The LangChain ecosystem

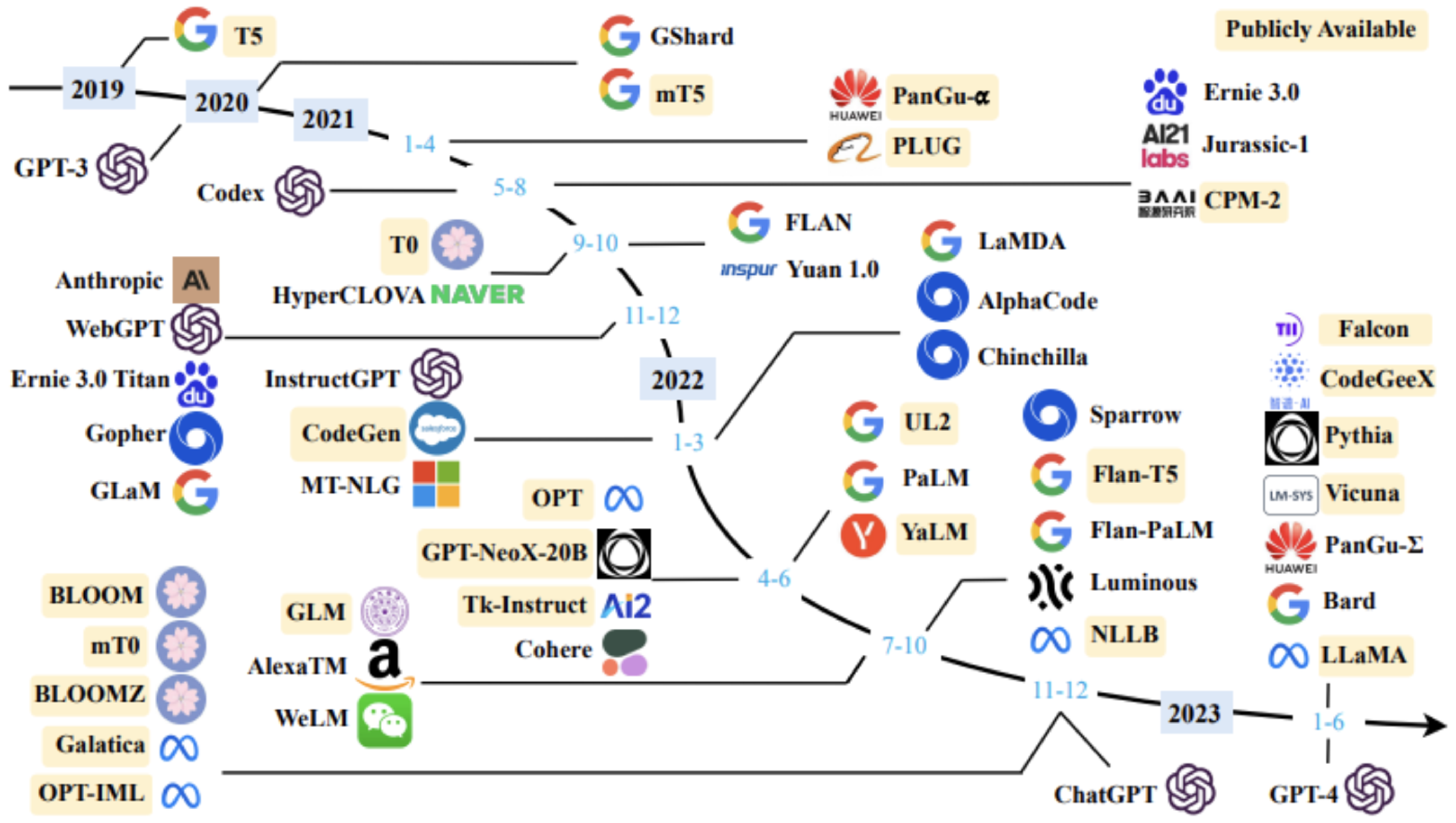
DEVELOPING LLM APPLICATIONS WITH LANGCHAIN

Jonathan Bennion
AI Engineer & LangChain Contributor

Meet your instructor...

