Go-Gourmet!

Fine-Tuning for uniform structure

Data-Preparation

Example:

Instruction: "Gourmet Burger Kitchen, London"

Response: "Name: Gourmet Burger Kitchen,\nCuisine: Gourmet Burgers,\nOpening Times: Mon-Sun: 11am-11pm,\nLocation: 44 Northcote Rd, London, SW11 1NZ, United Kingdom (51.461280, -0.166490),\nRating: 4.2 (Source: TripAdvisor),\nAverage Price Per Person: Moderate,\nThree Best Dishes:\n 1. Classic Beef Burger: A perfectly cooked beef patty with fresh lettuce, tomato, and GBK sauce.\n 2. Chicken Satay Burger: Grilled chicken topped with homemade peanut sauce, adding a unique twist to the classic burger.\n 3. Falafel Burger: A delicious option for vegetarians, served with homemade tzatziki and fresh salad.,\nPre-Booking Needed: Not necessary, but recommended for larger groups,\nDress Code: Casual,\nWebsite: https://www.gbk.co.uk/"

Dataset of Instructions and Responses

1200 Instructions and Responses

1000 Instructions and Responses — Training

200 Instructions and Responses — Validation

This time I have auto-split the dataset just before training.

Choose Base Model

MISTRAL-7B (32k context)

Convert the Dataset into MISTRAL format.

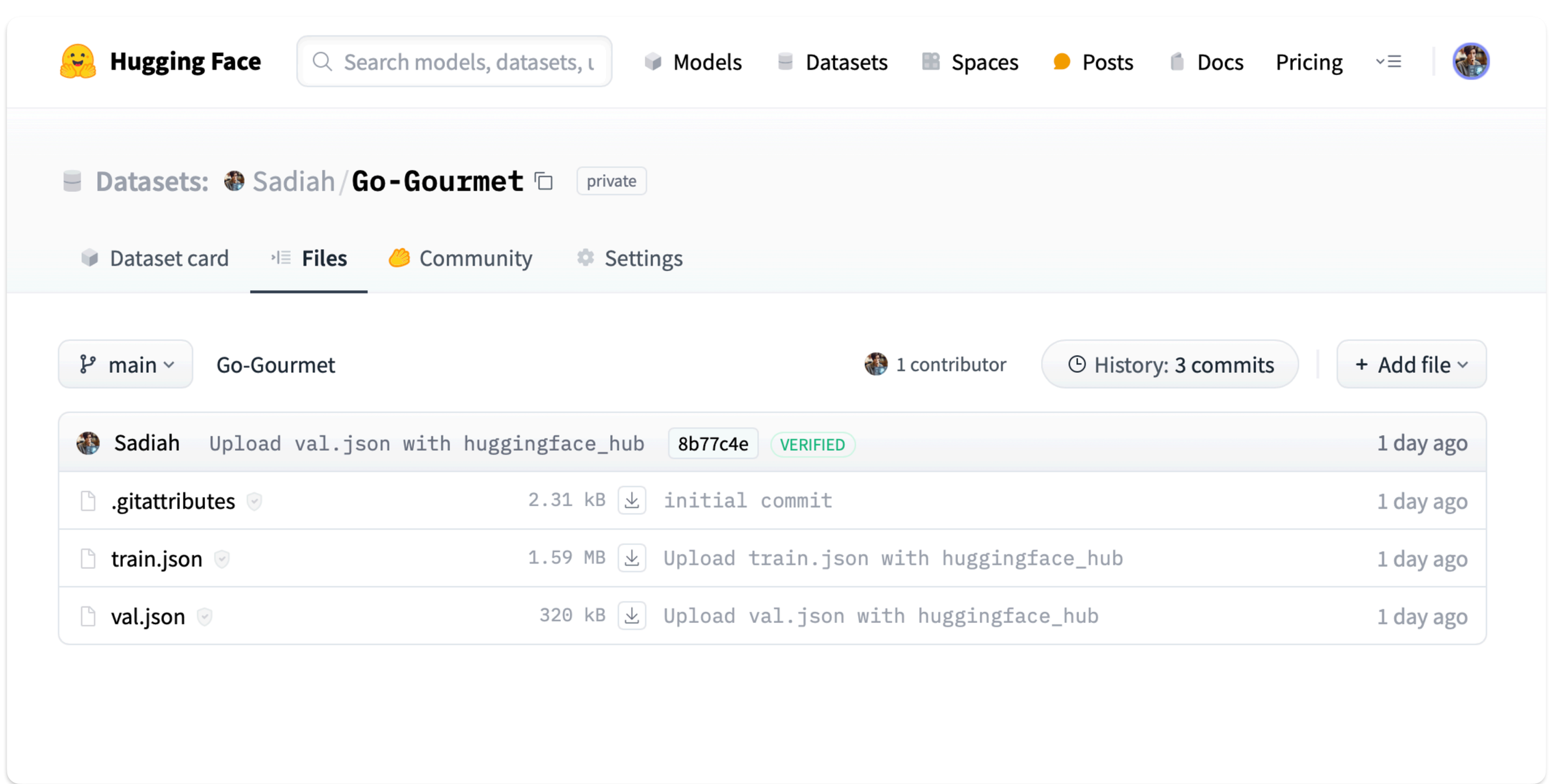
You can choose to not do this but since I will be reusing the dataset multiple times, I just want to have it handy.

