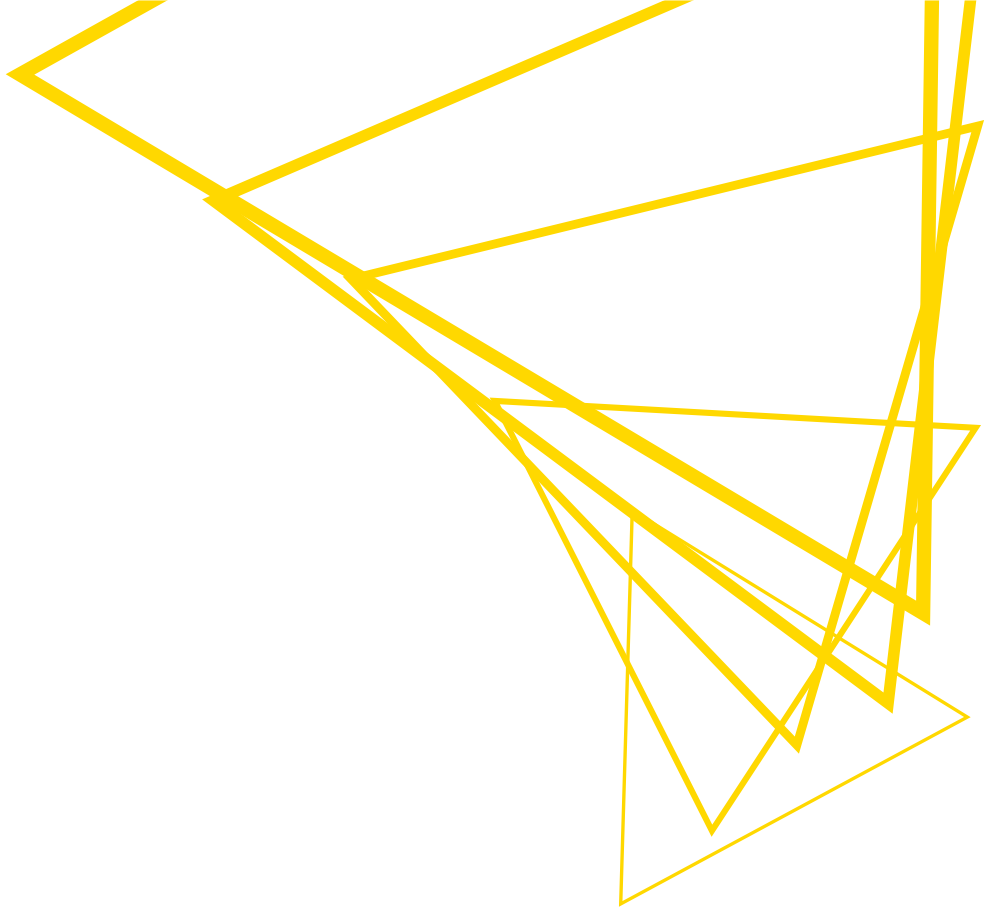


A Friendly Introduction to Codeless AI

Evangelism Team

December 2, 2023



Agenda

1. What's AI?
2. Intro to LLMs
3. The KNIME AI Extension
4. Basic LLM customization
5. Lab Session: *Generative AI and Sentiment Prediction*
6. Advanced LLM customization
7. Use case: *Building a Custom Chatbot*

What's AI?



AI in a Nutshell

- *"Can machines think?"*

–Alan Turing, *Computing Machinery and Intelligence* (1950)

- **“Artificial intelligence (AI)**, the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings. The term is frequently applied to the project of developing systems endowed with the intellectual processes characteristic of humans, such as the ability to reason, discover meaning, generalize, or learn from past experience.”

–Encyclopaedia Britannica

AI in a Nutshell

- AI currently encompasses a huge variety of subfields from the general (e.g., learning and perception), to the specific (e.g., playing chess, proving mathematical theorems, diagnosing diseases, etc.).
- **Weak AI**
 - also called Narrow AI or Artificial Narrow Intelligence is AI trained and focused to perform specific tasks.
 - weak AI drives most of the AI that surrounds us today (e.g., autonomous driving systems, vocal assistants, image generators, etc.).
- **Strong AI**
 - is made up of Artificial General Intelligence and Artificial Super Intelligence.
 - is a theoretical form of AI where a machine would have an intelligence equaled or superior to humans.
 - would have a self-aware consciousness that could solve problems, learn, and plan for the future.

AI vs. Deep Learning

Artificial intelligence

Any technique that enables machines to mimic human intelligence

Machine learning

Ability to learn without being explicitly programmed using past observations

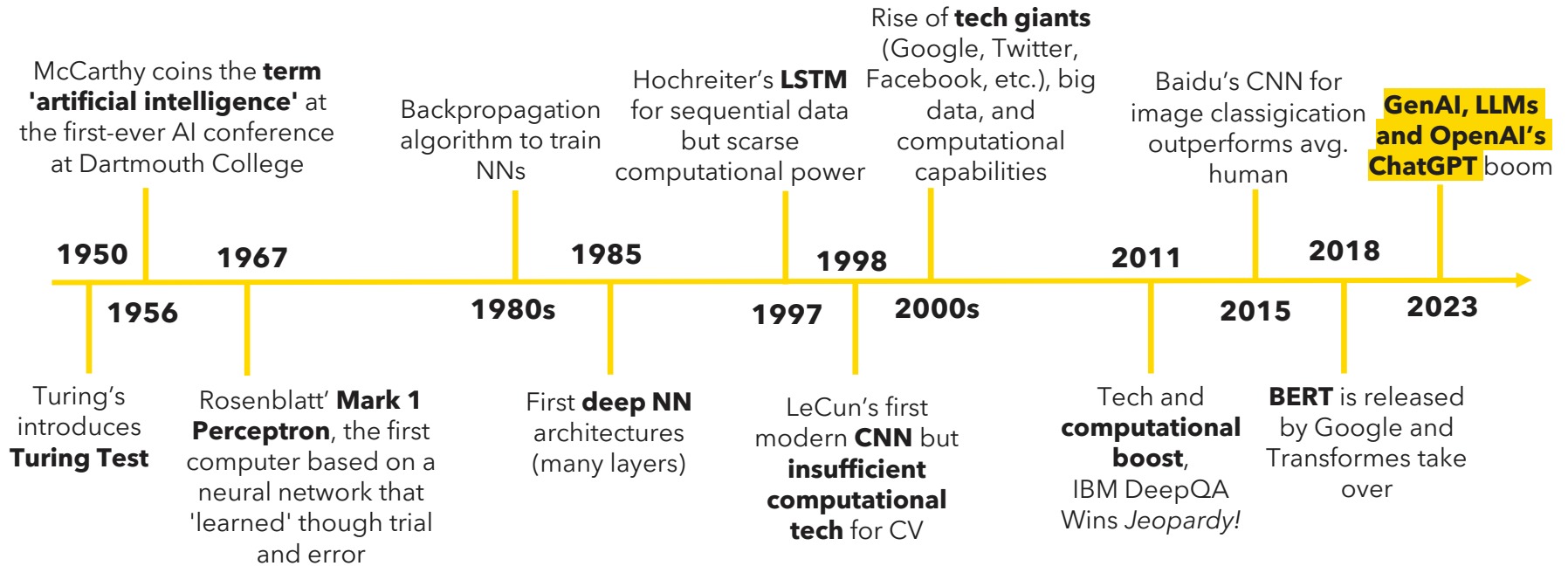
Artificial neural networks

Extract patterns using neural networks

Deep learning

Modern revolution of neural networks

AI in a Timeline



Intro to LLMs



LLMs in a Nutshell

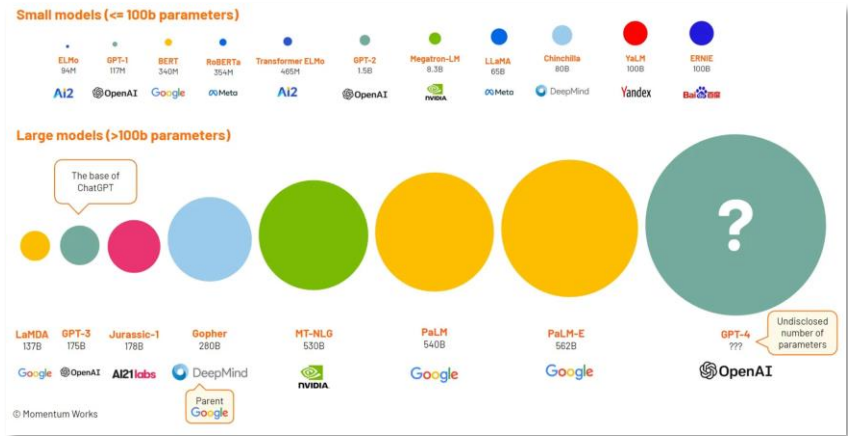
- **Large language models** (LLMs) are a class of deep neural networks that are revolutionizing human interaction with technology.
- Their ease of access and diffuse impact is redefining practices and procedures of various industries, including tech, design, education and professional services.
- You've likely come across LLMs, or you've almost certainly utilized them without realizing it.
 - Search engines leverage LLMs to improve the relevance and accuracy of search results by understanding the context and intent of user queries.
 - LLMs-powered chatbot are used to provide effective and engaging customer support.
 - ...

LLMs in a Nutshell

- LLMs serve as the foundation for **conversational AI systems** due to their unparalleled proficiency in understanding, generating, and simulating human conversational behaviors.
- LLMs are often **multipurpose** and **multimodal**
 - Multipurpose: they excel in a wide array of tasks (e.g., text summarization, code generation, translation, text editing, completion, Q&A, etc.).
 - Multimodal: they go beyond text and natural languages to understand and generate images, videos, and audios.
- OpenAI's ChatGPT and GPT-4, Google's Bard, Databricks' Dolly or Meta AI's LLaMa 2 are popular examples.

Why *large* language models?