- Jonathan Bennion, **AI Engineer**
- ML & AI at Facebook, Google, Amazon, Disney, EA
- Created *Logical Fallacy chain* in LangChain
- Contributor to **DeepEval**

The state of Large Language Models (LLMs)

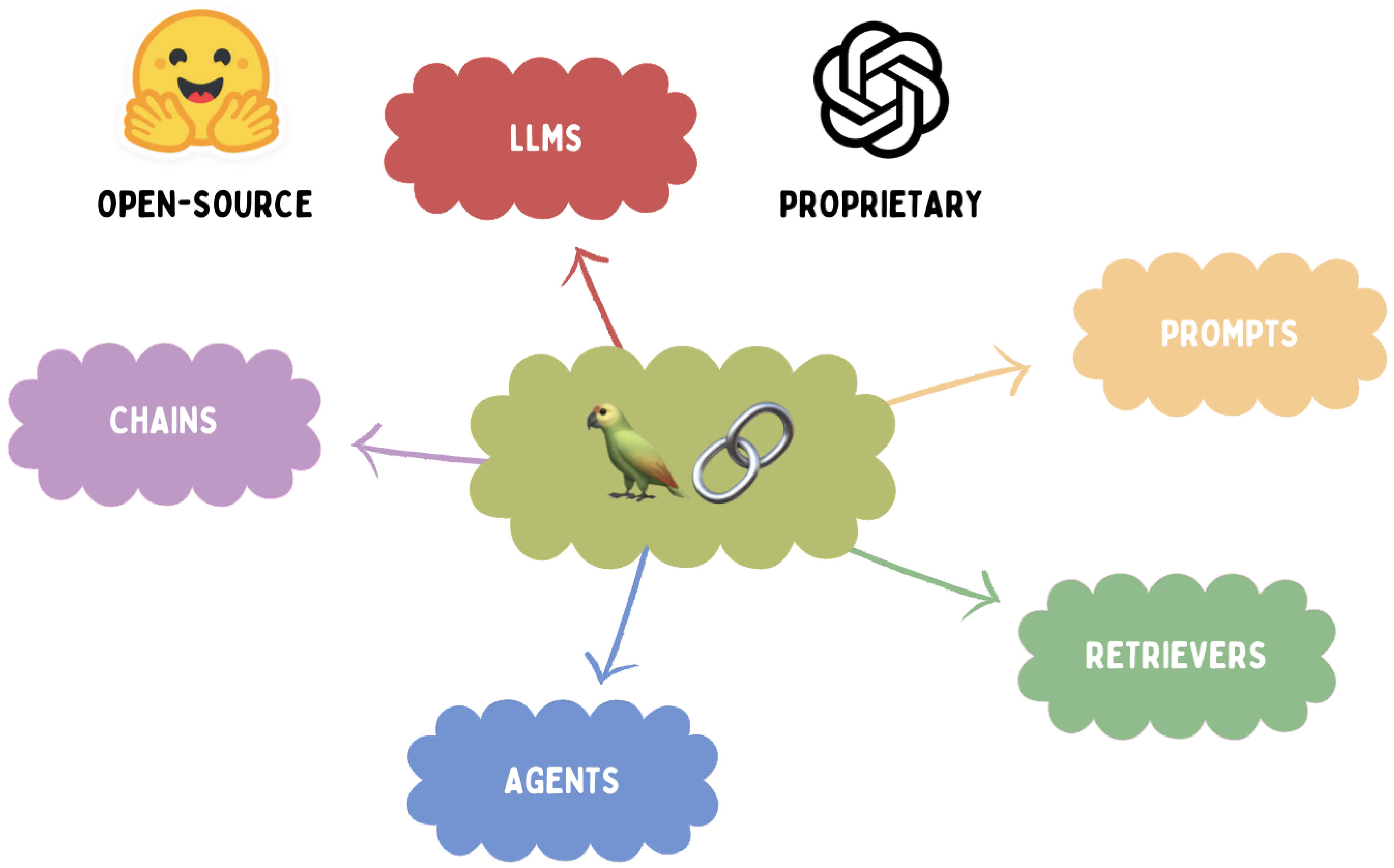


[1] https://arxiv.org/pdf/2303.18223.pdf

What is LangChain?