Example:

Mistral Format: "<s>[INST] Gourmet Burger Kitchen, London [/INST] Name: Gourmet Burger Kitchen,\nCuisine: Gourmet Burgers,\nOpening Times: Mon-Sun: 11am-11pm,\nLocation: 44 Northcote Rd, London, SW11 1NZ, United Kingdom (51.461280, -0.166490),\nRating: 4.2 (Source: TripAdvisor),\nAverage Price Per Person: Moderate,\nThree Best Dishes:\n 1. Classic Beef Burger: A perfectly cooked beef patty with fresh lettuce, tomato, and GBK sauce.\n 2. Chicken Satay Burger: Grilled chicken topped with homemade peanut sauce, adding a unique twist to the classic burger.\n 3. Falafel Burger: A delicious option for vegetarians, served with homemade tzatziki and fresh salad.,\nPre-Booking Needed: Not necessary, but recommended for larger groups,\nDress Code: Casual,\nWebsite: https://www.gbk.co.uk/ </s>"

Push the Dataset in MISTRAL format to Hugging Face

This is how your Dataset should look like on Hugging Face.



Fine-Tuning

Choose your Platform for Fine-Tuning.

A100-80G
$3.18/hr | 12 CPU | 90 GB RAM | 80 GB GPU

Install the Dependencies

Pass on the credentials

This is your Hugging Face Token

Load the Base-Transformer model and it's Tokeniser from Hugging Face with Quantization Configuration