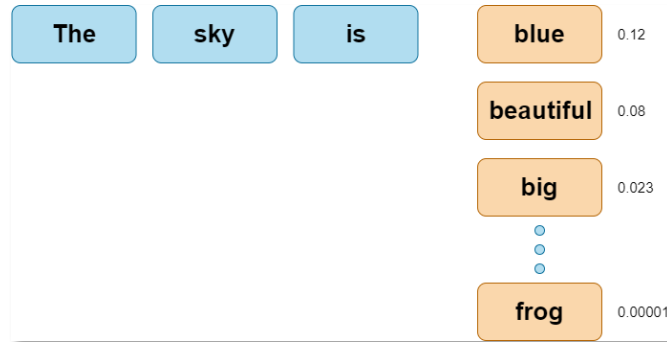
- LLMs emerged in 2018 with the advent of Transformers but have significantly increased
 - Size (no. of trainable parameters).
 - Proficiency.
- The adjective **large** refers to the billions of trainable parameters used to understand and generate responses in a human-like fashion
- A clear trend: the larger, the better
 - OpenAI's GPT-4 (~176T) > GPT-3 (175B)



Source: <https://thelowdown.momentum.asia/the-emergence-of-large-language-models-llms/>

How do LLMs “think”?

- LLMs function like highly sophisticated auto-completion systems (like text suggestions on smartphones).
- LLMs are trained to suggest the most likely next word or phrase based on previous extensive exposure to similar contexts.



This is because the model was trained on millions of texts and learned “sky” to be the best next token.

- This proficiency enables an LLM-based tool to provide the most probable answer to a question, with reliability contingent on the breadth and quality of the training data.

How do LLMs “think”?

- However, notice that LLMs possess only probabilistic knowledge and lack semantic understanding.
- Even though LLMs are called “intelligent”, they are not *really* aware of the meaning behind what they are saying.
- For instance, while they can correctly answer “1+2” as “3,” it doesn't imply a true comprehension of logic operations; it only implies that the most probable answer was “3”.

What makes LLMs so powerful?

Factor 1: Data

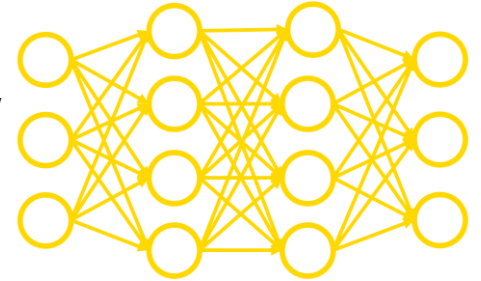
- LLMs were trained on very extensive unlabeled data, encompassing a diverse range of texts, sources and documents.
 - For example, OpenAI's ChatGPT was honed on 570 gigabytes of text sourced from a wide array of credible outlets, including academic papers and books.
- The model then applies self-supervised learning to generate its own labels or targets from the data. This process allows LLMs to learn and refine their linguistic abilities.
- LLMs have the capacity to further enrich their knowledge through interaction with users of the service.



What makes LLMs so powerful?

Factor 2: Architecture

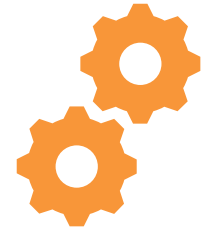
- LLMs build essentially on the foundation laid by Transformers that have a remarkable ability to capture complex relationships between words.
- Transformers achieve this through a **self-attention** mechanisms, that allows the model to simultaneously consider all words in a sentence and weigh their importance (= **attention score**) in relation to every other word.
- This means Transformers can discern nuanced connections and contextual relationships (even if the words are far apart in the sentence) and produce accurate and contextually relevant responses.



What makes LLMs so powerful?

Factor 3: Training

- LLMs' vast neural network architecture comprises billions of parameters, which essentially represent the model's learned knowledge.
 - During training, these parameters continually evolve and adapt to optimize the model's performance.
- One notable technique used during training is **autoregressive modeling**, where the model learns to predict the next word in a sentence based on the previous one.
- Training LLMs from scratch is exorbitantly costly in terms of computational power, time and data sources to the extent that this is primarily accessible only to tech giants.



Risks and limitations of LLMs

■ **Explainability**

- LLMs are referred to as "black box" models, because they are so complex that it is impossible for humans to peer inside and comprehend the decision-making process.
- This opacity can be harmful, leaves room for errors, or prevent technology adoption
- XAI techniques can help up to a certain point

■ **Hallucinations, toxic content, and copyright infringement**

- 'Hallucinating' incorrect facts
- Being biased, gullible or persuaded into creating toxic content
- Incur copyright infringement for generated images/texts

■ **Data privacy**

- Risk of leaking confidential data, as information input into the model operated via a third-party API may become the property of the provider and utilized for future training.

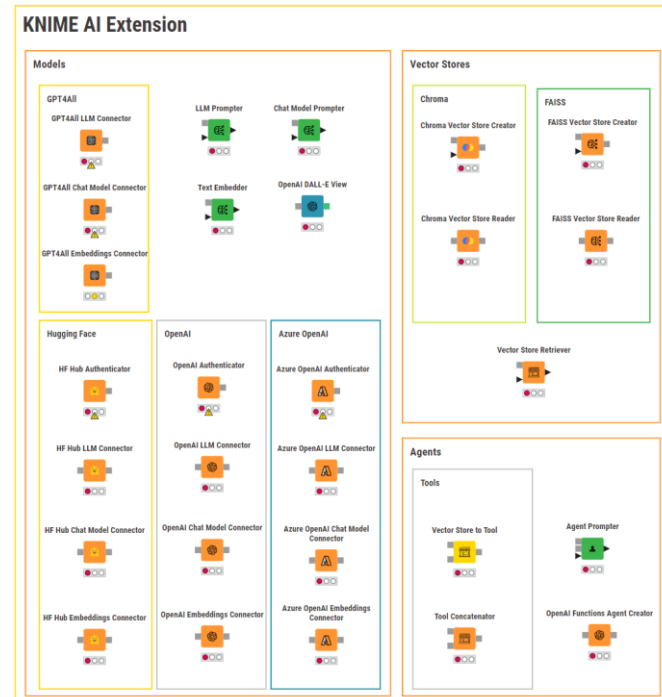
The KNIME AI Extension



KNIME AI Extension

LLMs can be leveraged in KNIME thanks to a dedicate extension that allows connections to open-source and closed-source models via API, or to open-source local models

- Authenticators
- LLM Connectors
- Chat Model Connectors
- Embeddings Connectors
- LLM and Embeddings Prompters
- Local Connectors and Embedders
- Vector Stores
- Agents



Open-Source vs. Closed-Source LLMs via API

HuggingFace

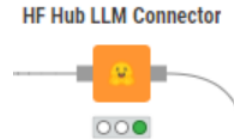
HF Hub Authenticator

- Authenticates to all HF Hub models.
- Requires flow variable of type credentials.



HF Hub LLM Connector

- Establishes a connection to a specific LLM hosted on HF.
- Requires LLM's Repo ID (e.g., *bigscience/mt0-large*).
- Model tasks can be text-generation, text2text-generation or summarization.



OpenAI

OpenAI Authenticator

- Provides authentication for all OpenAI models.
- Requires flow variable of type credentials.



OpenAI LLM Connector

- Establishes a connection to a specific LLM hosted on OpenAI (e.g., text-davinci-003).
- Customizable model settings (e.g., max tokens, temperature, etc.).



Open-Source Local LLMs

GPT4All

- **GPT4All LLM Connector**

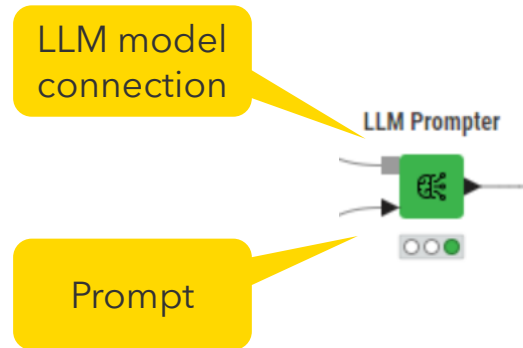
- Models can be downloaded from GPT4All.
- Accesses LLMs stored locally.
- Customizable model settings (e.g., max tokens, temperature, etc.).
- Longer execution times.

GPT4All LLM Connector



LLM Prompter

- Sends one prompt to the LLM and receives a response for each row in the input table.
- Rows are treated independently, i.e. the LLM can not remember the content of previous rows or how it responded to them.



Chat models

HuggingFace

- **HF Hub Chat Model Connector**
 - Requires chat LLM's Repo ID.
 - Model tasks: text-generation, text2text-generation or summarization.
 - Model-specific system prompt and/or prompt template required.
 - Customizable model settings.

HF Hub Chat Model Connector



OpenAI

- **OpenAI Chat Model Connector**
 - Establishes a connection to a specific chat model hosted on OpenAI (e.g., gpt-3.5-turbo, gpt-4).
 - Customizable model settings.

OpenAI Chat Model Connector



GPT4All

- **GPT4All Chat Model Connector**
 - Accesses chat model downloaded from GPT4All and stored locally.
 - Model-specific system prompt and/or prompt template required.
 - Customizable model settings.

GPT4All Chat Model Connector



Chat Model Prompter

- Takes a prompt and the conversation history between human and AI.
- Generates a response for the prompt with the knowledge of the previous conversation.

The (next) message that will be added to the conversation

Specifies the sender role of the messages, e.g., Human or AI

Dialog - 6:85 - Chat Model Prompter (First message with)

System Message

You are a helpful KNIME assistant.

Message

Hey there!

Conversation Settings

Message role

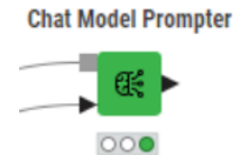
Type

Messages

Message

Cancel Ok

Describes how the model should behave





Basic LLM customization

Prompt Engineering

- A prompt is a concise and clear input text that serves as a query or instruction to a language model to generate desired outputs.
- It represents the most straightforward manner for human users to ask LLMs to solve a task.
- **Best practices for good prompt engineering:**
 - Put instructions at the beginning of the prompt.
 - Be specific, descriptive and detailed about the desired context, outcome, length, format, style, etc.
 - If possible, provide examples of the desired output format.
 - Avoid redundancy and ambiguous phrasing.
 - Instruct the model what to do vs. what not to do.