- **Open-source** framework for connecting:
  - Large Language Models (LLMs)
  - Data sources
  - Other functionality under a **unified syntax**
- Allows for scalability
- Contains modular components
- Supports **Python** and **JavaScript**

Core components of LangChain



Hugging Face

- **Open-source** repository of models, datasets, and tools

Creating a Hugging Face API key:

1. Sign up for a Hugging Face account
2. Navigate to `https://huggingface.co/settings/tokens`
3. Select New token and copy the key

Hugging Face (*Falcon-7b*):

```python
from langchain_huggingface import HuggingFaceEndpoint

llm = HuggingFaceEndpoint(
    repo_id='tiiuae/falcon-7b-instruct',
    huggingfacehub_api_token=huggingfacehub_api_token
)

question = 'Can you still have fun'
output = llm.invoke(question)

print(output)
```

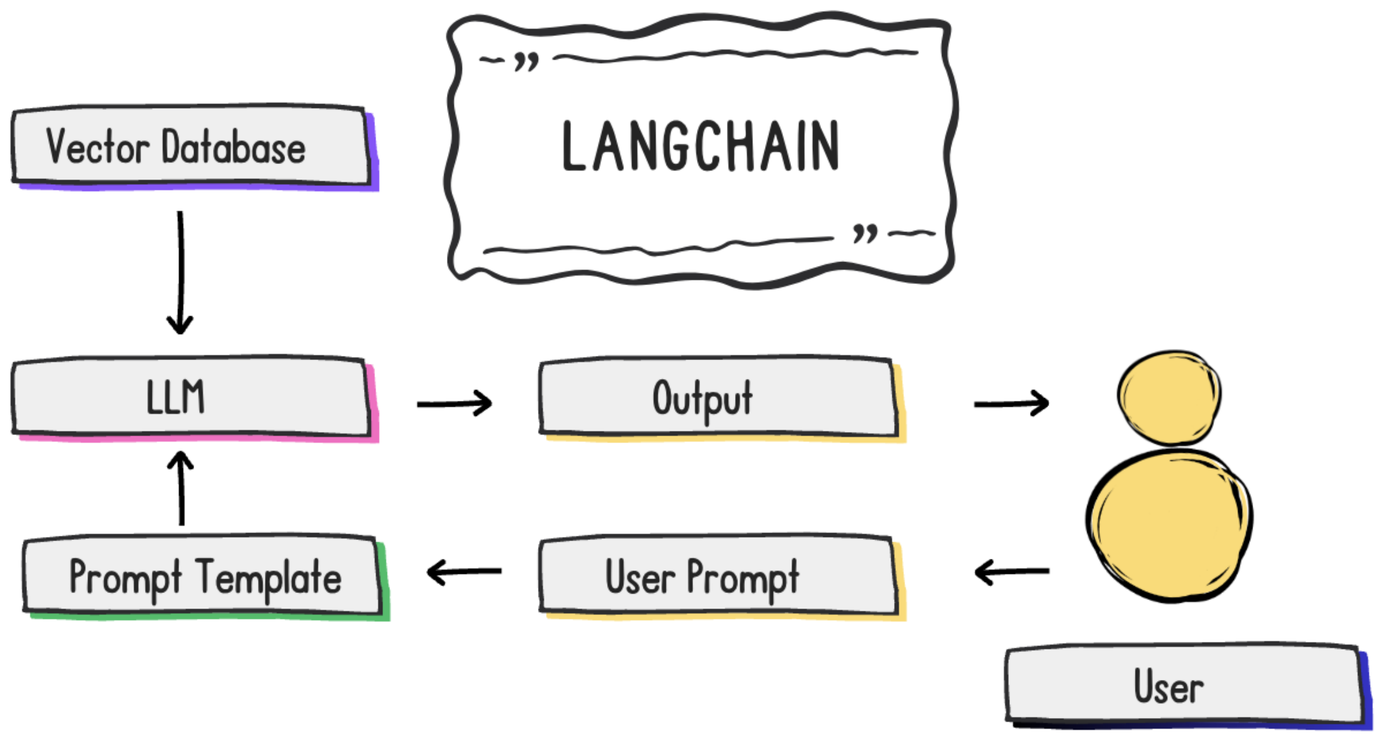
```
 in the rain?
Yes, you can still have fun in the
rain! There are plenty of
```