mistralai/Mistral-7B-Instruct-v0.2

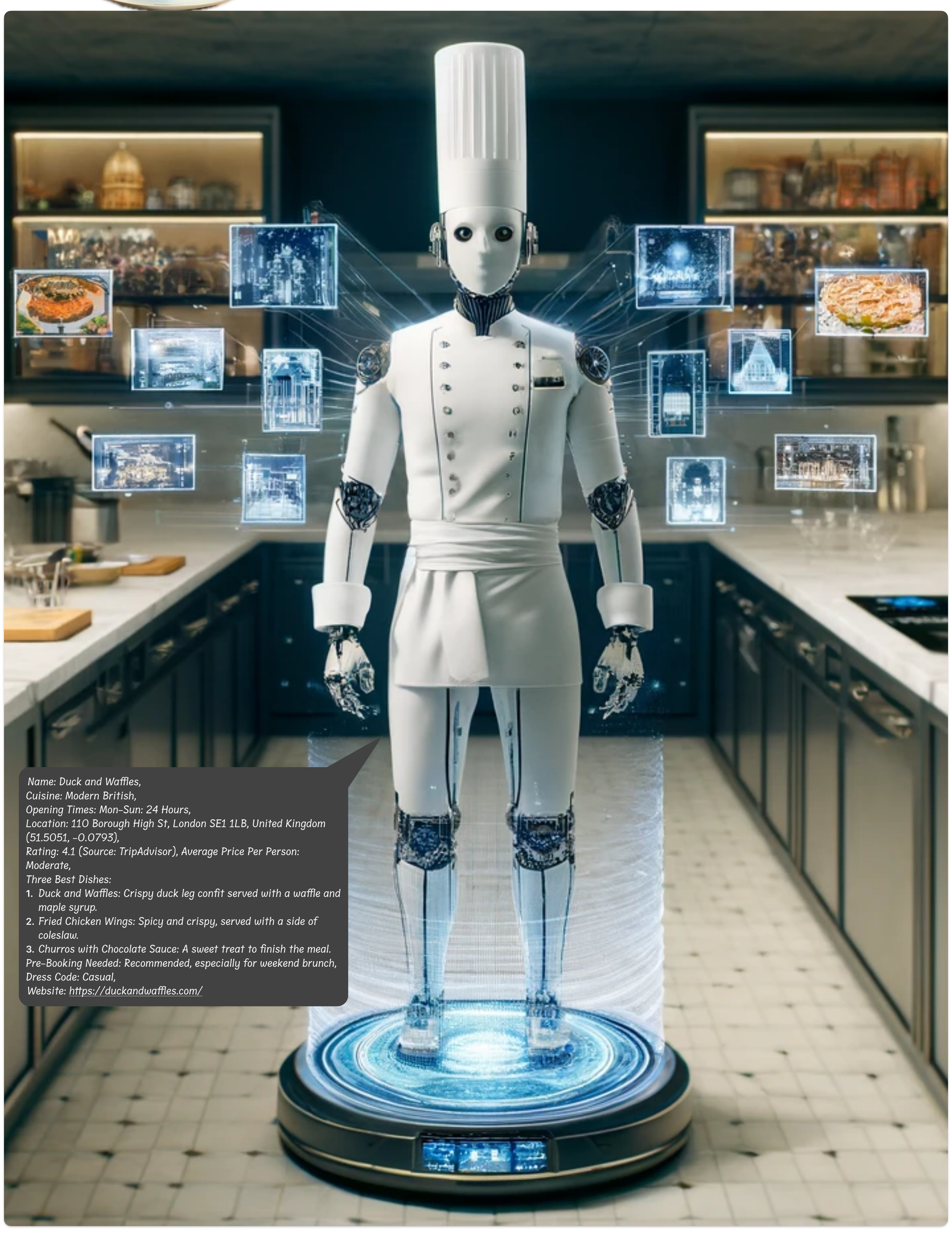
load_in_4bit,....
bb_4bit_quant_type: ...
bnb_4bit_compute_dtype, ...
bnb_4bit_use_double_quant,...
......add more

```
from transformers import AutoModelForCausalLM, AutoTokenizer

# Load the base model
model = AutoModelForCausalLM.from_pretrained(BASE_MODEL, quantization_config=bnb_config, trust_remote_code=True, device_map
# device_map= {"": 0}, # Load the entire model on the GPU 0
model.config.use_cache = False   # Disable caching of model outputs
model.config.pretraining_tp = 1  # Tensor parallelism settings

# Load tokenizer
tokenizer = AutoTokenizer.from_pretrained(BASE_MODEL, trust_remote_code=True)
tokenizer.pad_token = tokenizer.eos_token # Padding token = End of sequence token
tokenizer.padding_side = "right" # Sequnces are left indented
```

```
Token has not been saved to git credential helper. Pass `add_to_git_credential=True` if you want to set the git credential
as well.
Token is valid (permission: write).
Your token has been saved to /root/.cache/huggingface/token
Login successful
config.json: 100% 596/596 [00:00<00:00, 41.0kB/s]
model.safetensors.index.json: 100% 25.1k/25.1k [00:00<00:00, 1.97MB/s]
Downloading shards: 67% 2/3 [00:41<00:20, 20.07s/it]
model-00001-of-00003.safetensors: 100% 4.94G/4.94G [00:24<00:00, 325MB/s]
model-00002-of-00003.safetensors: 100% 5.00G/5.00G [00:16<00:00, 305MB/s]
model-00003-of-00003.safetensors: 78% 3.53G/4.54G [00:53<00:31, 32.4MB/s]
```



Load PEFT configuration for training

PEFT used: QLoRA

We will use Parameter Efficient Fine-Tuning which reduces the number of parameters we need to train.

Load the Go-Gourmet Dataset from Hugging Face

This is the Dataset that we just pushed above to the Hugging Face.