Prompt Engineering Example

```
joinSep("\n\n",  
"You are an AI expert in the coffee machine 'Barista Express BES875/SES875'.",  
"Use the following pieces of context from the manual book to answer the question at the end.",  
"If you don't know the answer, just say that you don't know, don't try to make up an answer.",  
"Context from the Manual Book:",  
join("(", joinSep("\n\n", $Retrieved documents$), ")"),  
join("Question: ($Questions$),"),  
"Helpful Answer:")
```

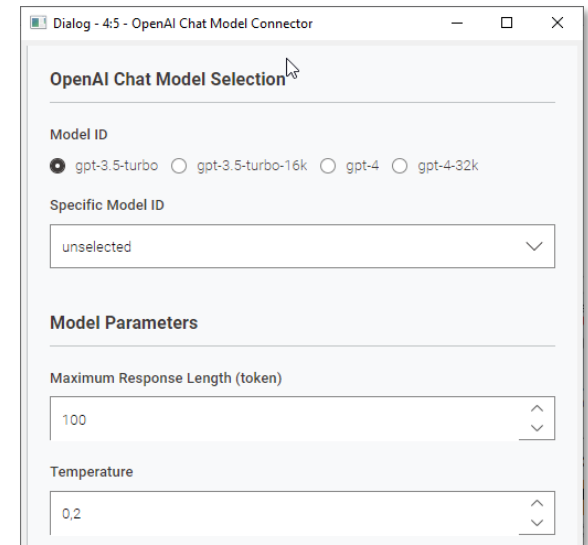
String Manipulation



Hyperparameter Tuning

- Hyperparameters are a set of configurable options determined by the user and can be tuned to guide, optimize or shape model performance with respect to a specific task.
- Key hyperparameters in LLMs:**
 - Temperature (0-2):** serves as a control knob over the level of randomness exhibited in the model's outputs. 0 = fully deterministic, 2 = high propensity to explore less likely options.
 - Context window:** the number of preceding tokens that the model considers when generating text.
 - Max number of tokens:** the maximum number of tokens that model generates in the output. It includes both the tokens generated by the model and the tokens present in the input prompt.

They are *external* factors that influence the model's learning process and outcome.



The screenshot shows a dialog box titled "Dialog - 4:5 - OpenAI Chat Model Connector". It contains the following sections:

- OpenAI Chat Model Selection:** This section includes a "Model ID" field with four radio button options: "gpt-3.5-turbo" (selected), "gpt-3.5-turbo-16k", "gpt-4", and "gpt-4-32k". Below it is a "Specific Model ID" dropdown menu currently set to "unselected".
- Model Parameters:** This section includes a "Maximum Response Length (token)" field with a value of "100" and a "Temperature" field with a value of "0.2". Both fields have up and down arrow buttons for adjustment.

Abstract geometric lines in the top right corner, consisting of several overlapping, thin white lines forming various triangular and polygonal shapes against a yellow background.

Lab Session: **Generative AI** and **Sentiment Prediction**

Demo

Demo: Generative AI

Closed-source AI for Text Generation (via API)

Workflow description: This workflow leverages OpenAI's ChatGPT to generate an answer to the question provided by the user in the prompt. It requires an OpenAI API key, which can be requested at: <https://openai.com/>

Table Creator



Write prompt

Open-source AI for Text Generation (via API)

Workflow description: This workflow leverages models available on Hugging Face Hub* to generate an answer to the question provided by the user in the prompt. Popular models are: bigscience/mtd-large or google/flan-t5-xl. It requires a Hugging Face access token, which can be requested for free at: <https://huggingface.co/>

* Make sure the "Hosted Inference API" by Hugging Face is available for the selected model.

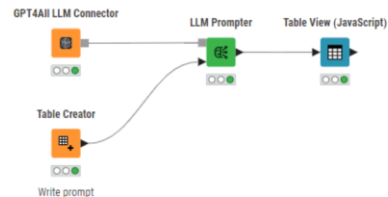
Table Creator



Write prompt

Open-source AI for Text Generation (local models)

Workflow description: This workflow leverages local models available on GPT4All to generate an answer to the question provided by the user in the prompt. A popular model is: gpt4all-falcon. Check model list at: <https://gpt4all.io/index.html>



Exercise 01: Sentiment Prediction with LLMs

- **Sentiment analysis** is trying to answer the question “What do people feel/think about a certain topic?”
- Exercise 01: Predict the sentiment (positive or negative) of movie reviews with open-source and closed-source LLMs via API
 - Authenticate and connect to the chosen LLM model
 - Engineer an effective prompt
 - Tweak hyperparameters (if needed)
 - Assess the performance of the LLM
- Find the exercise workflow *01_Sentiment_Predictor* in *Day 4 > AI*
- Temporary OpenAI API keys are available in *Day 4 > AI* in *ADILT-2023_OpenAI API key.txt*

The top right corner of the slide features several overlapping, thin white lines that form a series of irregular, angular shapes, resembling a stylized geometric pattern or a network of connections.

Advanced LLM customization

LLMs are smart but they do not know a lot of things!

R You
Who won the presidential elections in Argentina in November 2023?

ChatGPT
I'm sorry, but I don't have information on events or elections that occurred after my last update in January 2023. You might want to check the latest news or reliable sources for the most recent information on the Argentine presidential elections in November 2023.

Lack of up-to-date information

Not much we can do with prompt engineering or hyperparameter tuning

R You
What's the KNIME AI Learnathon?

ChatGPT
As of my last update in January 2023, I don't have specific information about a KNIME AI Learnathon. However, it's possible that events and activities like learnathons may have been organized by the KNIME community or the company itself after that date.

LLMs are smart but they do not know a lot of things!



You

Which KNIME node (or nodes) makes the most sense to use in deep learning for transfer learning?



ChatGPT

In KNIME, for deep learning transfer learning tasks, you'll primarily use the "Transfer Learning (Keras)" node along with other nodes for data preparation and model evaluation. Here's an overview:

Not much we can do with prompt engineering or hyperparameter tuning

Lack of domain-specific knowledge, leading to hallucinations or factually wrong information



You

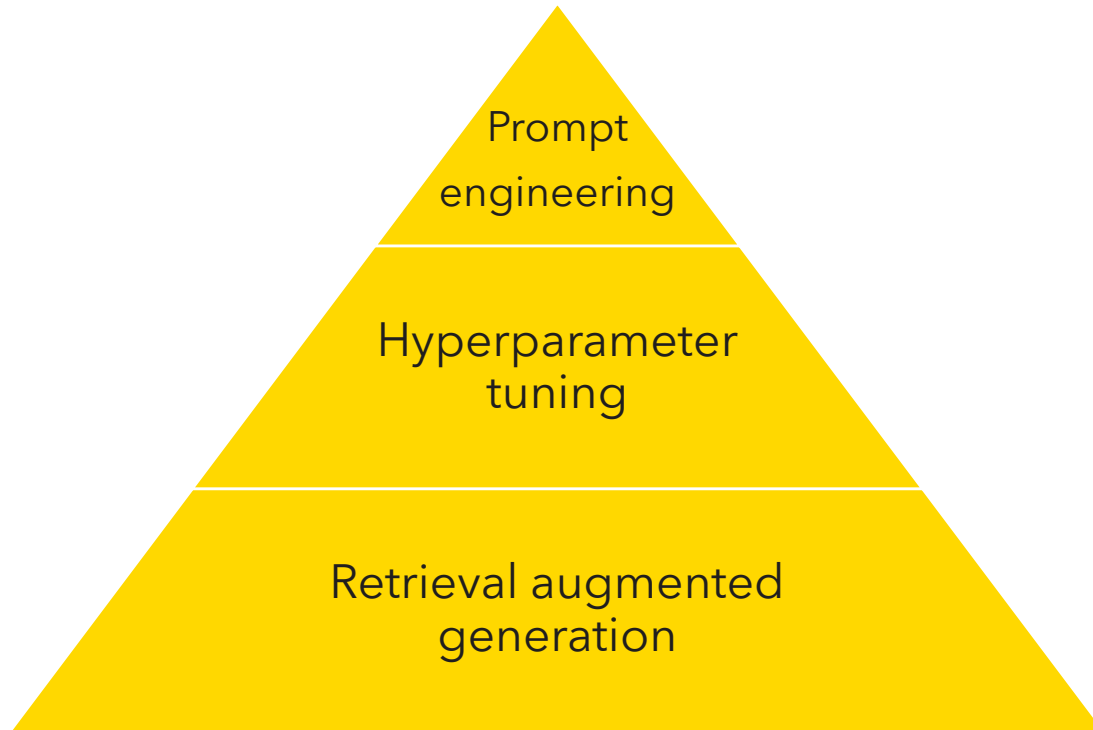
What KNIME node should I use to read PDFs I want to keep the text formatting?



ChatGPT

For reading PDFs while preserving text formatting in KNIME, you can use the "PDF Parser" node. This node is part of the "Text Processing" extension in KNIME and is designed to extract text from PDF documents while attempting to maintain the original formatting.