OpenAI (`gpt-3.5-turbo-instruct`):

```python
from langchain_openai import OpenAI

llm = OpenAI(
    model="gpt-3.5-turbo-instruct",
    api_key=openai_api_key
)


question = 'Can you still have fun'
output = llm.invoke(question)

print(output)
```

```
 without spending a lot of money?

Yes, you can still have fun without
spending a lot of money. You could do
activities like hiking, biking, playing
sports, going to the beach, camping...
```

Real-world usage



Examples:

- Natural language conversations with documents
- Automate tasks
- Data analysis

Note: course uses `langchain==0.3.13`

Let's practice!

DEVELOPING LLM APPLICATIONS WITH LANGCHAIN

Prompting strategies for chatbots

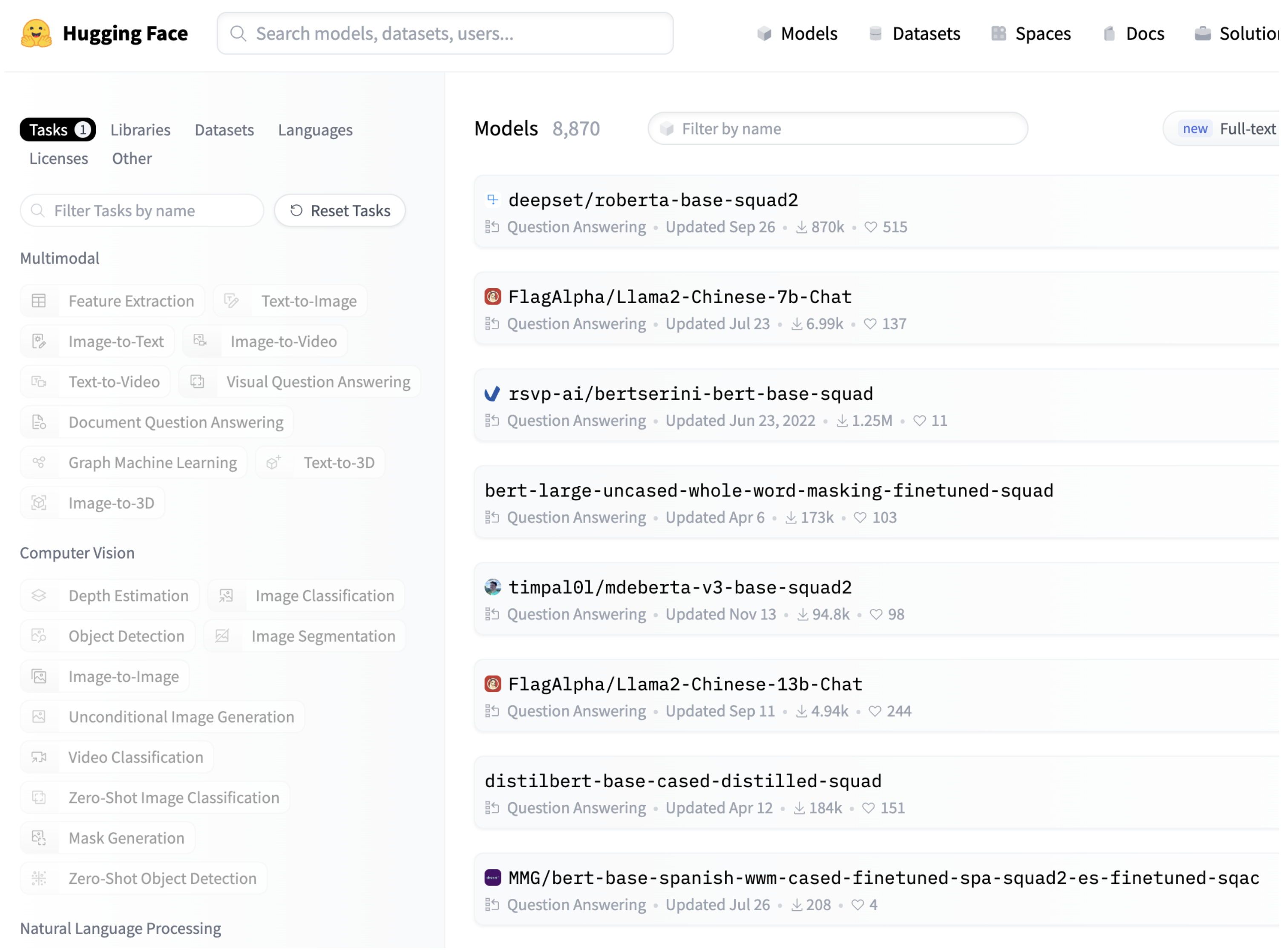
DEVELOPING LLM APPLICATIONS WITH LANGCHAIN

Jonathan Bennion
AI Engineer & LangChain Contributor

Finding the right model

Hugging Face is a searchable hub for chat models

- *Fine-tuned models* for more domain-specific use cases



Hugging Face | Search models, datasets, users... | Models | Datasets | Spaces | Docs | Solutio

Tasks 1 | Libraries | Datasets | Languages | Licenses | Other

Filter Tasks by name | Reset Tasks

Multimodal

Feature Extraction | Text-to-Image | Image-to-Text | Image-to-Video | Text-to-Video | Visual Question Answering | Document Question Answering | Graph Machine Learning | Text-to-3D | Image-to-3D

Computer Vision

Depth Estimation | Image Classification | Object Detection | Image Segmentation | Image-to-Image | Unconditional Image Generation | Video Classification | Zero-Shot Image Classification | Mask Generation | Zero-Shot Object Detection

Natural Language Processing

Models 8,870 | Filter by name | new Full-text

deepset/roberta-base-squad2
Question Answering • Updated Sep 26 • 870k • 515

FlagAlpha/Llama2-Chinese-7b-Chat
Question Answering • Updated Jul 23 • 6.99k • 137

rsvp-ai/bertserini-bert-base-squad
Question Answering • Updated Jun 23, 2022 • 1.25M • 11

bert-large-uncased-whole-word-masking-finetuned-squad
Question Answering • Updated Apr 6 • 173k • 103

timpal0l/mdeberta-v3-base-squad2
Question Answering • Updated Nov 13 • 94.8k • 98

FlagAlpha/Llama2-Chinese-13b-Chat
Question Answering • Updated Sep 11 • 4.94k • 244

