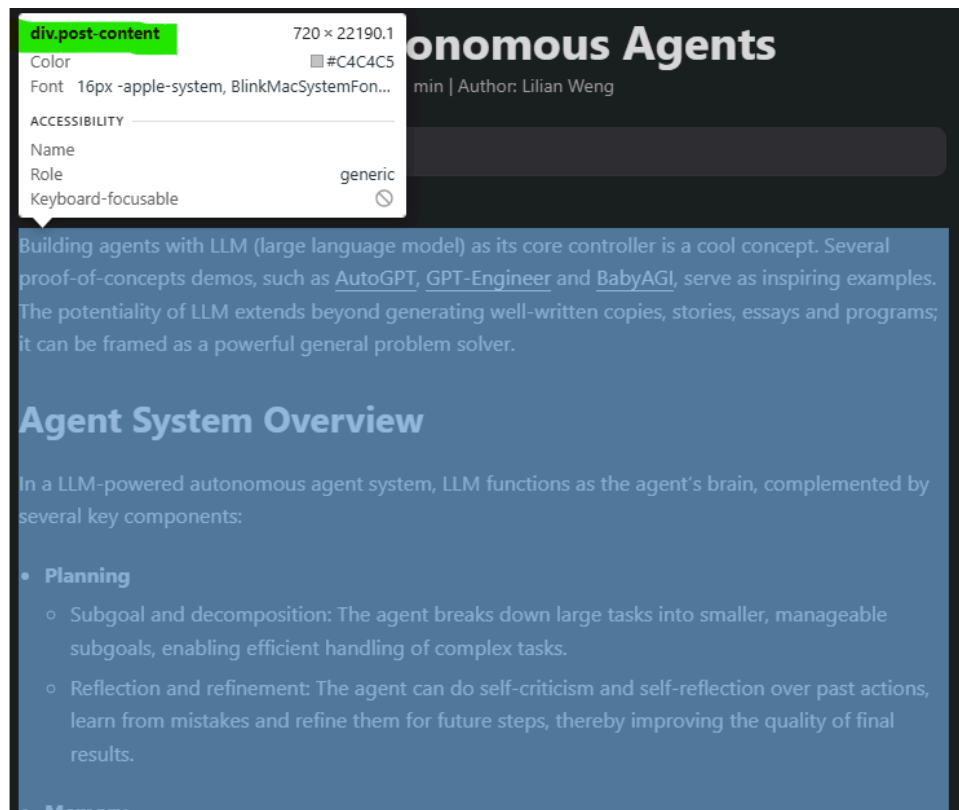
- Use `beautifulsoup4`

Title:



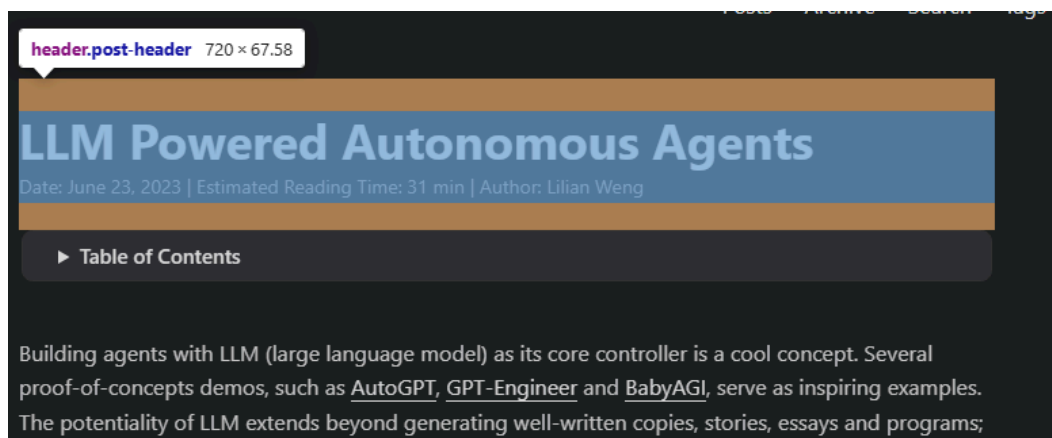
```
<h1 class="post-title"> LLM Powered Autonomous Agents </h1> == $0
```

