CS 696 Applied Large Langauge Models
Spring Semester, 2025
Doc 4 Running Models
Jan 23, 2025

Accessing via Juptyer

https://jupyterlab.readthedocs.io/en/stable/index.html

conda install conda-forge::transformers

Juptyer Lab

HandsOn.ipynb | mlx.ipynb | Class Demo.ipynb

Code | Python 3 (ipykernel)

Chapter 1

[4]:
```
from transformers import AutoModelForCausalLM, AutoTokenizer

# Load model and tokenizer
model = AutoModelForCausalLM.from_pretrained(
    "microsoft/Phi-3-mini-4k-instruct",
    #device_map="auto",
    attn_implementation='eager',
    torch_dtype="auto",
    trust_remote_code=True,
)
tokenizer = AutoTokenizer.from_pretrained("microsoft/Phi-3-min
```

Loading checkpoint shards: 100% 2/2 [00:00<00:00, 4.22it/s]

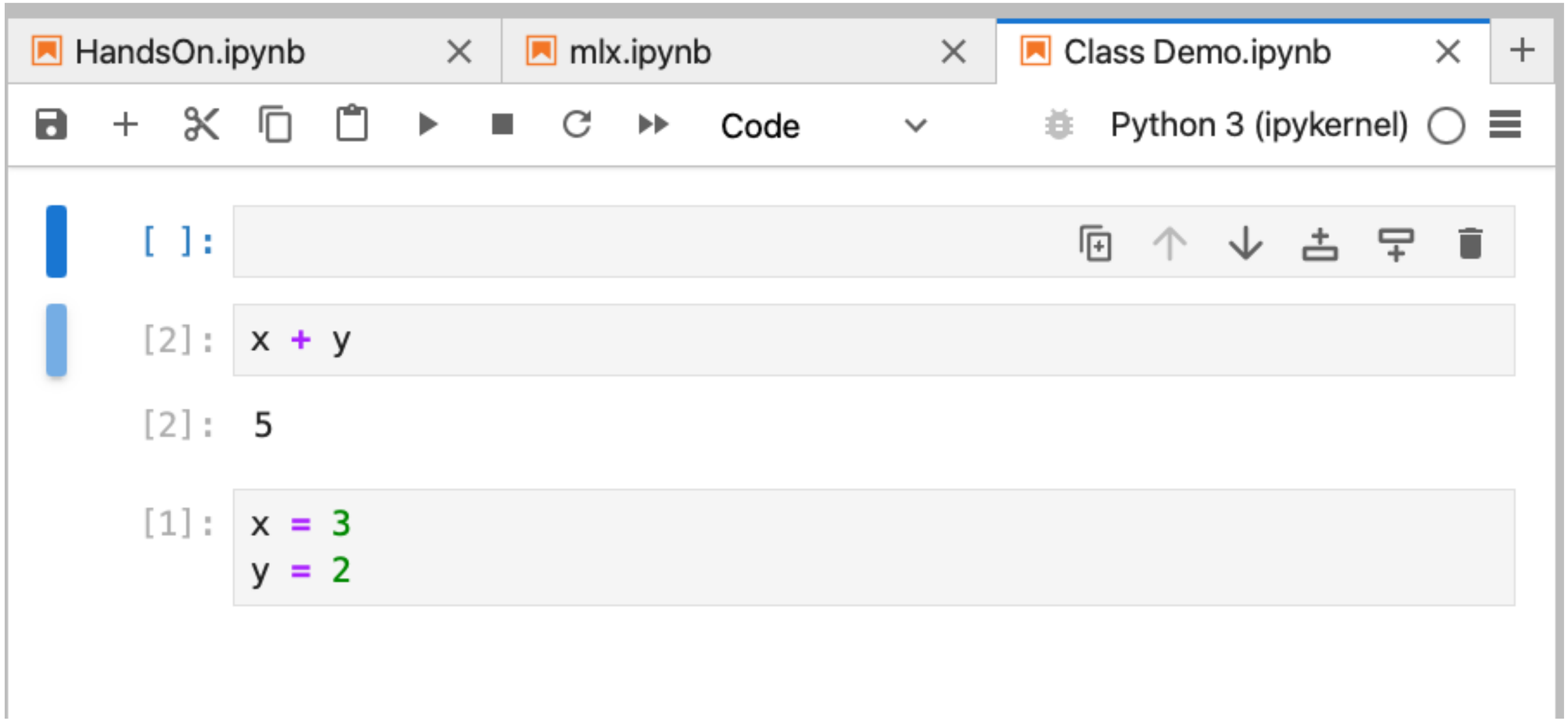
[2]:
```
from transformers import pipeline

# Create a pipeline
generator = pipeline(
    "text-generation",
    model=model,
    tokenizer=tokenizer,
    return_full_text=True,
    max_new_tokens=500,
    do_sample=False
)
```

