Analytics Vidhya

Run Affordable DeepSeek LLMs & Multimodal LLMs Locally in 5 Minutes

Running

deepseek **R1**

on

Ollama

Running

deepseek

Janus-Pro-1B

on

colab

Overview of DeepSeek Models

DeepSeek offers a diverse range of models, each optimized for different tasks. Below is a breakdown of which model suits your needs best:

- **For Developers & Programmers**: The DeepSeek-Coder and DeepSeek-Coder-V2 models are designed for coding tasks such as writing and debugging code.
- **For General Users**: The DeepSeek-V3 model is a versatile option capable of handling a wide range of queries, from casual conversations to complex content generation.
- **For Researchers & Advanced Users**: The DeepSeek- R1 model specializes in advanced reasoning and logical analysis, making it ideal for problem-solving and research applications.
- **For Vision Tasks**: The DeepSeek-Janus family and DeepSeek-VL models are tailored for multimodal tasks, including image generation and processing.
-

Running DeepSeek R1 on Ollama

Step 1: Install Ollama

To run DeepSeek models on your local machine, you need to install Ollama:

- Download Ollama: Click here to download*
- For Linux users: Run the following command in your terminal:bashCopyEdit

```
curl -fsSL https://ollama.com/install.sh | sh
```

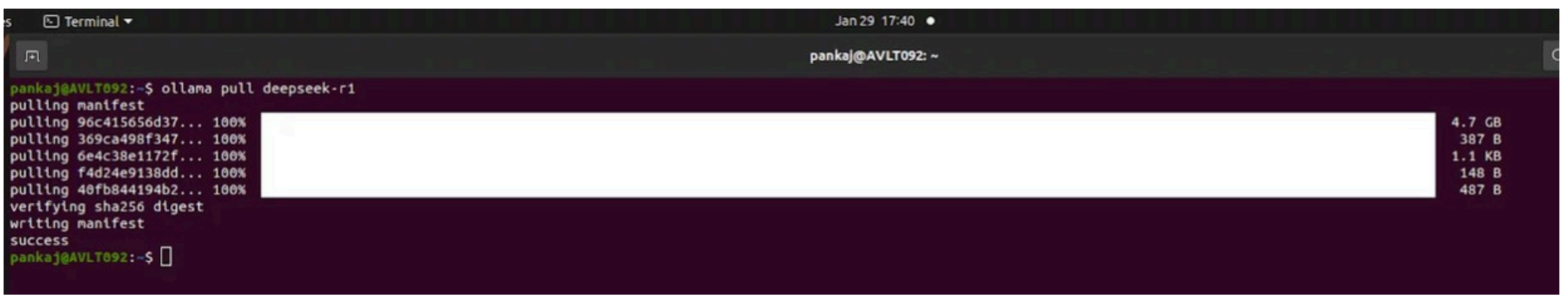
Step 2: Pull the DeepSeek R1 Model (distilled variant) Once Ollama is installed, open your Command Line Interface (CLI) and pull the model:

```
ollama pull deepseek-r1
```

*If you have downloaded the PDF, you can able to click on the link

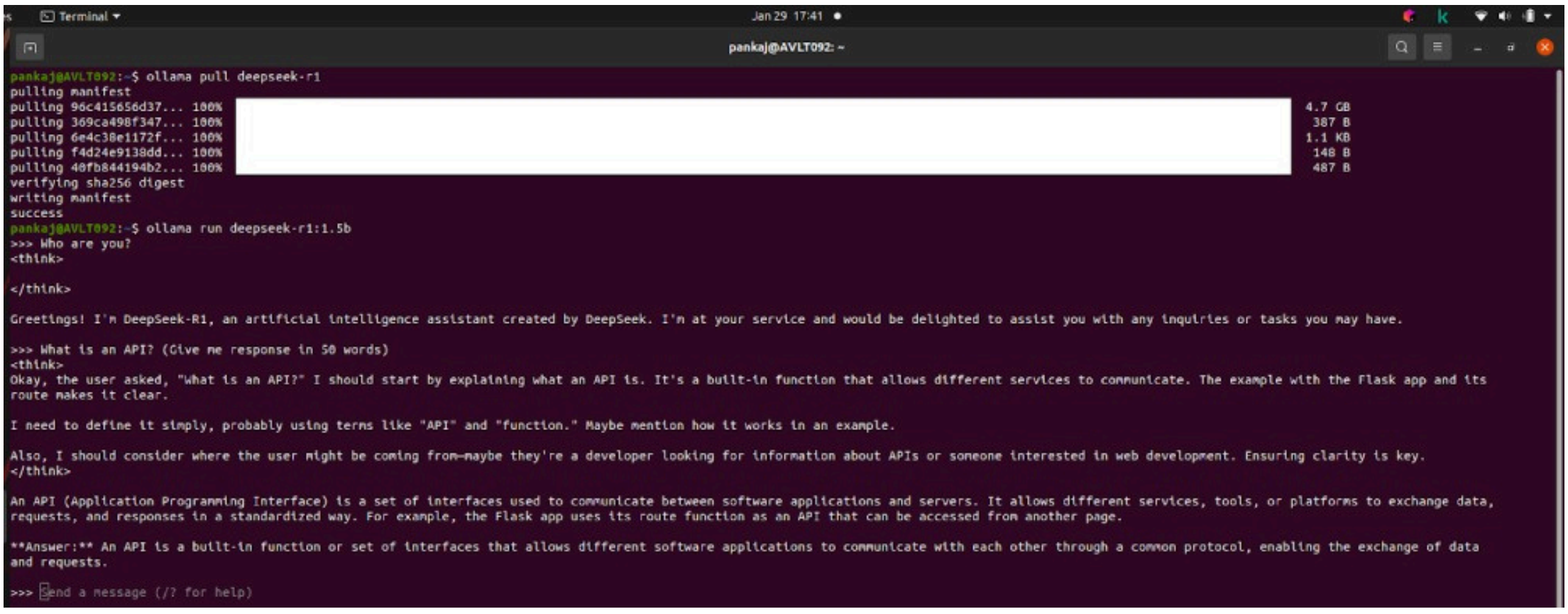
You can explore other DeepSeek models available on Ollama here: Ollama Model Search.