Add Training Configuration

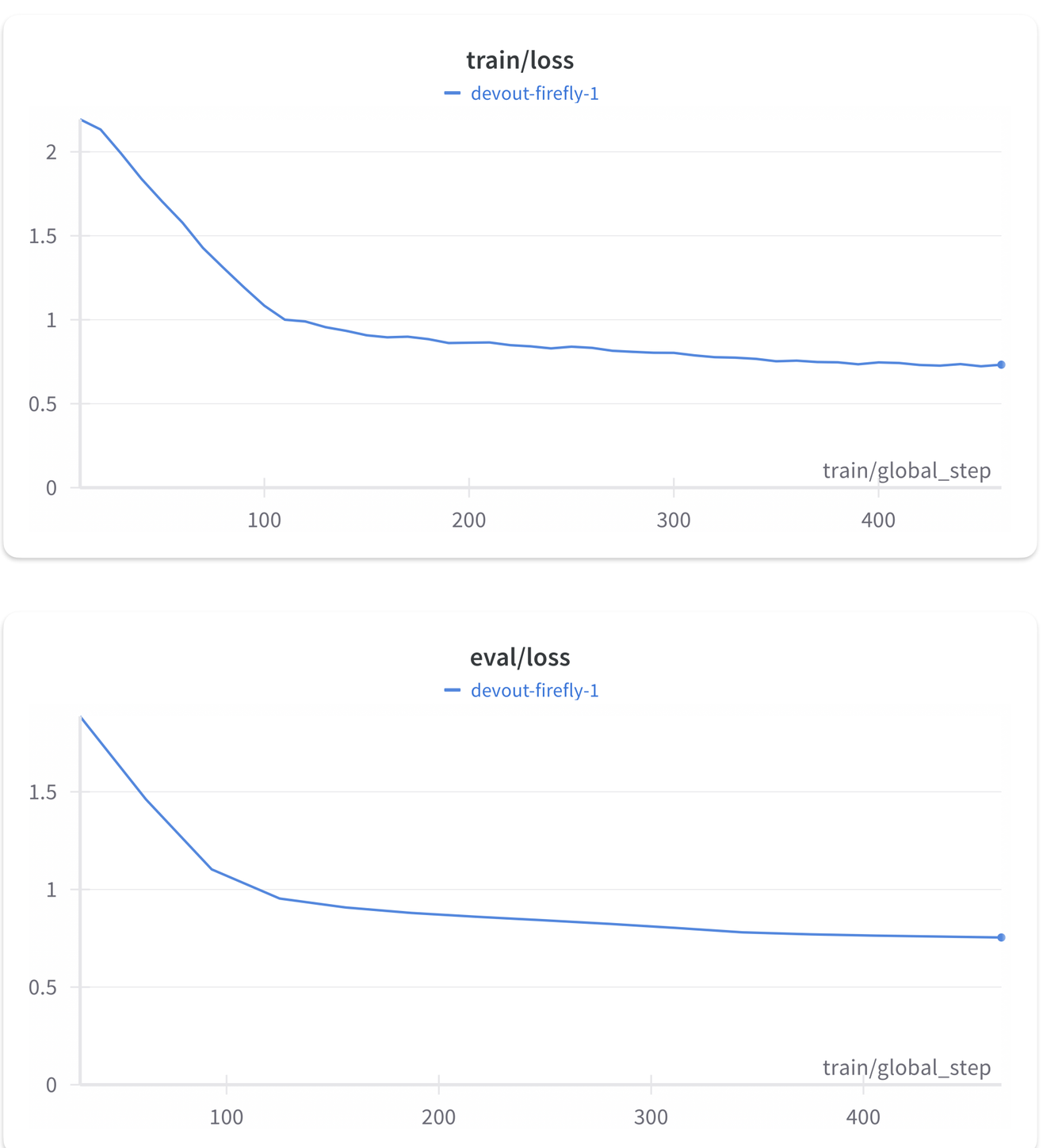
Batch Size: ...
Learning Rate: ...
Optimiser: ...
Epochs: ...
......add more

Start training loop

Time taken- 2 hrs (approx)

[465/465 2:03:36, Epoch 14/15]

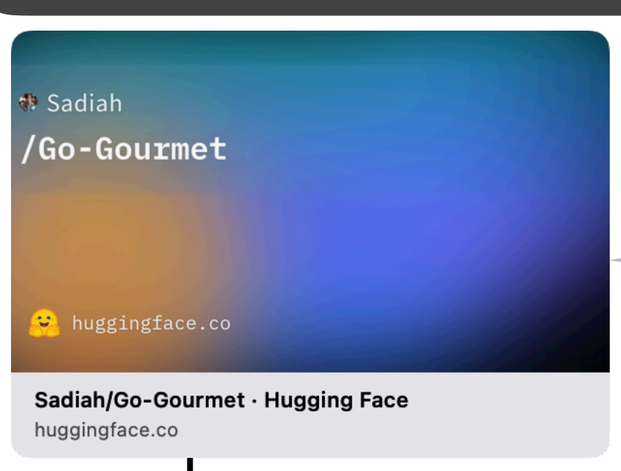
Epoch	Training Loss	Validation Loss
0	1.990300	1.886203
1	1.578500	1.461587
2	1.193700	1.102770
4	0.907400	0.907790
5	0.884800	0.879671
6	0.864800	0.859949
8	0.809000	0.823289
9	0.787900	0.802690
10	0.767200	0.780369
12	0.746000	0.763486
13	0.726700	0.758479
14	0.732400	0.754083



Merge Adaptor with Base Model

Adapter model weights in techniques like LoRA (Low-Rank Adaptation) are generally incomplete compared to the full weights of a large language model (Base LLM).
So we must load both the base model and adapter but doing so separately requires loading both components during inference, which can add latency. Merging simplifies the model by combining everything into a single entity.