Add a new layer to customize LLMs' responses



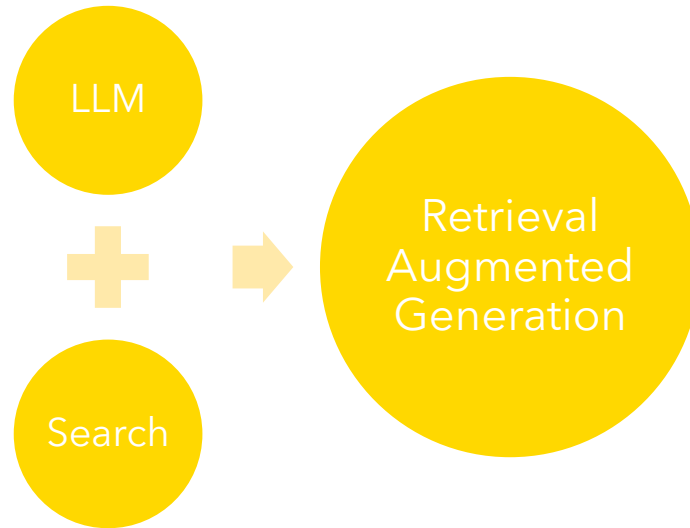
Retrieval Augmented Generation

- Retrieval augmented generation (RAG) is an AI framework that overcomes knowledge limitations of generalist LLMs.
- Its goal is to leverage LLMs' general ability to understand and generate responses in natural languages BUT:
 - Making LLMs more knowledgeable.
 - Their response more relevant to the user/application.
 - Mitigating risks of hallucinations, biases or non-factual information.
- How?
 - Giving generalist LLMs access to **user-curated** and **domain-relevant knowledge bases** (e.g., data sources with specific knowledge, terminology, context or up-to-date information) to customize responses for specific applications.

Put simply: RAG is LLMs on steroids 😊

Retrieval Augmented Generation

- Or in other words: Search for related documents and figure out the answer

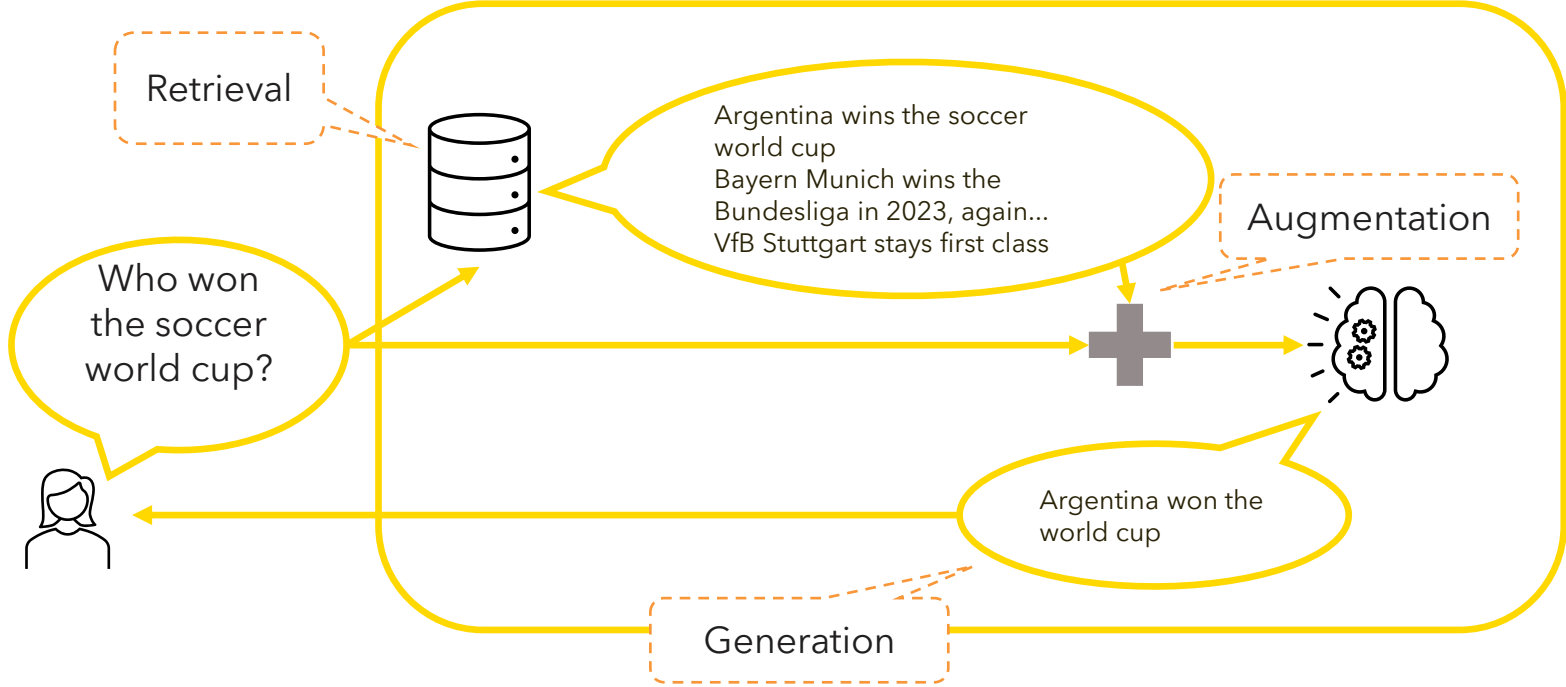


Retrieval Augmented Generation

The RAG process involves three steps:

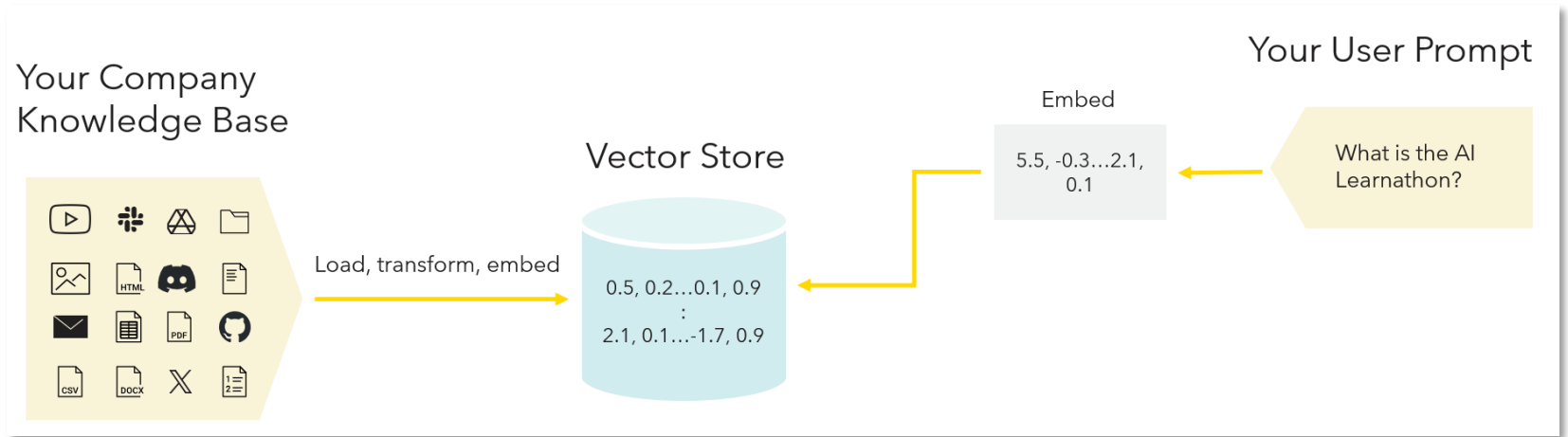
- **Retrieval.** Retrieve relevant information from a knowledge source based on the input query.
- **Augmentation.** Augment the input query or prompt with the retrieved information. This enhances the model's understanding by providing additional context from the retrieved sources.
- **Generation.** Generate a more informed and contextually rich response based on the augmented input leveraging the generative power of the model.

The RAG process visualized



How to RAG?

- We need:
 - A user prompt ✓
 - A custom Knowledge Base ✓
 - An Embedder
 - A Vector Store



What is an Embeddings model?

The Oscar for best actress goes to ...



Embeddings Model



| | | | |
|---|---|----|---|
| 5 | 7 | -2 | 3 |
|---|---|----|---|

What is an Embeddings model?

The Oscar for best actress goes to ...



Embeddings Model



| | | | |
|---|---|----|---|
| 5 | 7 | -2 | 3 |
|---|---|----|---|

The Oscar for best actress goes to ...



Embeddings Model



Argentina wins the Soccer world cup

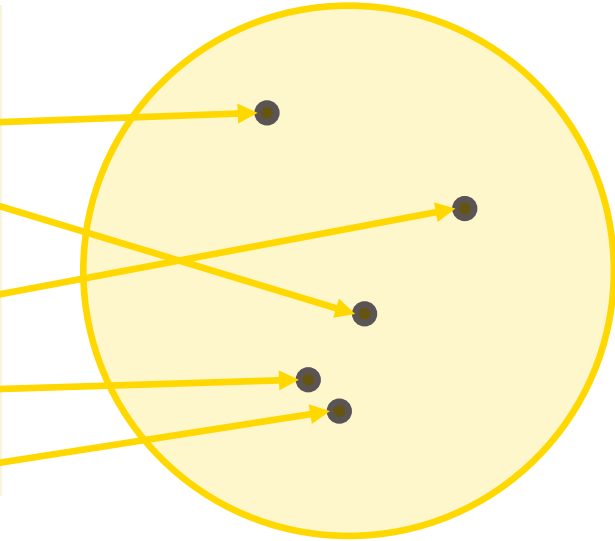


Embeddings Model



What is a Vector Store?

