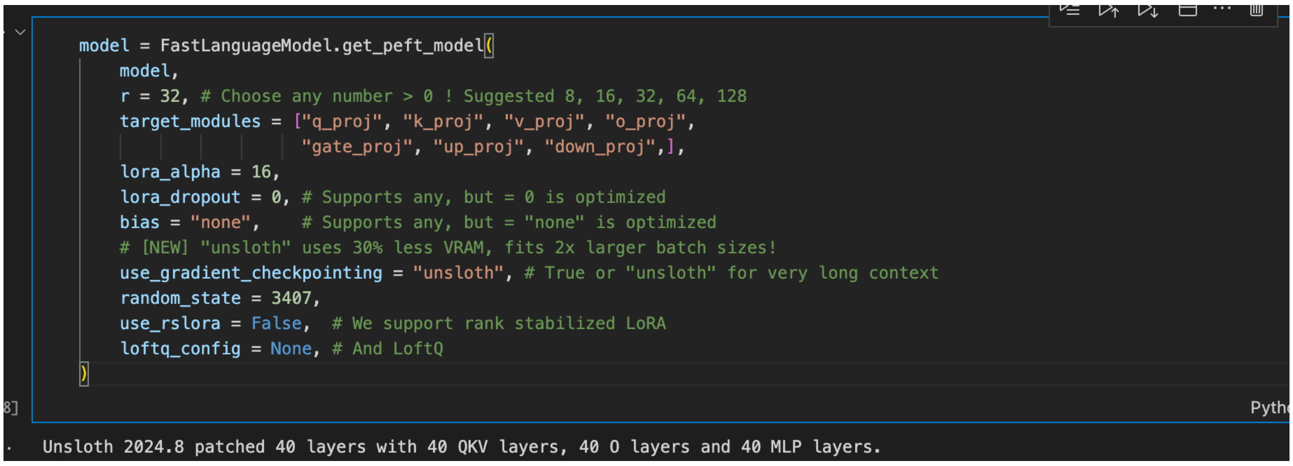
Approach For Building Model

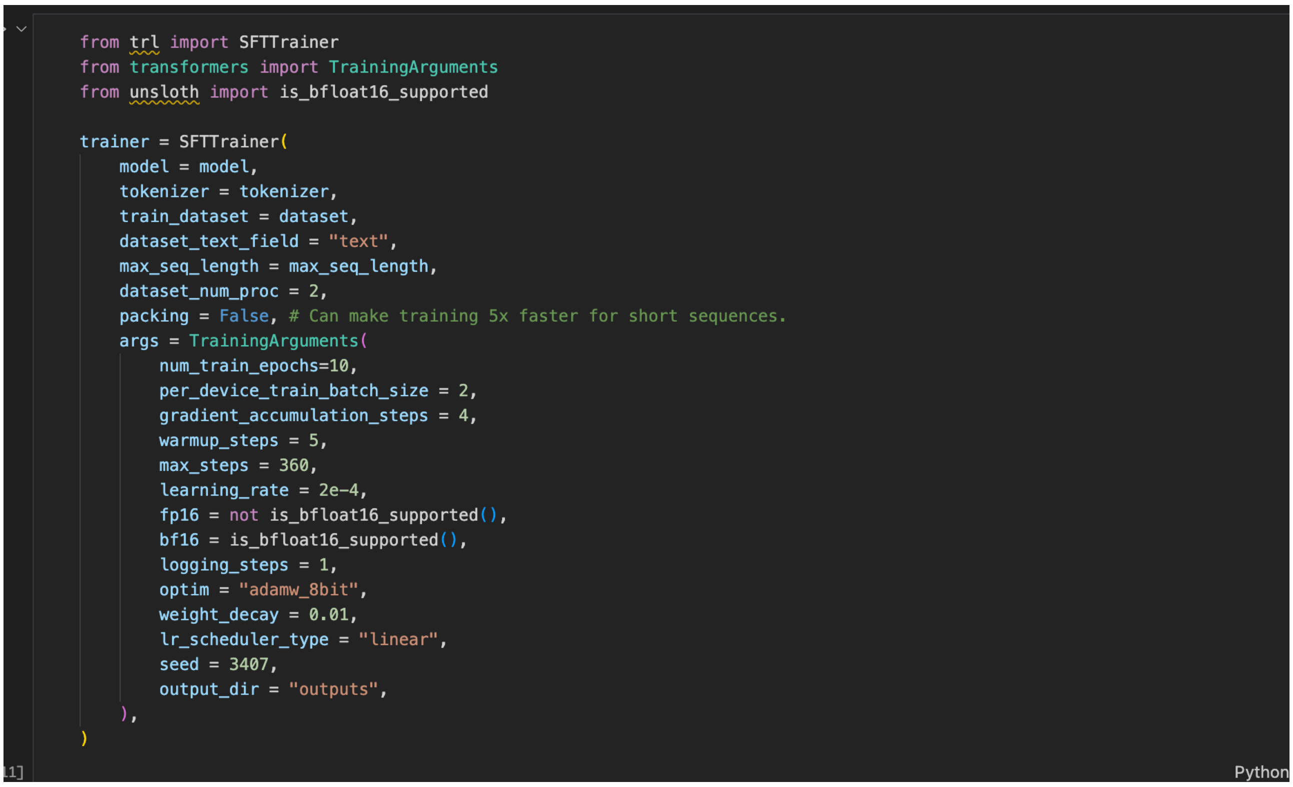
- As the Train dataset size is around 250k and for Test is around 130k , and as the images given in the sample need to be downloaded , So first we downloaded the image samples and store that in local after creating the metadata of the whole things.
- Then there could be two approaches which we decide to work that is one could be fine-tuning the Multi- Modal LLM like Open-AI Clip , PaliGemma , Llava and another approach will be to extract the Captions of the Images and Finetune any LLM like Gemma 2 , Mistral , LLama others.
- But we got less resources to fine-tune any multi-modal LLM , It takes almost 1 hours to tune with only 1000 samples so we finalise over approach to extract the captions and fine-tune it with Mistral-Nemo-12 Billion Parameters.
- We used Paddle-OCR to extract all the image captions of train and test set then make that saved in side the metadata_df with extracted - text
- Then we did LORA Fine-tuning of the Mistral-Nemo model using Unsloth and downloaded the model which is then tuned with thses model parameters :



```
model = FastLanguageModel.get_peft_model(
    model,
    r = 32, # Choose any number > 0 ! Suggested 8, 16, 32, 64, 128
    target_modules = ["q_proj", "k_proj", "v_proj", "o_proj",
                      "gate_proj", "up_proj", "down_proj",],
    lora_alpha = 16,
    lora_dropout = 0, # Supports any, but = 0 is optimized
    bias = "none",    # Supports any, but = "none" is optimized