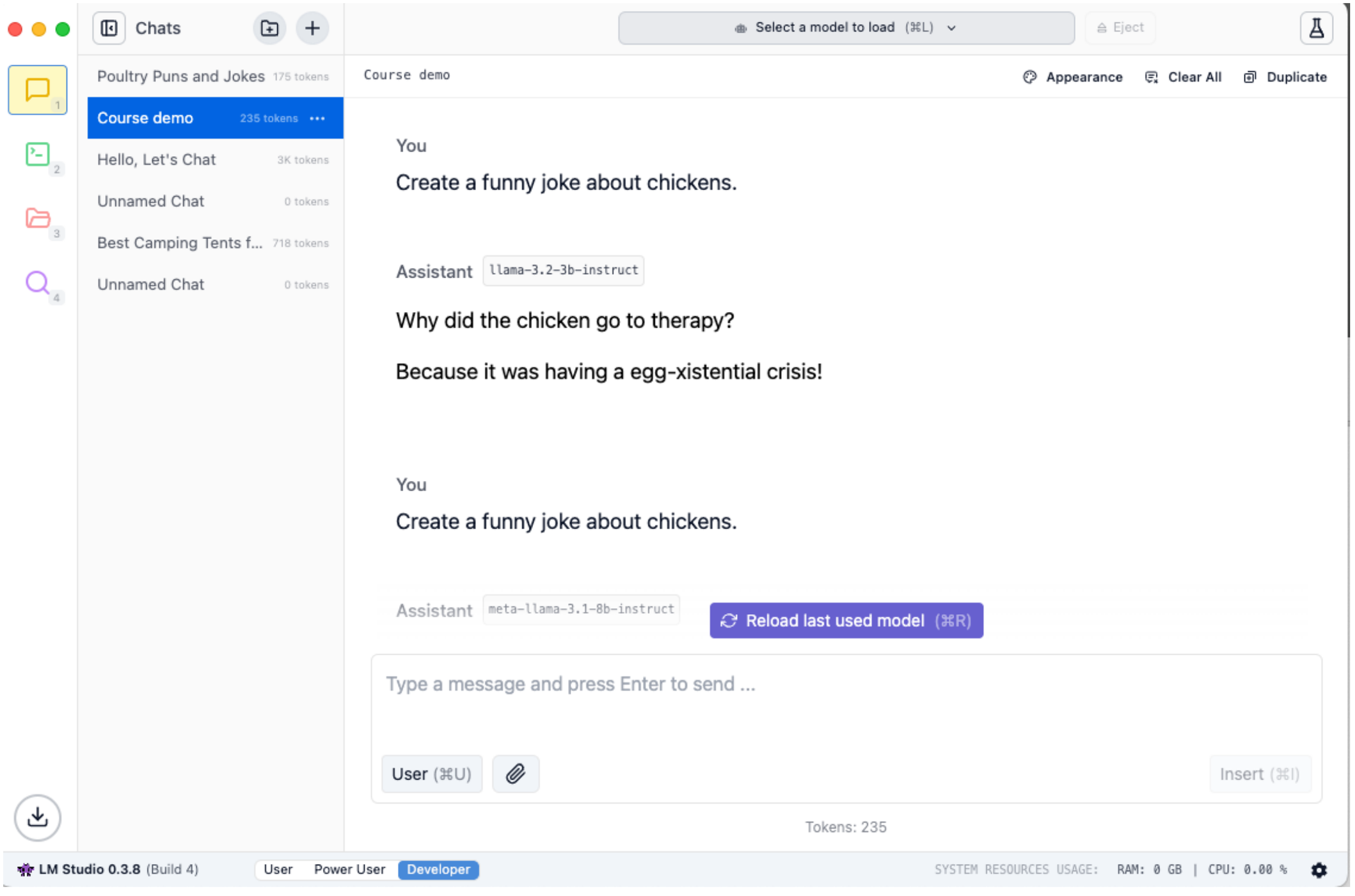
```
Device set to use mps:0
```

[3]:
```
# The prompt (user input / query)
messages = [
```

Juptyer Lab


HandsOn.ipynb
mlx.ipynb
Class Demo.ipynb
Code
Python 3 (ipykernel)
[ ]:
[2]:
x + y
[2]:
5
[1]:
x = 3
y = 2

LM Studio https://lmstudio.ai https://lmstudio.ai/docs



mlx_lm https://pypi.org/project/mlx-lm/

For macOS

Python

```
from mlx_lm import load, generate

model, tokenizer = load("mlx-community/Mistral-7B-Instruct-v0.3-4bit")
prompt = "Write a story about Einstein"

messages = [{"role": "user", "content": prompt}]
prompt = tokenizer.apply_chat_template(
    messages, add_generation_prompt=True
)

text = generate(model, tokenizer, prompt=prompt, verbose=True)
```

mlx_lm https://pypi.org/project/mlx-lm/

For macOS

Python - Streaming

```
from mlx_lm import load, generate

model, tokenizer = load("mlx-community/Mistral-7B-Instruct-v0.3-4bit")
prompt = "Write a story about Einstein"

messages = [{"role": "user", "content": prompt}]
prompt = tokenizer.apply_chat_template(
    messages, add_generation_prompt=True
)

text = generate(model, tokenizer, prompt=prompt, verbose=True)
```

mlx_lm

https://pypi.org/project/mlx-lm/

Command line

```
mlx_lm.generate
    --model /Users/rwhitney/.cache/lm-studio/models/mlx-communityLlama-3.2-3B-Instruct-4bit
    --prompt "hello"

        Prompt: <|begin_of_text|><|start_header_id|>system<|end_header_id|>

        Cutting Knowledge Date: December 2023
        Today Date: 22 Jan 2025

