Scientific Papers RAG System - Complete macOS Setup Guide

A comprehensive guide to set up the Scientific Papers Retrieval-Augmented Generation (RAG) system on macOS from scratch, including all dependencies and configurations.

■ Prerequisites

• **macOS**: 10.15 (Catalina) or later
• **Hardware**: Mac with at least 8GB RAM (16GB+ recommended for better performance)
• **Storage**: At least 10GB free space for dependencies and models
• **Internet**: Stable connection for downloading dependencies

■ Complete Installation Guide

Step 1: Install Homebrew (Package Manager)

Open Terminal (Applications → Utilities → Terminal) and run:

/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

Follow the prompts and add Homebrew to your PATH when instructed.

Step 2: Install Python 3.14

Install Python 3.14 via Homebrew

brew install python@3.14

Verify installation

python3 --version

\1: \1

Step 3: Install Ollama (Local LLM Runtime)

Install Ollama

brew install ollama

Start Ollama service

brew services start ollama

Verify Ollama is running

ollama --version

Step 4: Download Required LLM Models

Download Llama2 model (7B parameters, ~3.8GB)

ollama pull llama2

Download LLaVA vision model (7B parameters, ~4.7GB)

ollama pull llava

Verify models are installed

ollama list

\1:

NAME ID SIZE MODIFIED

llama2 latest 3.8 GB X minutes ago

llava latest 4.7 GB X minutes ago

Step 5: Install Python Dependencies

Install core dependencies

pip3 install --upgrade pip

Install RAG system packages

pip3 install faiss-cpu sentence-transformers ollama

Install PDF processing libraries

pip3 install PyPDF2 pymupdf pdfplumber

Install LangChain for text processing

pip3 install langchain langchain-community langchain-text-splitters

Install additional dependencies

pip3 install numpy tqdm requests httpx pillow

Verify key installations

python3 -c "import faiss; print('FAISS: OK')"
python3 -c "import sentence_transformers; print('SentenceTransformers: OK')"
python3 -c "import ollama; print('Ollama client: OK')"

Step 6: Download the RAG System

Choose one of these options:

Clone the repository

git clone
cd docDatabase

Create project directory

mkdir ~/Documents/docDatabase

cd ~/Documents/docDatabase

Create the required directory structure

mkdir -p papers data embeddings logs

Then download/copy these files to the project directory:

• `rag_builder.py`
• `rag_query.py`
• `utils.py`
• `config.py`
• `image_processor.py`

Step 7: Verify System Setup

Test that everything is working:

cd ~/Documents/docDatabase

Test Python imports

python3 -c "

import faiss

import sentence_transformers

import ollama

from utils import PDFProcessor

print('■ All imports successful!')

"

Test Ollama connection

python3 -c "

import ollama

client = ollama.Client()

response = client.list()

print('■ Ollama connected, models:', [m['name'] for m in response['models']])

"

■ Usage Instructions

1. Prepare Your PDF Papers

Create papers directory if it doesn't exist

mkdir -p ~/Documents/docDatabase/papers

Copy your PDF files to the papers directory

cp /path/to/your/papers/*.pdf ~/Documents/docDatabase/papers/

2. Build the RAG Database

cd ~/Documents/docDatabase

Run the RAG builder (this will take several minutes)

python3 rag_builder.py

\1:

• Processes each PDF (text extraction + image analysis)
• Creates text embeddings using sentence-transformers
• Saves FAISS vector database
• Generates ~5-15 minutes for 5 PDFs

3. Query Your Papers

Start the interactive query system

python3 rag_query.py

\1:

• "What machine learning algorithms are discussed?"
• "Describe any graphs or charts in the papers"
• "What are the main findings about deep learning?"
• "Tell me about the experimental methodology"

4. Monitor System Resources

Check memory usage during processing

top -pid $(pgrep python3)

Check available disk space

df -h

Check Ollama status

brew services list | grep ollama

■■ Configuration Options

Memory Optimization

Edit \1 to adjust for your system:

For 8GB RAM systems

CHUNKING = {
"chunk_size": 500, # Smaller chunks
"chunk_overlap": 50, # Less overlap
"min_chunk_size": 100
}