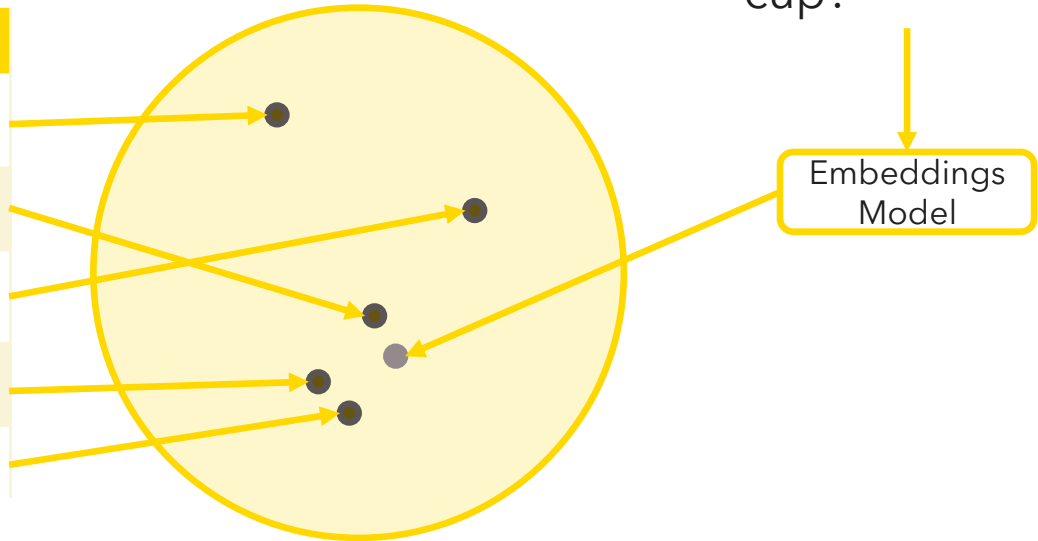
| Document | Embedding |
|---|-------------|
| The 2023 Oscar for best actress goes to ... | 5, 7, -2, 3 |
| Argentina wins the soccer world cup | 4, 6, -2, 4 |
| The Kansas City Chiefs win the superbowl in 2023 | 4, 6, 5, -1 |
| Bayern Munich wins the Bundesliga in 2023, again... | 0, 4, 8, 9 |
| VfB Stuttgart stays first class | 0, 1, 9, 8 |



How can we use it for Semantic Search?

Who won the soccer world cup?

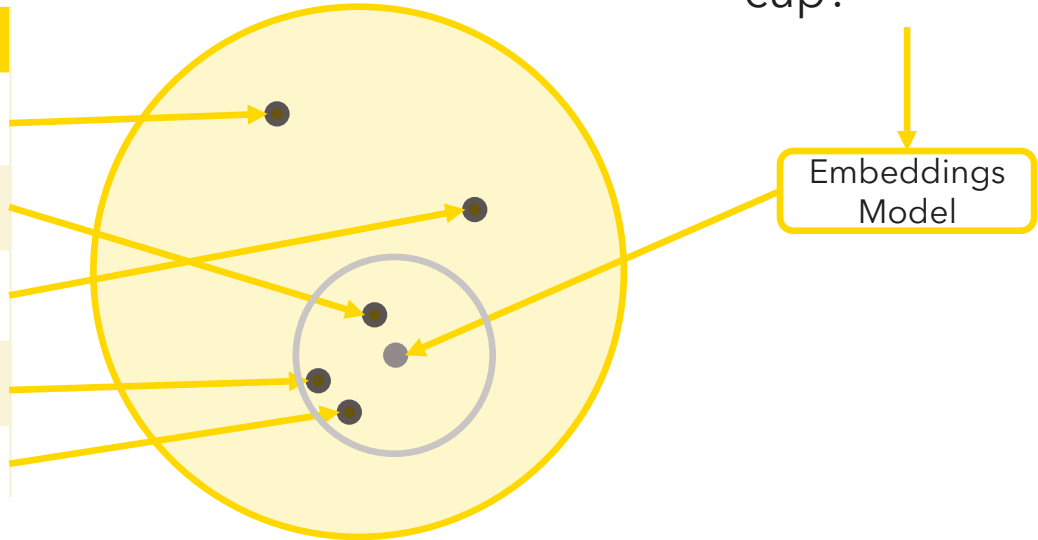
| Document | Embedding |
|---|-------------|
| The 2023 Oscar for best actress goes to ... | 5, 7, -2, 3 |
| Argentina wins the soccer world cup | 4, 6, -2, 4 |
| The Kansas City Chiefs win the superbowl in 2023 | 4, 6, 5, -1 |
| Bayern Munich wins the Bundesliga in 2023, again... | 0, 4, 8, 9 |
| VfB Stuttgart stays first class | 0, 1, 9, 8 |



How can we use it for Semantic Search?

Who won the soccer world cup?

| Document | Embedding |
|---|-------------|
| The 2023 Oscar for best actress goes to ... | 5, 7, -2, 3 |
| Argentina wins the soccer world cup | 4, 6, -2, 4 |
| The Kansas City Chiefs win the superbowl in 2023 | 4, 6, 5, -1 |
| Bayern Munich wins the Bundesliga in 2023, again... | 0, 4, 8, 9 |
| VfB Stuttgart stays first class | 0, 1, 9, 8 |

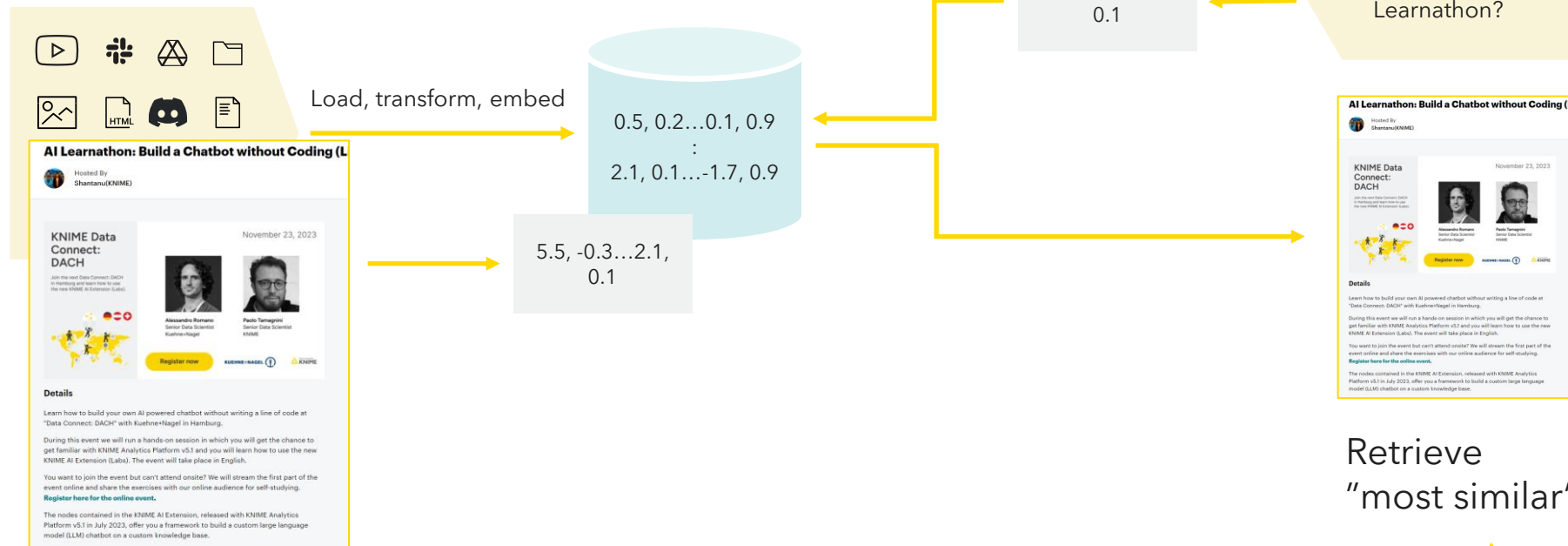


Embeddings Model

Our LLM can now answer what the KNIME AI Learnathon is!

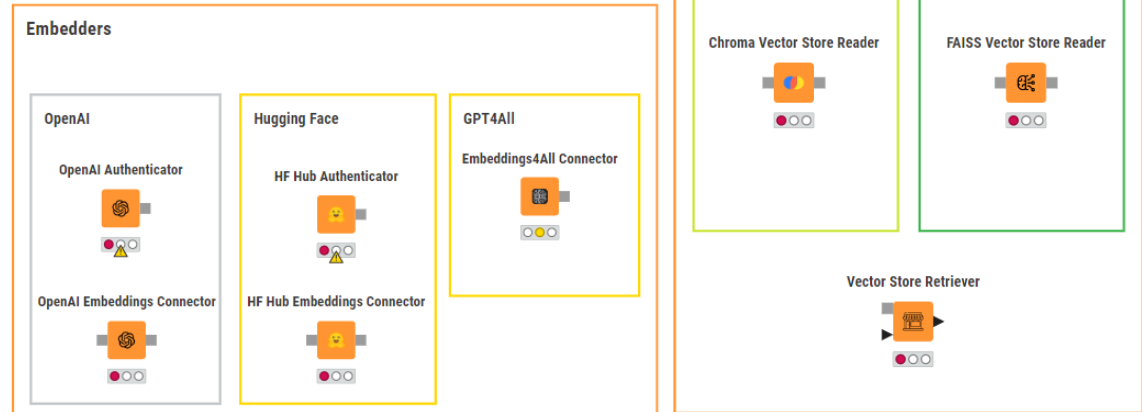
Your Company Knowledge Base

Your User Prompt



RAG in KNIME

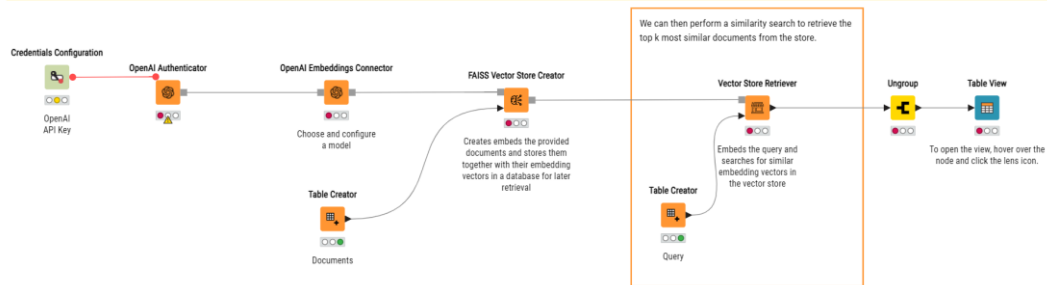
- The KNIME AI Extension has embedder and vector stores nodes!
- Easy configuration
 - Select embedding model.
 - Provide documents to embed.
 - Create vector store with embedded documents.
 - Retrieve most similar document.