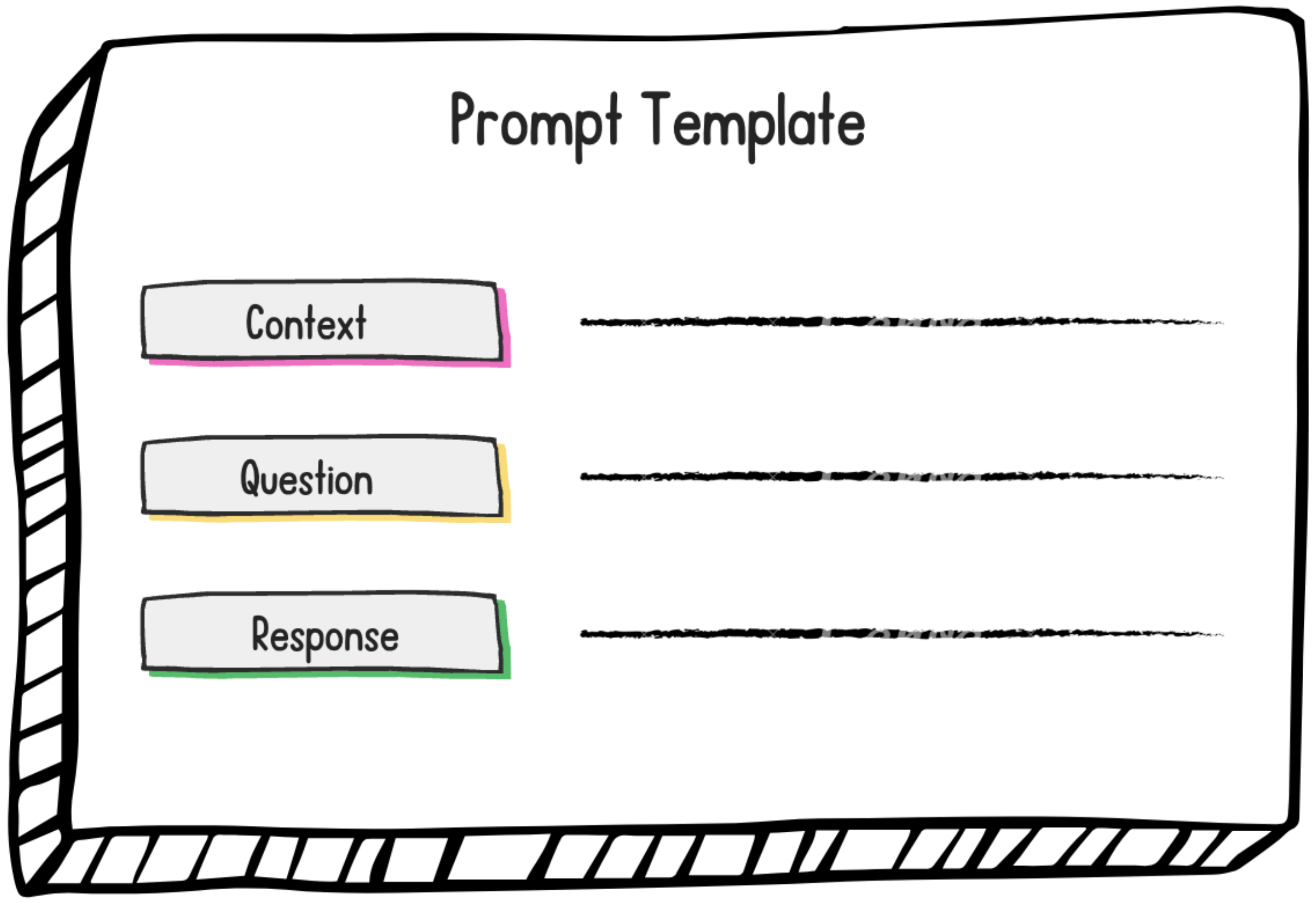
distilbert-base-cased-distilled-squad
Question Answering • Updated Apr 12 • 184k • 151

MMG/bert-base-spanish-wwm-cased-finetuned-spa-squad2-es-finetuned-sqac
Question Answering • Updated Jul 26 • 208 • 4

[1] https://huggingface.co/models?pipeline_tag=question-answering&sort=trending

Prompt templates

- **Recipes** for generating prompts
- *Flexible* and *modular*
- Can contain: instructions, examples, and additional context



Prompt templates

```python
from langchain_core.prompts import PromptTemplate

template = "You are an artificial intelligence assistant, answer the question. {question}"
prompt_template = PromptTemplate(template=template, input_variables=["question"])

print(prompt_template.invoke({"question": "What is LangChain?"}))
```

```
text='You are an artificial intelligence assistant, answer the question. What is LangChain?'
```

