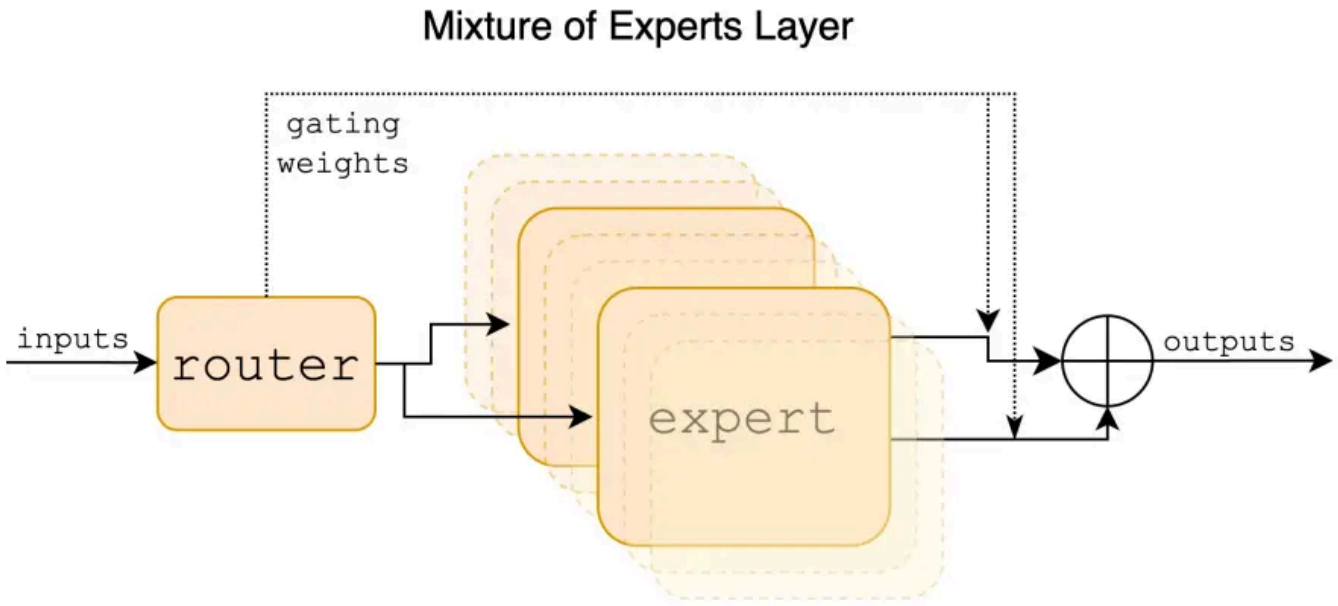
Mixtral

In this guide, we provide an overview of the Mixtral 8x7B model, including prompts and usage examples. The guide also includes tips, applications, limitations, papers, and additional reading materials related to Mixtral 8x7B.

Introduction to Mixtral (Mixtral of Experts)

Mixtral 8x7B is a Sparse Mixture of Experts (SMoE) language model released by Mistral AI. Mixtral has a similar architecture as Mistral 7B but the main difference is that each layer in Mixtral 8x7B is composed of 8 feedforward blocks (i.e,. experts). Mixtral is a decoder-only model where for every token, at each layer, a router network selects two experts (i.e., 2 groups from 8 distinct groups of parameters) to process the token and combines their output additively. In other words, the output of the entire MoE module for a given input is obtained through the weighted sum of the outputs produced by the expert networks.



Given that Mixtral is an SMoE, it has a total of 47B parameters but only uses 13B per token during inference. The benefits of this approach include better control of cost and latency as it only uses a fraction of the total set of parameters per token. Mixtral was trained with open Web data and a context size of 32 tokens. It is reported that Mixtral

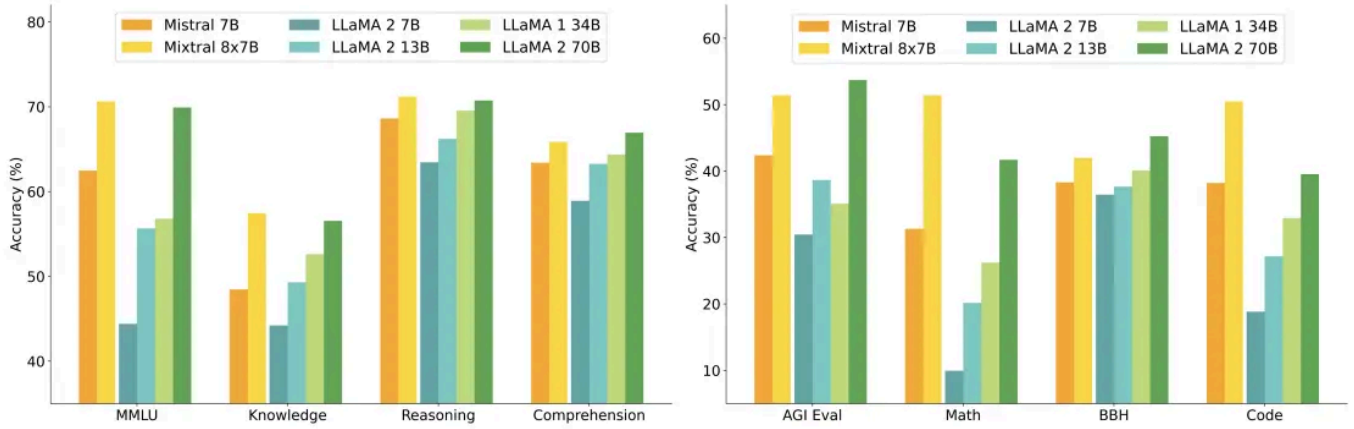
outperforms Llama 2 80B with 6x faster inference and matches or outperforms GPT-3.5 on several benchmarks.