RAG in KNIME

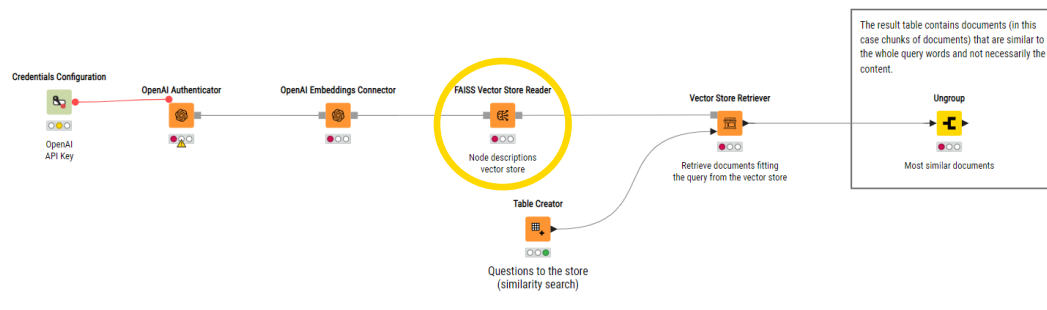
Create a Vector Store with OpenAI Embeddings

- This workflow shows how to create a vector store from a KNIME table using OpenAI Embeddings and the FAISS Vector Store Creator.
- In order to run the workflow you need an OpenAI API key. If you don't have one already, register with OpenAI and create a new API key under <https://platform.openai.com/account/api-keys>
- To learn more about the workflow, click the left bar and check the description section.



Read an existing Vector Store

- This workflow shows how to read an existing vector store into KNIME.
- In order to run the workflow you need an OpenAI API key. If you don't have one already, register with OpenAI and create a new API key under <https://platform.openai.com/account/api-keys>.
- To learn more about the workflow, click the left bar and check the description section.

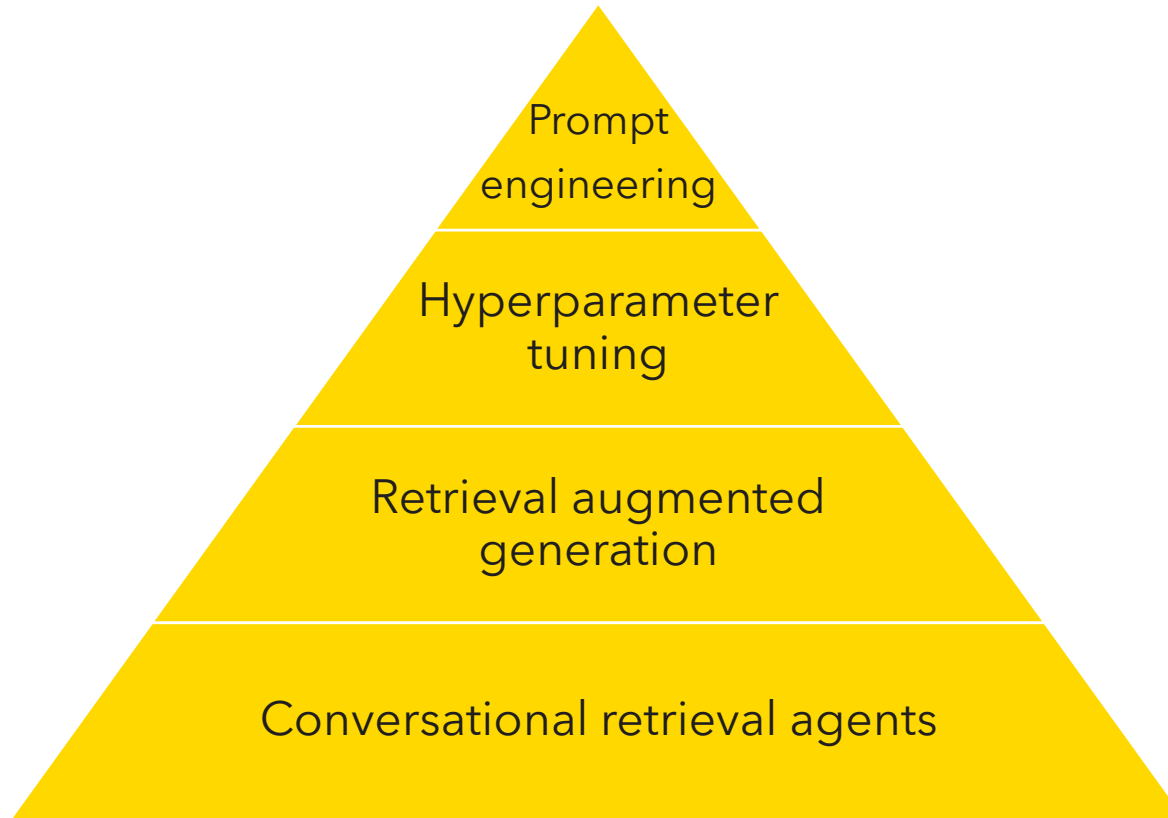


[Download the workflows](#) from the KNIME Community Hub

RAG is cool but is not *intelligent*

- RAG allows to obtain domain-specific answers but it's not smart about which tools (= vector stores), if any, to use.
- It requires a hard-coded behavior to obtain the answer we expect
- Can we do better? Can we have a truly *intelligent* system that can interact with us and choose smartly where to get answers to our questions?

Add a new layer to customize LLMs' responses

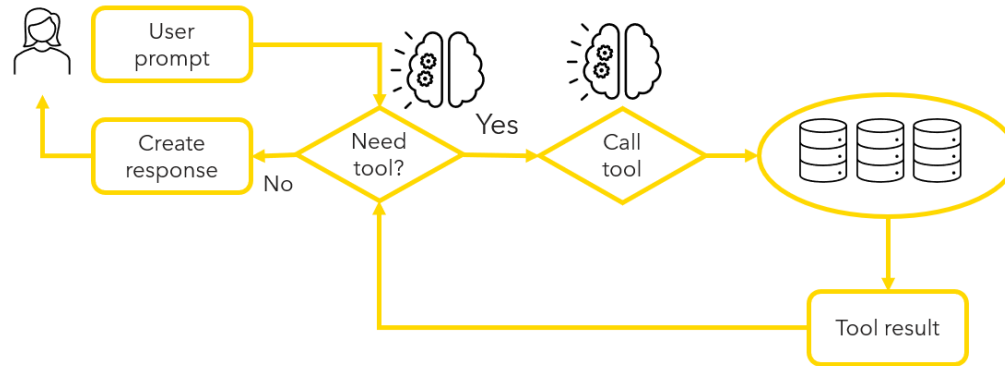


Conversational Retrieval Agents

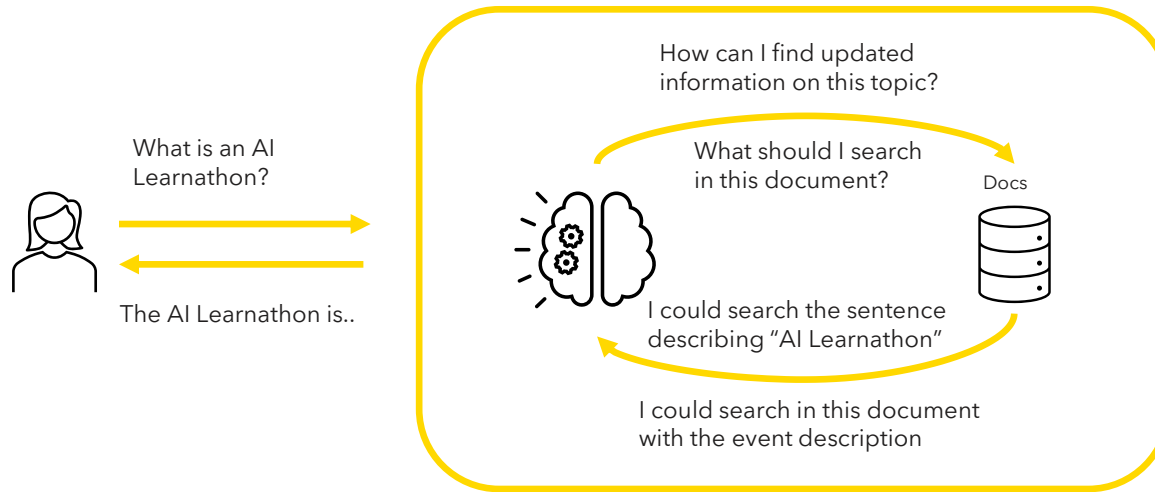
- Conversational Retrieval Agents enhance the LLM with the ability to chat and use tools to answer specific questions.
- By "agents", we mean a system where the **sequence of steps or reasoning behavior is not hard-coded**, fixed or known ahead of time, but is rather determined by a language model.
- Agents rely on the conversational capabilities of generalist LLMs but are also endowed with a suite of specialized tools (usually one or more vector stores).

Conversational Retrieval Agents

- Depending on the user's prompt and hyperparameters, **the agent understands which, if any, of these tools to employ** to best provide a compelling response.
- Agents can be instructed to perform specific functions or roles in a certain way.
 - For example, an agent can be prompted to write a political text as if it was a poet of the Renaissance or a soccer commentator.