This step may take some time, so wait for the download to complete.



Step 3: Run the Model Locally
Once the model is downloaded, you can run it using the command:

```
ollama run deepseek-r1:1.5b
```



The model is now available to use on the local machine and is answering my questions without any hiccups.

Running DeepSeek-Janus-Pro-1B on Google Colab

In this section, we'll try out DeepSeek-Janus-Pro-1B using Google Colab. Before starting, make sure to set the runtime to T4 GPU for optimal performance.

Step 1: Clone the DeepSeek-Janus Repository

Run the following command in a Colab notebook:

```
!git clone https://github.com/deepseek-ai/Janus.git
```

Step 2: Install Dependencies
Navigate to the cloned directory and install the required packages:

```
%cd Janus
!pip install -e .
!pip install flash-attn
```

Step 3: Load the Model and Move It to GPU
Now, we'll import necessary libraries and load the model onto CUDA (GPU):

Analytics Vidhya

```
import torch
from transformers import AutoModelForCausalLM
from janus.models import MultiModalityCausalLM,
VLChatProcessor
from janus.utils.io import load_pil_images

# Define model path
model_path = "deepseek-ai/Janus-Pro-1B"

# Load processor and tokenizer
vl_chat_processor =
VLChatProcessor.from_pretrained(model_path)
tokenizer = vl_chat_processor.tokenizer

# Load model with remote code enabled
vl_gpt =
AutoModelForCausalLM.from_pretrained(model_path,
trust_remote_code=True)

# Move model to GPU
vl_gpt = vl_gpt.to(torch.bfloat16).cuda().eval()
```

Step 4: Pass an Image for Processing

Now, let’s pass an image to the model and generate a response.

📷 Input Image

Latest Articles

How to Access DeepSeek Janus Pro 7B?

DeepSeek Janus Pro 7B, a state-of-the-art multimodal AI that outperforms competitors in reasoning, text-to-image, and instruction-following.

Analytics Vidhya

Initializing the Prompt and System Role

```
image_path = "/content/snapshot.png"
question = "What's in the image?"

conversation = [
    {"role": "<|User|>", "content": f"<image_placeholder>\n{question}",
"images": [image_path]},
    {"role": "<|Assistant|>", "content": ""}
]
```

Processing the Input

```
# Load image
pil_images = load_pil_images(conversation)

# Prepare inputs for the model
prepare_inputs = vl_chat_processor(conversations=conversation, images=pil_images,
force_batchify=True).to(vl_gpt.device)
inputs_embeds = vl_gpt.prepare_inputs_embeds(**prepare_inputs)

# Generate response
outputs = vl_gpt.language_model.generate(
    inputs_embeds=inputs_embeds,
    attention_mask=prepare_inputs.attention_mask,
    pad_token_id=tokenizer.eos_token_id,
    bos_token_id=tokenizer.bos_token_id,
    eos_token_id=tokenizer.eos_token_id,
    max_new_tokens=512,
    do_sample=False,
    use_cache=True,
)

# Decode and print response
answer = tokenizer.decode(outputs[0].cpu().tolist(), skip_special_tokens=True)
print(f"{prepare_inputs['sft_format'][0]}", answer)
```

Analytics Vidhya

Initializing the Prompt and System Role

```
image_path = "/content/snapshot.png"
question = "What's in the image?"

conversation = [
    {"role": "<|User|>", "content": f"<image_placeholder>\n{question}",
"images": [image_path]},
    {"role": "<|Assistant|>", "content": ""}
]
```

Processing the Input

```
# Load image
pil_images = load_pil_images(conversation)

# Prepare inputs for the model
prepare_inputs = vl_chat_processor(conversations=conversation, images=pil_images,
force_batchify=True).to(vl_gpt.device)
inputs_embeds = vl_gpt.prepare_inputs_embeds(**prepare_inputs)

# Generate response
outputs = vl_gpt.language_model.generate(
    inputs_embeds=inputs_embeds,
    attention_mask=prepare_inputs.attention_mask,
    pad_token_id=tokenizer.eos_token_id,
    bos_token_id=tokenizer.bos_token_id,
    eos_token_id=tokenizer.eos_token_id,
    max_new_tokens=512,
    do_sample=False,
    use_cache=True,
)

# Decode and print response
answer = tokenizer.decode(outputs[0].cpu().tolist(), skip_special_tokens=True)
print(f"{prepare_inputs['sft_format'][0]}", answer)
```