For 16GB+ RAM systems

CHUNKING = {
"chunk_size": 1000, # Larger chunks
"chunk_overlap": 100, # More overlap
"min_chunk_size": 200
}

Disable Image Processing (if needed)

In \1:

IMAGE_PROCESSING = {

"enabled": False, # Set to False to disable

... other settings

}

■ Troubleshooting

Common Issues and Solutions

Add Python to PATH

echo 'export PATH="/usr/local/bin:$PATH"' >> ~/.zshrc

source ~/.zshrc

Reinstall with specific Python version

python3 -m pip install faiss-cpu --force-reinstall

Restart Ollama service

brew services restart ollama

Check if running on correct port

curl http://localhost:11434/api/version

Process fewer PDFs at once

Or increase swap space

sudo sysctl vm.swapusage

Install additional PDF dependencies

brew install poppler

pip3 install pdfminer.six

Performance Optimization

Use more CPU cores for embeddings

export MKL_NUM_THREADS=4

export OMP_NUM_THREADS=4

Run with higher priority

sudo nice -n -10 python3 rag_builder.py

Process PDFs individually

python3 -c "

from rag_builder import RAGBuilderFAISS

builder = RAGBuilderFAISS()

builder.process_single_pdf('papers/your_paper.pdf')

"

■ System Requirements & Performance

Minimum Requirements:

• **RAM**: 8GB (4GB for system + 4GB for models)
• **Storage**: 5GB (models + dependencies)
• **CPU**: Intel/Apple Silicon Mac
• **Time**: ~2-3 minutes per PDF

Recommended Requirements:

• **RAM**: 16GB+ (better performance)
• **Storage**: 10GB+ (room for more models)
• **CPU**: Apple Silicon M1/M2/M3 (faster inference)
• **Time**: ~1-2 minutes per PDF

Expected Performance:

• **5 PDFs**: ~10-15 minutes total processing
• **Memory Usage**: ~3-5GB during processing
• **Database Size**: ~50-100MB for embeddings
• **Query Speed**: 2-5 seconds per question

■ Updating the System

Update Ollama

brew update && brew upgrade ollama

Update Python packages

pip3 install --upgrade faiss-cpu sentence-transformers ollama langchain

Update models (if new versions available)

ollama pull llama2

ollama pull llava

■ Directory Structure

After setup, your directory should look like:

~/Documents/docDatabase/

■■■ papers/ # Your PDF files

■ ■■■ paper1.pdf

■ ■■■ paper2.pdf

■ ■■■ ...

■■■ embeddings/ # Generated vector database

■ ■■■ faiss_index.bin

■ ■■■ document_store.pkl

■■■ data/ # Processed documents info

■ ■■■ processed_documents.json

■■■ logs/ # System logs

■■■ rag_builder.py # Main builder script

■■■ rag_query.py # Query interface

■■■ utils.py # Utility functions

■■■ config.py # Configuration

■■■ image_processor.py # Image analysis

■■■ SETUP_GUIDE_MACOS.md # This guide

■ Quick Start Summary

For experienced users, here's the condensed version:

1. Install dependencies

/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

brew install python@3.14 ollama

brew services start ollama

2. Install Python packages

pip3 install faiss-cpu sentence-transformers ollama PyPDF2 pymupdf pdfplumber langchain langchain-text-splitters

3. Download models

ollama pull llama2 && ollama pull llava

4. Setup project

mkdir -p ~/Documents/docDatabase/{papers,data,embeddings,logs}

cd ~/Documents/docDatabase

Copy RAG system files here

5. Run system

cp /path/to/your/papers/*.pdf papers/

python3 rag_builder.py

python3 rag_query.py

■ Support

If you encounter issues:

1. **\1**: Look in the `\1` directory for detailed error messages
2. **\1**: Ensure all components are compatible
3. **\1**: Run individual tests for Python, Ollama, and dependencies
4. **\1**: Use Activity Monitor to check CPU/memory usage
5. **\1**: Remove and reinstall problematic components

■ **\1** You now have a fully functional scientific papers RAG system with multimodal capabilities running locally on your Mac!