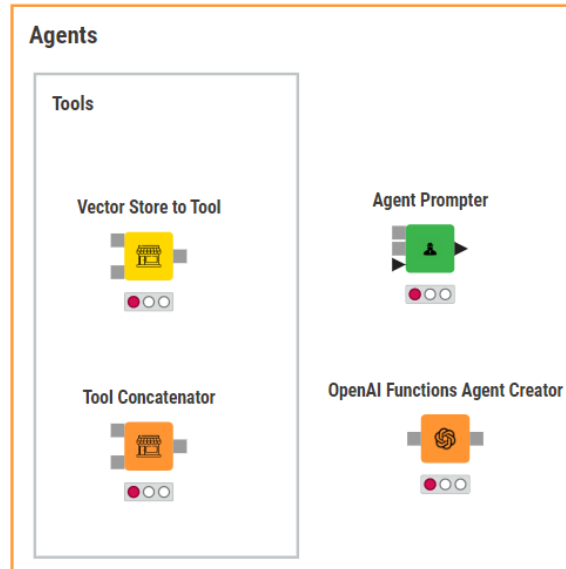


Agents: a workflow of prompts to simulate reasoning



Conversational Retrieval Agents in KNIME

- The KNIME AI Extension allows the creation of agents!
- Currently, only possible with OpenAI or Azure Open AI



Conversational Retrieval Agents in KNIME

- Decide how the agent should behave
 - Act as a...
 - How verbose answers should be.
 - Force it to always use the tool first vs. let it decide.
- Give it the tools (=vector stores)
 - Provide a tool name and description.
 - Decide how many doc should be retrieved.
- Prompt the agent!
 - Provide function and tools.
 - Provide conversation history.
 - Provide user prompt.

OpenAI Functions Agent Creator



Vector Store to Tool



Agent Prompter



OpenAI Functions Agent Creator



single document

Vector Store to Tool

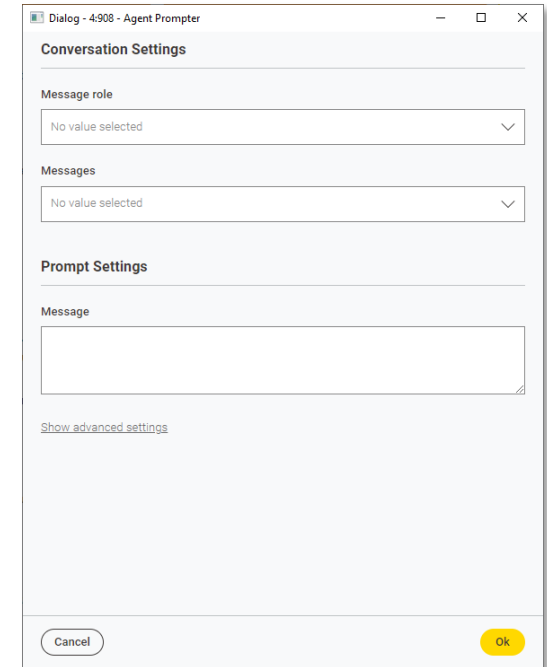


Turn the vector store into a tool

Table Reader



Conversation history



The top right corner of the slide features several overlapping, thin white lines that form a complex, abstract geometric pattern. These lines create a series of interconnected triangles and polygons, some of which are partially cut off by the edge of the frame. The overall effect is a modern, minimalist design element.

Use case: Building a Custom Chatbot

Use case: Building a Custom Chatbot

The manual instruction book for our Berlin Office coffee machine is extremely complex



GENERAL OPERATION

Selecting the Filter Size

Insert either the 1 CUP or 2 CUP filter basket into the portafilter. Press the FILTER SIZE button to select SINGLE if using the 1 CUP filter basket, or DOUBLE if using the 2 CUP filter basket.

Setting the Grind Size

The grind size will affect the rate of water flow through the coffee in the filter basket and the flavour of the espresso.

We recommend you start at setting 8 and adjust as required.

Setting the Grind Amount

Set the GRIND AMOUNT dial to the 3 o'clock position as a starting point. Adjust towards LESS or MORE as required to adjust the dose amount.

Selecting a filter basket

Use Single Wall filter baskets if grinding fresh whole coffee beans.

Use Dual Wall filter baskets if using pre-ground coffee.

Trimming the Dose

The Razor™ precision dose trimming tool allows you to trim the puck to the right level for a consistent extraction.

Autom...

- Push the...
- Grind if you has

Manua...

- Push the...
- To s...

Temp...

- On...
- As a...

the Razor™
Precision Dose Trimming Tool

Insert the Razor™ dose trimming tool into the filter basket until the shoulders of the tool rest on the rim of the basket.

Rotate the Razor™ dose trimming tool back and forth while holding the portafilter on an angle over a knock box to trim off excess coffee grinds.

Preset Shot Volume – 2 CUP button

Press the 2 CUP button once to extract a double shot of espresso at the preset volume (approx. 60ml).

Re-Programming Shot Volume – 1 CUP or 2 Cup

- To begin programming, press the Programme button once.
- Press 1 CUP or 2 CUP button to start espresso extraction. Press 1 CUP or 2 CUP button again once desired volume of espresso has been extracted. The machine will beep twice to indicate the new 1 CUP or 2 CUP volume has been set.

Resetting Default Volumes

To reset the machine to the default 1 CUP and 2 CUP shot volumes, press and hold the PROGRAMME button until the machine beeps three times. Release the PROGRAMME button.

NOTE

Default settings: 1 CUP shot volume is approx. 30ml. 2 CUP shot volume is approx. 60ml.

Manual Shot Volume

Press and hold the 1 CUP or 2 CUP button. The extraction will enter low pressure pre-infusion until the button is released. When the 1 CUP or 2 CUP button is released, extraction will continue at full pressure. Press the 1 CUP or 2 CUP button again once the desired volume of espresso has been extracted.

Pressing the 1 CUP or 2 CUP button during a preset shot volume will immediately stop the extraction.

Use case: Building a Custom Chatbot

The manual instruction book for our Berlin Office coffee machine is extremely complex

tell me about the temp control milk jug



The temperature control milk jug allows you to control the temperature of the milk for your coffee. Here are the steps to use it effectively: 1. Ensure that the milk is within the correct temperature range of 60-65°C. 2. After achieving the desired temperature, return the temperature control dial to the STANDBY position before removing the jug. 3. If the milk is not hot enough, continue heating it until the side of the jug becomes hot to touch. 4. To adjust the milk temperature, press one of the temperature adjustment buttons within 5 seconds of entering Advanced Temperature Mode. 5. The indicator strip on the milk jug will change to indicate the selected temperature. 6. It is recommended to keep the milk temperature between 60-65°C for optimal results. Let me know if you need any further assistance!


milch jug 1/2 <

ADVANCED TEMPER.


Adjusting Water Temperature

To enter Advanced Temperature Mode:

1. Press the POWER button to switch the machine off.
2. Press & hold the PROGRAMME button, then press the POWER button. The machine will beep once. The illuminated button will indicate the current temperature selected. To adjust the water temperature, press one of the following buttons within 5 seconds of entering Advanced Temperature Mode. The machine will beep twice to confirm the change. If no buttons are pressed within 5 seconds after entering Advanced Temperature Mode, no change will be made & the machine will return to STANDBY mode.




POWER



PROGRAM

- When the STEAM/HOT WATER light stops flashing, return the dial back to STANDBY position.
- Then within 8 seconds, insert the steam wand tip (1–2cm) below the surface of the milk, close to the right hand side of the jug at the 3 o'clock position and quickly turn the dial back to the STEAM position. Pausing the steam avoids the milk splashing out of the jug (this pause function lasts for 8 seconds only).
- The indicator strip on the milk jug will change colour to highlight the exact milk temperature. We recommend milk temperature to be between 60-65°C.
- Once the milk is at the correct temperature (60–65°C), return the dial to the STANDBY position BEFORE removing the jug.
- Set the jug to one side. Immediately wipe the steam wand with a damp cloth. Then, with the steam wand tip directed over the drip tray, briefly turn steam on to clear any remaining milk from inside the steam wand.
- Pour milk directly into the espresso.

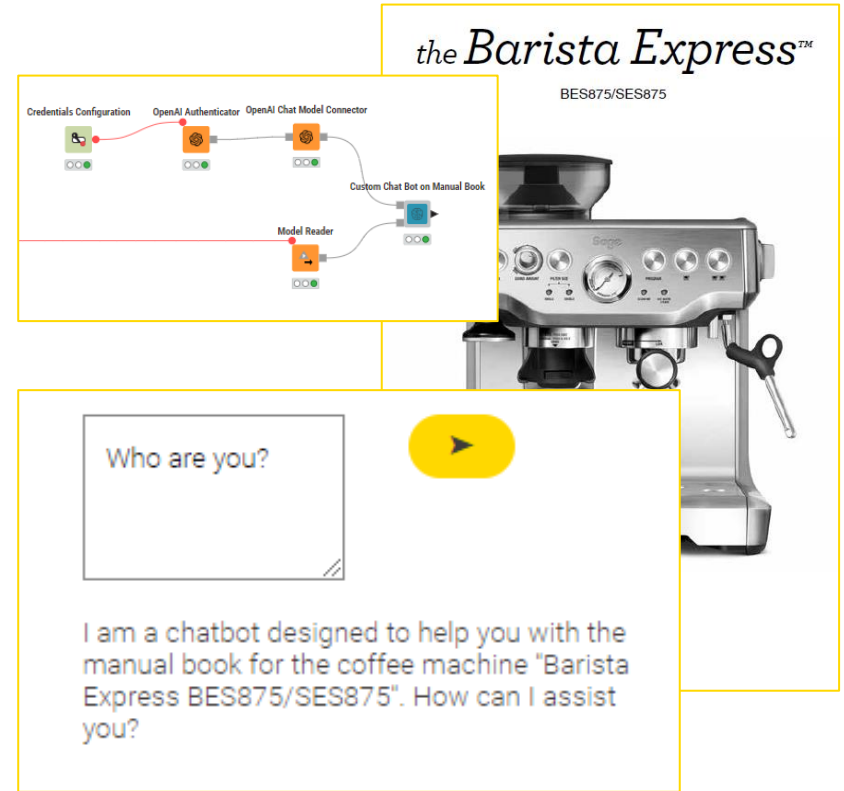
 **NOTE**

After 5 minutes of continuous steaming, the

Default Temperature setting:
In Advanced Temperature Mode, press the Programme button.