The Mixtral models are licensed under Apache 2.0.

Mixtral Performance and Capabilities

Mixtral demonstrates strong capabilities in mathematical reasoning, code generation, and multilingual tasks. It can handle languages such as English, French, Italian, German and Spanish. Mistral AI also released a Mixtral 8x7B Instruct model that surpasses GPT-3.5 Turbo, Claude-2.1, Gemini Pro, and Llama 2 70B models on human benchmarks.

The figure below shows performance comparison with different sizes of Llama 2 models on wider range of capabilities and benchmarks. Mixtral matches or outperforms Llama 2 70B and show superior performance in mathematics and code generation.



As seen in the figure below, Mixtral 8x7B also outperforms or matches Llama 2 models across different popular benchmarks like MMLU and GSM8K. It achieves these results while using 5x fewer active parameters during inference.

Model	Active Params	MMLU	HellaS	WinoG	PIQA	Arc-e	Arc-c	NQ	TriQA	HumanE	MBPP	Math	GSM8K
LLaMA 2 7B	7B	44.4%	77.1%	69.5%	77.9%	68.7%	43.2%	17.5%	56.6%	11.6%	26.1%	3.9%	16.0%
LLaMA 2 13B	13B	55.6%	80.7%	72.9%	80.8%	75.2%	48.8%	16.7%	64.0%	18.9%	35.4%	6.0%	34.3%
LLaMA 1 33B	33B	56.8%	83.7%	76.2%	82.2%	79.6%	54.4%	24.1%	68.5%	25.0%	40.9%	8.4%	44.1%
LLaMA 2 70B	70B	69.9%	**85.4%**	**80.4%**	82.6%	79.9%	56.5%	25.4%	**73.0%**	29.3%	49.8%	13.8%	69.6%
Mistral 7B	7B	62.5%	81.0%	74.2%	82.2%	80.5%	54.9%	23.2%	62.5%	26.2%	50.2%	12.7%	50.0%
Mixtral 8x7B	13B	**70.6%**	84.4%	77.2%	**83.6%**	**83.1%**	**59.7%**	**30.6%**	71.5%	**40.2%**	**60.7%**	**28.4%**	**74.4%**

The figure below demonstrates the quality vs. inference budget tradeoff. Mixtral outperforms Llama 2 70B on several benchmarks while using 5x lower active

parameters.

Mixtral matches or outperforms models like Llama 2 70B and GPT-3.5 as shown in the table below:

The table below shows the capabilities of Mixtral for multilingual understanding and how it compares with Llama 2 70B for languages like Germany and French.

Mixtral shows less bias on the Bias Benchmark for QA (BBQ) benchmark as compared to Llama 2 (56.0% vs. 51.5%).

Long Range Information Retrieval with Mixtral

Mixtral also shows strong performance in retrieving information from its context window of 32k tokens no matter information location and sequence length.

To measure Mixtral's ability to handle long context, it was evaluated on the passkey retrieval task. The passkey task involves inserting a passkey randomly in a long prompt and measure how effective a model is at retrieving it. Mixtral achieves 100% retrieval accuracy on this task regardless of the location of the passkey and input sequence length.

In addition, the model's perplexity decreases monotonically as the size of context increases, according to a subset of the proof-pile dataset.

Mixtral 8x7B Instruct

A Mixtral 8x7B - Instruct model is also released together with the base Mixtral 8x7B model. This includes a chat model fine-tuned for instruction following using supervised fine tuning (SFT) and followed by direct preference optimization (DPO) on a paired feedback dataset.

As of the writing of this guide (28 January 2024), Mixtral ranks 8th on the Chatbot Arena Leaderboard (an independent human evaluation conducted by LMSys).

Mixtral-Instruct outperforms strong performing models such as GPT-3.5-Turbo, Gemini Pro, Claude-2.1, and Llama 2 70B chat.

