Print to PDF

Self-Hosted Models via Runhouse

This example goes over how to use LangChain and Runhouse to interact with models hosted on your own GPU, or on-demand GPUs on AWS, GCP, AWS, or Lambda.

For more information, see Runhouse or the Runhouse docs.

```
from langchain.llms import SelfHostedPipeline, SelfHostedHuggingFaceLLM
from langchain import PromptTemplate, LLMChain
import runhouse as rh
```

```
# For an on-demand A100 with GCP, Azure, or Lambda
gpu = rh.cluster(name="rh-a10x", instance_type="A100:1", use_spot=False)

# For an on-demand A10G with AWS (no single A100s on AWS)
# gpu = rh.cluster(name='rh-a10x', instance_type='g5.2xlarge', provider='aws')

# For an existing cluster
# gpu = rh.cluster(ips=['<ip of the cluster>'],
#                  ssh_creds={'ssh_user': '...',
'ssh_private_key':'<path_to_key>'},
#                  name='rh-a10x')
```

```
template = """Question: {question}

Answer: Let's think step by step."""

prompt = PromptTemplate(template=template, input_variables=["question"])
```

```
llm = SelfHostedHuggingFaceLLM(model_id="gpt2", hardware=gpu, model_reqs=
["pip:./", "transformers", "torch"])
```

```
llm_chain = LLMChain(prompt=prompt, llm=llm)
```

Skip to main content

```
question = "What NFL team won the Super Bowl in the year Justin Beiber was born?"

llm_chain.run(question)
```

```
INFO | 2023-02-17 05:42:23,537 | Running _generate_text via gRPC
INFO | 2023-02-17 05:42:24,016 | Time to send message: 0.48 seconds
```

```
"\n\nLet's say we're talking sports teams who won the Super Bowl in the year
Justin Beiber"
```

You can also load more custom models through the SelfHostedHuggingFaceLLM interface:

```
llm = SelfHostedHuggingFaceLLM(
    model_id="google/flan-t5-small",
    task="text2text-generation",
    hardware=gpu,
)
```

```
llm("What is the capital of Germany?")
```

```
INFO | 2023-02-17 05:54:21,681 | Running _generate_text via gRPC
INFO | 2023-02-17 05:54:21,937 | Time to send message: 0.25 seconds
```

```
'berlin'
```

Using a custom load function, we can load a custom pipeline directly on the remote hardware:

```
def load_pipeline():
    from transformers import AutoModelForCausalLM, AutoTokenizer, pipeline  # Need
to be inside the fn in notebooks
    model_id = "gpt2"
    tokenizer = AutoTokenizer.from_pretrained(model_id)
    model = AutoModelForCausalLM.from_pretrained(model_id)
    pipe = pipeline(
        "text-generation", model=model, tokenizer=tokenizer, max_new_tokens=10
    )
    return pipe
```

Skip to main content

```
def inference_fn(pipeline, prompt, stop = None):
    return pipeline(prompt)[0]["generated_text"][len(prompt):]
```

```
llm = SelfHostedHuggingFaceLLM(model_load_fn=load_pipeline, hardware=gpu,
inference_fn=inference_fn)
```

```
llm("Who is the current US president?")
```

```
INFO | 2023-02-17 05:42:59,219 | Running _generate_text via gRPC
INFO | 2023-02-17 05:42:59,522 | Time to send message: 0.3 seconds
```

```
'john w. bush'
```

You can send your pipeline directly over the wire to your model, but this will only work for small models (<2 Gb), and will be pretty slow:

```
pipeline = load_pipeline()
llm = SelfHostedPipeline.from_pipeline(
    pipeline=pipeline, hardware=gpu, model_reqs=model_reqs
)
```

Instead, we can also send it to the hardware's filesystem, which will be much faster.

```
rh.blob(pickle.dumps(pipeline), path="models/pipeline.pkl").save().to(gpu,
path="models")

llm = SelfHostedPipeline.from_pipeline(pipeline="models/pipeline.pkl",
hardware=gpu)
```