Use model after training

Once you have trained a model using either the SFTTrainer, PPOTrainer, or DPOTrainer, you will have a fine-tuned model that can be used for text generation. In this section, we'll walk through the process of loading the fine-tuned model and generating text. If you need to run an inference server with the trained model, you can explore libraries such as `text-generation-inference`.

Load and Generate

If you have fine-tuned a model fully, meaning without the use of PEFT you can simply load it like any other language model in transformers. E.g. the value head that was trained during the PPO training is no longer needed and if you load the model with the original transformer class it will be ignored:

```
from transformers import AutoTokenizer, AutoModelForCausalLM

model_name_or_path = "kashif/stack-llama-2" #path/to/your/model/or/name/on/hub
device = "cpu" # or "cuda" if you have a GPU
```

```
model = AutoModelForCausalLM.from_pretrained(model_name_or_path).to(device)
tokenizer = AutoTokenizer.from_pretrained(model_name_or_path)

inputs = tokenizer.encode("This movie was really", return_tensors="pt").to(device)
outputs = model.generate(inputs)
print(tokenizer.decode(outputs[0]))
```

Alternatively you can also use the pipeline:

```
from transformers import pipeline

model_name_or_path = "kashif/stack-llama-2" #path/to/your/model/or/name/on/hub
pipe = pipeline("text-generation", model=model_name_or_path)
print(pipe("This movie was really")[0]["generated_text"])
```

Use Adapters PEFT

```
from peft import PeftConfig, PeftModel
from transformers import AutoModelForCausalLM, AutoTokenizer

base_model_name = "kashif/stack-llama-2" #path/to/your/model/or/name/on/hub"
adapter_model_name = "path/to/my/adapter"

model = AutoModelForCausalLM.from_pretrained(base_model_name)
model = PeftModel.from_pretrained(model, adapter_model_name)
```

```
tokenizer = AutoTokenizer.from_pretrained(base_model_name)
```

You can also merge the adapters into the base model so you can use the model like a normal transformers model, however the checkpoint will be significantly bigger:

```
model = AutoModelForCausalLM.from_pretrained(base_model_name)
model = PeftModel.from_pretrained(model, adapter_model_name)

model = model.merge_and_unload()
model.save_pretrained("merged_adapters")
```

Once you have the model loaded and either merged the adapters or keep them separately on top you can run generation as with a normal model outlined above.

<> Update on GitHub

← PPO Training FAQ

Customize the Training →