        <|eot_id|><|start_header_id|>user<|end_header_id|>

        hello<|eot_id|><|start_header_id|>assistant<|end_header_id|>


        Hello! How can I assist you today?
        ==========
        Prompt: 37 tokens, 41.083 tokens-per-sec
        Generation: 10 tokens, 44.464 tokens-per-sec
        Peak memory: 1.856 GB
```

mlx_lm

https://pypi.org/project/mlx-lm/

Command line

```
mlx_lm.chat
    --model mlx-communityLlama-3.2-3B-Instruct-4bit
```

```
Fetching 6 files: 100%|                                                  | 6/6 [00:00<00:00, 6267.95it/s]
[INFO] Starting chat session with mlx-community/Llama-3.2-3B-Instruct-4bit. To exit, enter 'q'.
>> Write a story about Einstein
It was a crisp winter morning in Princeton, New Jersey, and Albert Einstein was sitting in his
favorite armchair, sipping a cup of coffee and staring out the window at the snow-covered trees.
He had spent the previous night lost in thought, pondering the mysteries of the universe.
```

mlx_lm

https://pypi.org/project/mlx-lm/

```
def loss_fn(w, x, y):
   return mx.mean(mx.square(w * x - y))

w = mx.array(1.0)
x = mx.array([0.5, -0.5])
y = mx.array([1.5, -1.5])

# Computes the gradient of loss_fn with respect to w:
grad_fn = mx.grad(loss_fn)
dloss_dw = grad_fn(w, x, y)
# Prints array(-1, dtype=float32)
print(dloss_dw)

# To get the gradient with respect to x we can do:
grad_fn = mx.grad(loss_fn, argnums=1)
dloss_dx = grad_fn(w, x, y)
# Prints array([-1, 1], dtype=float32)
print(dloss_dx)
```