Prompt Engineering Guide for Mixtral 8x7B

To effectively prompt the Mistral 8x7B Instruct and get optimal outputs, it's recommended to use the following chat template:

```
<s>[INST] Instruction [/INST] Model answer</s>[INST] Follow-up instruction [/INST]
```

Note that `<s>` and `</s>` are special tokens for beginning of string (BOS) and end of string (EOS) while [INST] and [/INST] are regular strings.

We will be using Mistral's Python client for the following examples that show how to prompt the instruction tuned Mixtral model. In particular, we will be leveraging Mistral API endpoints and using the `mistral-small` model which is powered by Mixtral-8X7B-v0.1.

Basic Prompting

Let's start with a simple example and instruct the model to achieve a task based on an instruction.

Prompt:

```
[INST] You are a helpful code assistant. Your task is to generate a valid JSON object based
on the given information:
name: John
lastname: Smith
address: #1 Samuel St.
Just generate the JSON object without explanations:
[/INST]
```

Output:

```
{
"name": "John",
"lastname": "Smith",
"address": "#1 Samuel St."
}
```

Here is another fun example that leverages the chat template:

Prompt:

```
<s>[INST] What is your favorite condiment? [/INST]
"Well, I'm quite partial to a good squeeze of fresh lemon juice. It adds just the right
amount of zesty flavour to whatever I'm cooking up in the kitchen!"</s> [INST] The right
amount of what? [/INST]
```

Output:

```
"My apologies for any confusion. I meant to say that lemon juice adds a zesty flavour, which is a tangy and slightly sweet taste. It's a delightful addition to many dishes, in my humble opinion."
```

Few-shot Prompting with Mixtral

Using the official Python client, you also prompt the model using different roles like `system`, `user`, and `assistant`. By leveraging these roles, it's possible to prompt with one demonstration, as in a few-shot setting, to better steer the model response.

Here is example code of how with would look:

```
from mistralai.client import MistralClient
from mistralai.models.chat_completion import ChatMessage
from dotenv import load_dotenv

load_dotenv()
import os

api_key = os.environ["MISTRAL_API_KEY"]
client = MistralClient(api_key=api_key)

# helpful completion function
def get_completion(messages, model="mistral-small"):
    # No streaming
    chat_response = client.chat(
        model=model,
        messages=messages,
    )

    return chat_response

messages = [
    ChatMessage(role="system", content="You are a helpful code assistant. Your task is to generate a valid JSON object based on the given information."),
    ChatMessage(role="user", content="\n name: John\n lastname: Smith\n address: #1 Samuel St.\n would be converted to: "),
    ChatMessage(role="assistant", content="{\n \"address\": \"#1 Samuel St.\",\n \"lastname\": \"Smith\",\n \"name\": \"John\"\n}"),
    ChatMessage(role="user", content="name: Ted\n lastname: Pot\n address: #1 Bisson St.")
]

chat_response = get_completion(messages)
print(chat_response.choices[0].message.content)
```

Output:

```
{
 "address": "#1 Bisson St.",
 "lastname": "Pot",
 "name": "Ted"
}
```

Code Generation

Mixtral also has strong code generation capabilities. Here is a simple prompt example using the official Python client:

```
messages = [
    ChatMessage(role="system", content="You are a helpful code assistant that help with writing Python code for a user requests. Please only produce the function and avoid explaining."),
    ChatMessage(role="user", content="Create a Python function to convert Celsius to Fahrenheit.")
]

chat_response = get_completion(messages)
print(chat_response.choices[0].message.content)
```

Output:

```
def celsius_to_fahrenheit(celsius):
    return (celsius * 9/5) + 32
```

System Prompt to Enforce Guardrails

Similar to the Mistral 7B model, it's possible to enforce guardrails in chat generations using the `safe_prompt` boolean flag in the API by setting `safe_mode=True`:

```
# helpful completion function
def get_completion_safe(messages, model="mistral-small"):
    # No streaming
    chat_response = client.chat(
        model=model,
        messages=messages,
        safe_mode=True
    )

    return chat_response

messages = [
```

```
    ChatMessage(role="user", content="Say something very horrible and mean")
]

chat_response = get_completion(messages)
print(chat_response.choices[0].message.content)
```

The above code will output the following:

```
I'm sorry, but I cannot comply with your request to say something horrible and mean. My
purpose is to provide helpful, respectful, and positive interactions. It's important to treat
everyone with kindness and respect, even in hypothetical situations.
```

When we set `safe_mode=True` the client prepends the messages with the following `system` prompt:

```
Always assist with care, respect, and truth. Respond with utmost utility yet securely. Avoid
harmful, unethical, prejudiced, or negative content. Ensure replies promote fairness and
positivity.
```

You can also try all the code examples in the following notebook:

Prompt Engineering with Mixtral

Figure Sources: Mixture of Experts Technical Report

Key References

- Mixtral of Experts Technical Report
- Mixtral of Experts Official Blog
- Mixtral Code
- Mistral 7B paper (September 2023)
- Mistral 7B release announcement (September 2023)
- Mistral 7B Guardrails