    # [NEW] "unsloth" uses 30% less VRAM, fits 2x larger batch sizes!
    use_gradient_checkpointing = "unsloth", # True or "unsloth" for very long context
    random_state = 3407,
    use_rslora = False,  # We support rank stabilized LoRA
    loftq_config = None, # And LoftQ
)
```

```
Unsloth 2024.8 patched 40 layers with 40 QKV layers, 40 O layers and 40 MLP layers.
```

- And the using SFT trainer from TRL we just passed the arguments below and tuned it :



```
from trl import SFTTrainer
from transformers import TrainingArguments
from unsloth import is_bfloat16_supported

trainer = SFTTrainer(
    model = model,
    tokenizer = tokenizer,
    train_dataset = dataset,
    dataset_text_field = "text",
    max_seq_length = max_seq_length,
    dataset_num_proc = 2,
    packing = False, # Can make training 5x faster for short sequences.
    args = TrainingArguments(
        num_train_epochs=10,
        per_device_train_batch_size = 2,
        gradient_accumulation_steps = 4,
        warmup_steps = 5,
        max_steps = 360,
        learning_rate = 2e-4,
        fp16 = not is_bfloat16_supported(),
        bf16 = is_bfloat16_supported(),
        logging_steps = 1,
        optim = "adamw_8bit",
        weight_decay = 0.01,
        lr_scheduler_type = "linear",
        seed = 3407,
        output_dir = "outputs",
    ),
)
```

- Then save the safetensors , tokenisers and configs .
- Then For inference we load the tuned model and passed the test set with caption extracted text using Paddle -OCR and save the submission.csv file , which is used to submit the inference.