Use case: Building a Custom Chatbot

- A) Create the Knowledge Base
 - We import the PDF manual, and we create a vector store representing our knowledge base
- B) Use an LLM for a Completion Task
 - Going through a spreadsheet and automatically answer questions using the knowledge base
- C) Build Chat Bot Data App
 - We customize the LLM to play a precise role providing the vector store and a UI for the user



Use case: Building a Custom Chatbot

- **A) Create the Knowledge Base**
 - Import the manual book PDF
 - Connect to Open AI for an embedding model
 - Convert the PDF text in a list of sentences
 - Convert the sentences to vectors
 - Save the vector store
- B) Use an LLM for a Completion Task
- C) Build Chat Bot Data App

A) Create Knowledge Base

Step 1: Authenticate with OpenAI

Search, drag and drop OpenAI Authenticator, add the flow variable connection from the Credential Configuration node. Configure the node by selecting "credentials", then execute.

Step 2: Select embedding model to create vectors

Search, drag and drop OpenAI Embedding Connector node and execute with default settings. This will pick the AI embedding model "text-embedding-ada-002" to create vectors from the input sentences.

Step 3: Divide PDF text into sentences

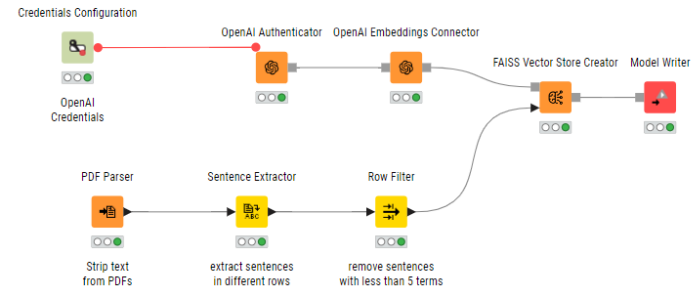
Search, drag and drop Sentence Extractor node and execute on the column "Document" from the "PDF Parser" node. This will split the document cell in multiple rows: one row for each sentence. Then use a Row Filter node to remove all sentences below 5 terms.

Step 4: Create the vector store

Search, drag and drop FAISS Vector Store Creator node, connect to the OpenAI Embeddings Connector and your string sections output. Execute the node on the column with the strings to create the vector store.

Step 5: Save the vector store

Save the vector store by adding a Model Writer node. To save properly you can use a relative path: selected "Relative to" then "Workflow Data Area" and specify the name of the vector store such as "vector_store.model"



Use case: Building a Custom Chatbot

- A) Create the Knowledge Base
- **B) Use an LLM for a Completion Task**
 - Iterate through a spreadsheet with questions
 - Find sections from the PDF via the vector store
 - Engineer a prompt via the String Manipulation
 - Call the LLM to answer the question
 - Save the answers and compared them with expected answers
- C) Build Chat Bot Data App

B) Use an LLM for a Completion Task via the Knowledge Base

Step 1: Authenticate with OpenAI and select an LLM model

Similarly to the past exercise use the **OpenAI Authenticator** with the **Credentials Configuration** flow variable input. Then add an **OpenAI Chat Model Connector** and select "gpt-3.5-turbo".

Step 2: LLM Hyperparameter Tuning

Inside the **OpenAI LLM Connector** customize the parameters "max tokens" and "temperature".

Step 3: Adopt the Knowledge Base from the Vector Store

Add a **Model Reader** node to read from the workflow data area the vector store you previously created. Reading again from disk the vector store is good practice to learn how to deploy a data app with a fixed knowledge base. If you feel lazy you can also use the output of the **FAISS Vector Store Creator** from the past exercise.

Step 4: Connect to the Vector Store Retriever and the LLM Prompter

Drag in the **Vector Store Retriever** node and the **LLM Prompter** node, in-between add a **String Manipulation** node for Prompt Engineering.

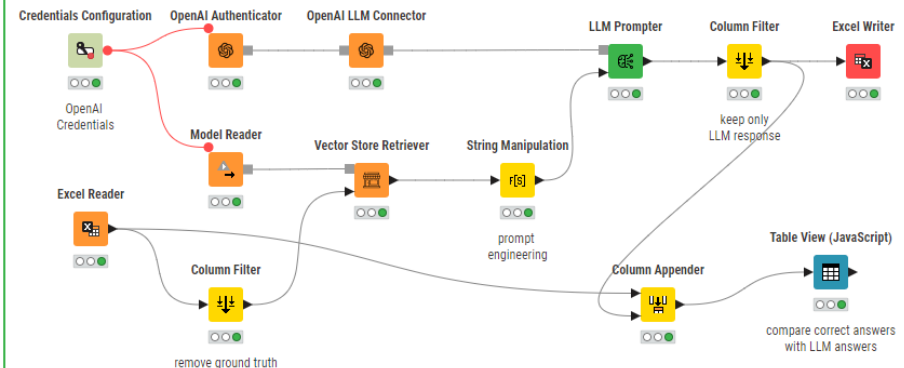
Prompt Engineering Hint



double click
to open

Step 5: Export Results to an Excel File

You can save the table via an **Excel Writer** node. Optionally you can compare with a **Table View** node the answers by the LLM and the ones we imported with the questions for reference.



Use case: Building a Custom Chatbot

- A) Create the Knowledge Base
- B) Use an LLM for a Completion Task
- **C) Build Chat Bot Data App**
 - Hyperparameter Tuning
 - Character/Prompt Engineering
 - Vector Store to Tool
 - Set up Agent
 - Build and Customize UI

C) Create Chat Bot Data App on Knowledge Base

Step 1: Authenticate with OpenAI and select a chat model

Similarly to the past exercise use the OpenAI Authenticator with the Credentials Configuration flow variable input. Then add an OpenAI Chat Model Connector and select "gpt-3.5-turbo".

Step 2: Chat Model Hyperparameter Tuning

Inside the OpenAI Chat Model Connector customize the parameters.

Step 3: Adopt the Knowledge Base from the Vector Store

Add a Model Reader node to read from the workflow data area the vector store is good practice to learn how to deploy.

Step 4: Connect to the input of the Component

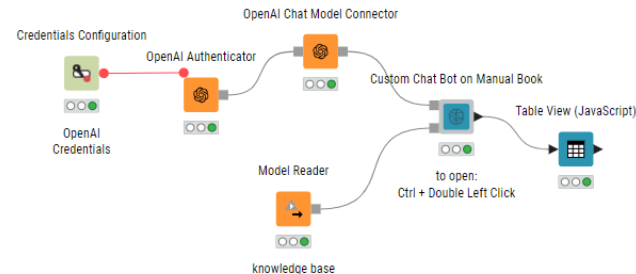
Connect the Connector output to the top input of the component, then the Model Reader output to the bottom input.

Then open the component (Ctrl + Double Left Click on it) for the next step.





Who are you?



I am a chatbot designed to help you with the manual book for the coffee machine "Barista Express BES875/SES875". How can I assist you?



Other possible use cases

-  Legal Expert Chat Bot:
 - **Knowledge base:** documents describing previous court cases
 - **Goal:** to help lawyers to browse long documents
-  Customer Care Chat Bot:
 - **Knowledge base:** emails and forum activity of previous questions AND/OR product documentation
 - **Goal:** To help customers with their technical issues
-  Healthcare Expert Chat Bot:
 - **Knowledge base:** patients records with previous clinical results and outcome
 - **Goal:** To help doctors to quickly find similar cases and how they were treated
-  Machinery Expert Chat Bot:
 - **Knowledge base:** the latest updated instruction manual book describing components and practices
 - **Goal:** To help end-users or engineers to quickly find information on maintenance and usage

(Optional) Exercise 02: Building a Custom Chatbot

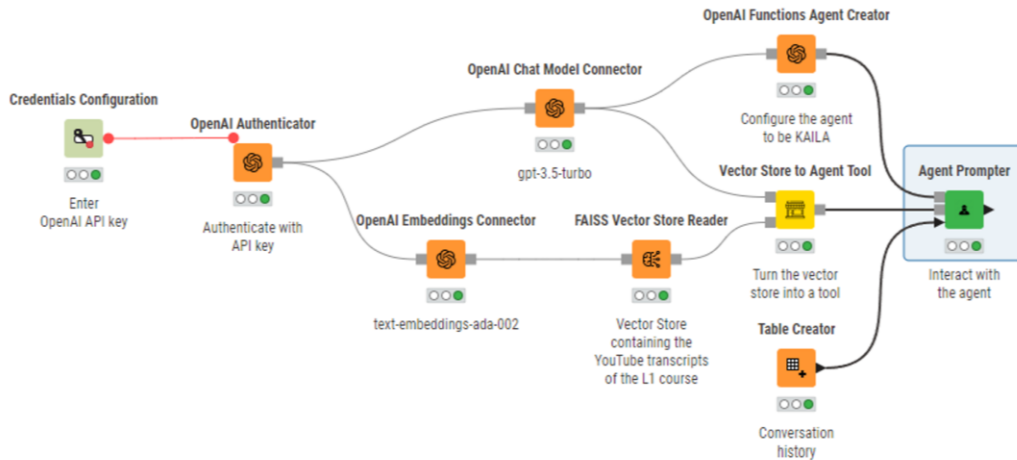
- Exercise 02: Build a Chatbot on a Knowledge base to answer questions about the coffee machine
 - Create the Knowledge Base
 - Use an LLM for a Completion Task
 - Build Chat Bot Data App
- Find the exercise workflow *02_Build a Chatbot on a Knowledge base* in *Day 4 > AI*
- Temporary OpenAI API keys are available in *Day 4 > AI* in *ADILT-2023_OpenAI API key.txt*

Wrap-up



What we have learnt

- What LLMs are, how they think, key factors behind their power, and risks & limitations
- How LLMs can be leveraged in KNIME via the KNIME AI Extension
- 4 levels to customize LLM responses: from prompt engineering to conversational retrieval agents
- Use cases: from simple text generation to building a custom chatbot
- Readings on KNIME Blog:
 - Intro to LLMs → <https://www.knime.com/blog/friendly-intro-to-large-language-models>
 - Customize LLM responses → <https://www.knime.com/blog/4-levels-llm-customization>



You're now all set to conquer the world with AI! 😊