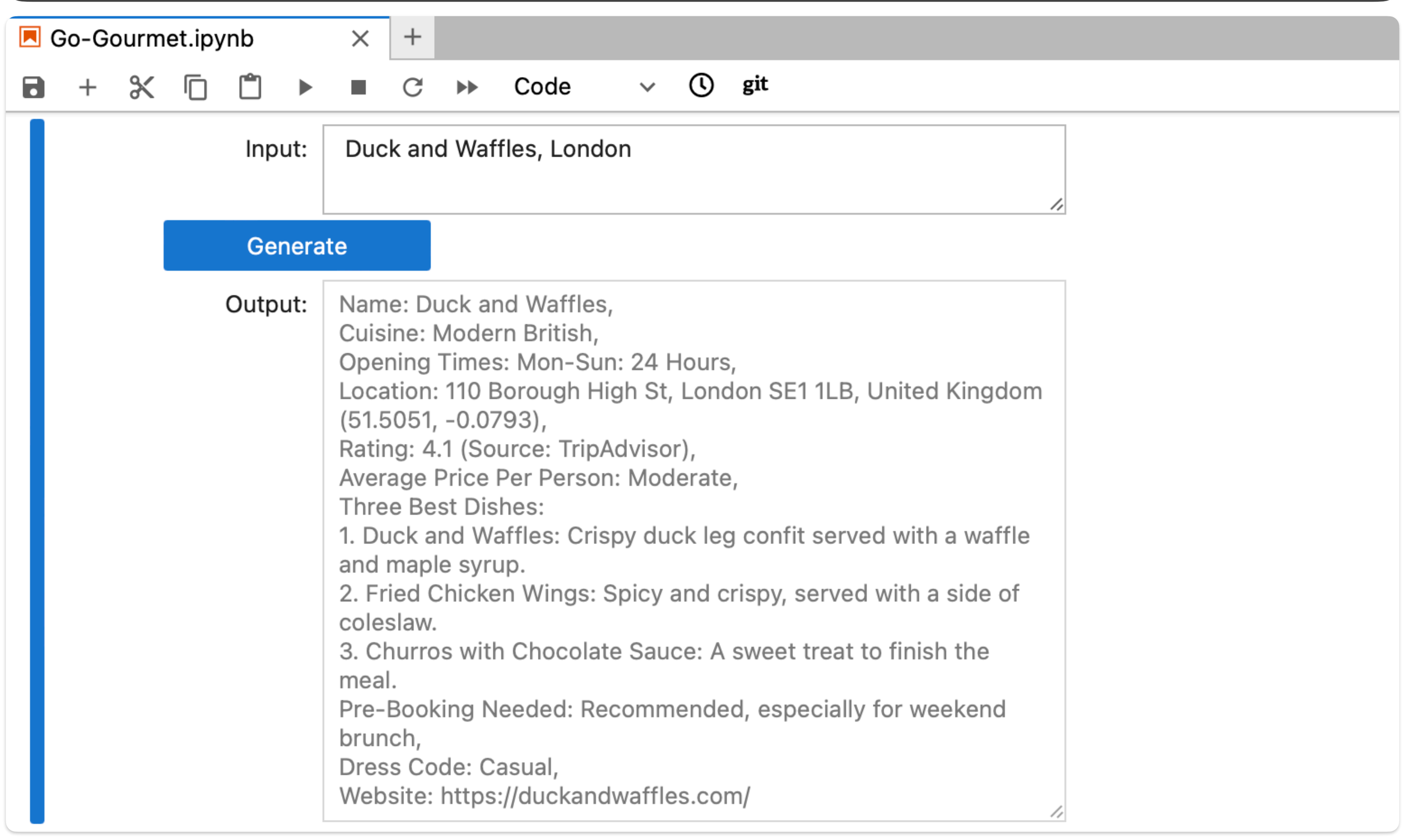
Save & Push Merged Model & Tokeniser to Hub



Here is a link to my trained Model on Hugging face.

Load the Fine-Tuned Model for Inference

Inference



Copyright © 2024 by @sadiahzahoor.
All rights reserved.