Content:



```
> <div class="post-content">...</div> == $0
```

Header:



```
<header class="post-header"> == $0
```

```
## Web based loader
```

```
from langchain_community.document_loaders import WebBaseLoader
import bs4
loader=WebBaseLoader(web_paths=("https://lilianweng.github.io/posts/2023-06-23-agent/"),
                      bs_kwargs=dict(parse_only=bs4.SoupStrainer(
                        class_=("post-title","post-content","post-header")
                      ))
)
```

```
loader.load()
```

Output:

```
[Document(metadata={'source': 'https://lilianweng.github.io/posts/2023-06-23-agent/'}, page_content='\n\n    LLM Powered Autonomous Agents\n\n    \nDate: June 23, 2023 | Estimated...']
```

arXiv Loader

```
pip install arxiv
```

```
pip install pymupdf
```

It helps you automatically fetch and load research papers from arXiv.org, directly into your Python code so you can use them in your LangChain pipeline (like load → split → embed → store → ask).

What Is arXiv (pronounced "archive")?

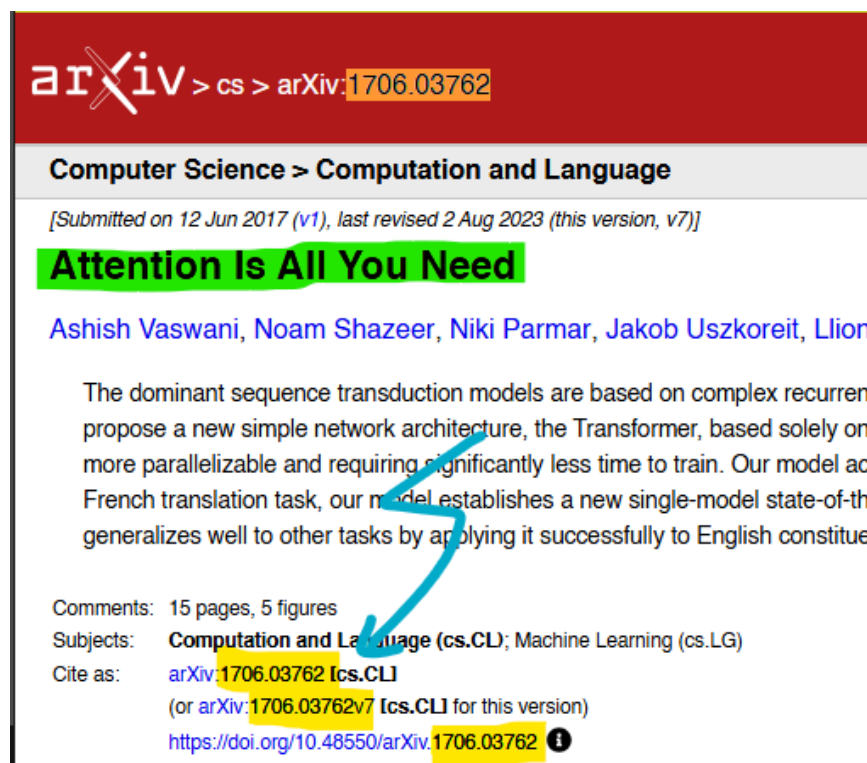
- A free **open-access repository** for scientific papers (physics, CS, AI, etc.).
- Researchers share cutting-edge work *before* formal peer review ("preprints").
- **Website:** arxiv.org

```
from langchain_community.document_loaders import ArxivLoader
docs = ArxivLoader(query="1706.03762", load_max_docs=2).load()
len(docs)
```

Output:

1

- **1706.03762** → arXiv paper ID or search term



- **load_max_docs=2** → Downloads **up to 2** papers
- Returns them as **documents** (you can now split, embed, etc.)

docs

Output:

```
[Document(metadata={'Published': '2023-08-02', 'Title': 'Attention Is All You Need', 'Authors': 'Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N...
```

```
print(f"""
Title: {docs[0].metadata['Title']}
Authors: {docs[0].metadata['Authors']}
Published: {docs[0].metadata['Published']}
""")
```

```
Title: Attention Is All You Need
Authors: Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin
Published: 2023-08-02
```

Wikipedia Loader

```
pip install wikipedia
```

```
from langchain_community.document_loaders import WikipediaLoader
docs = WikipediaLoader(query="Generative AI", load_max_docs=2).load()
len(docs)
print(docs)
```

Output:

```
Document(metadata={'title': 'Generative artificial intelligence', 'summary': 'Generative artificial intelligence (Generative AI, GenAI, or GAI) is a subfield of artificial...
```


Title: Attention Is All You Need

Authors: Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

Published: 2023-08-02