Integrating PromptTemplate with LLMs

```
from langchain_huggingface import HuggingFaceEndpoint

llm = HuggingFaceEndpoint(repo_id='tiiuae/falcon-7b-instruct', huggingfacehub_api_token=huggingfacehub_api_toke
llm_chain = prompt_template | llm

question = "What is LangChain?"
print(llm_chain.invoke({"question": question}))
```

```
LangChain is an artificial intelligence language model that uses a neural network to generate human-like text
```

- LangChain Expression Language (**LCEL**)
- **Chain:** connect calls to different components

Chat models

```python
from langchain_core.prompts import ChatPromptTemplate

prompt_template = ChatPromptTemplate.from_messages(
    [
        ("system", "You are soto zen master Roshi."),
        ("human", "What is the essence of Zen?"),
        ("ai", "When you are hungry, eat. When you are tired, sleep.")
        ("human", "Respond to the question: {question}")
    ]
)
```

Integrating ChatPromptTemplate

```python
from langchain_openai import ChatOpenAI

llm = ChatOpenAI(model="gpt-4o-mini", api_key=openai_api_key)

llm_chain = prompt_template | llm
question='What is the sound of one hand clapping?'

response = llm_chain.invoke({"question": question})
print(response.content)
```

```
The sound of one hand clapping is not something that can be easily explained or
understood through words alone. It is a question that has been pondered by Zen
practitioners for centuries, and its purpose is to provoke a deeper inquiry into
the nature of reality and the self. In Zen practice, we often engage...
```

Let's practice!

DEVELOPING LLM APPLICATIONS WITH LANGCHAIN

Few-shot prompting

DEVELOPING LLM APPLICATIONS WITH LANGCHAIN

Jonathan Bennion
AI Engineer & LangChain Contributor

Limitations of standard prompt templates

- `PromptTemplate` + `ChatPromptTemplate`
-   Handling small numbers of examples
-   Don't scale to larger numbers
- `FewShotPromptTemplate`

```
examples = [
    {
        "question": "..."
        "answer": "..."
    },
    ...
]
```

Building an example set

```
examples = [
    {
        "question": "Does Henry Campbell have any pets?",
        "answer": "Henry Campbell has a dog called Pluto."
    },
    ...
]
```

```
# Convert DataFrame to list of dicts
examples = df.to_dict(orient="records")
```

Formatting the examples

```python
from langchain_core.prompts import FewShotPromptTemplate, PromptTemplate

example_prompt = PromptTemplate.from_template("Question: {question}\n{answer}")
```

```python
prompt = example_prompt.invoke({"question": "What is the capital of Italy?"
                                "answer": "Rome"})
print(prompt.text)
```

```
Question: What is the capital of Italy?
Rome
```

FewShotPromptTemplate

```
prompt_template = FewShotPromptTemplate(
    examples=examples,
    example_prompt=example_prompt,
    suffix="Question: {input}",
    input_variables=["input"]
)
```

- `examples` : the list of dicts
- `example_prompt` : formatted template
- `suffix` : suffix to add to the input
- `input_variables`

Invoking the few-shot prompt template

```python
prompt = prompt_template.invoke({"input": "What is the name of Henry Campbell's dog?"})
print(prompt.text)
```

```
Question: Does Henry Campbell have any pets?
Henry Campbell has a dog called Pluto.
...

Question: What is the name of Henry Campbell's dog?
```

Integration with a chain

```python
llm = ChatOpenAI(model="gpt-4o-mini", api_key="...")

llm_chain = prompt_template | llm
response = llm_chain.invoke({"input": "What is the name of Henry Campbell's dog?"})
print(response.content)
```

```
The name of Henry Campbell's dog is Pluto.
```

Let's practice!

DEVELOPING LLM APPLICATIONS WITH LANGCHAIN