Step 1: Load the meta-llama/Llama-3.2-3B Model and Tokenizer

In [1]:
```
from transformers import AutoModelForCausalLM, AutoTokenizer

model_name = "meta-llama/Llama-3.2-3B"
tokenizer = AutoTokenizer.from_pretrained(model_name, trust_remote_code=True
tokenizer.pad_token = tokenizer.eos_token
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    load_in_4bit=True,  # if using bitsandbytes
    device_map="auto",
    trust_remote_code=True
)
```

```
The `load_in_4bit` and `load_in_8bit` arguments are deprecated and will be r
emoved in the future versions. Please, pass a `BitsAndBytesConfig` object in
`quantization_config` argument instead.
Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]
```

Step 2: Prepare the LoRA Configuration with PEFT

In [2]:
```
from peft import PeftModel, get_peft_model, LoraConfig, TaskType

peft_config = LoraConfig(
    r=8,
    lora_alpha=32,
    lora_dropout=0.1,
    bias="none",
    task_type=TaskType.CAUSAL_LM
)

# Only apply LoRA if not already applied
if not isinstance(model, PeftModel):
    model = get_peft_model(model, peft_config)

model.print_trainable_parameters()
```

```
trainable params: 2,293,760 || all params: 3,215,043,584 || trainable%: 0.07
13
```

Step 3: Load and Preprocess Dataset¶

zh-en

In [3]:
```
from datasets import load_dataset

# Load full dataset (zh-en)
dataset_zh_en = load_dataset("wmt19", "zh-en")
```

In [4]:
```
print(len(dataset_zh_en["train"]))
print(len(dataset_zh_en["validation"]))
```

```
25984574
3981
```

In [5]:
```
# Set slice sizes
TRAIN_SIZE = 70_000
VAL_SIZE = 1000

# Randomly shuffle and select subsets
small_dataset_zh_en = {
    "train": dataset_zh_en["train"].shuffle(seed=42).select(range(TRAIN_SIZE
    "validation": dataset_zh_en["validation"].shuffle(seed=42).select(range(
}
```

In [6]:
```
print(len(small_dataset_zh_en["train"]))
print(len(small_dataset_zh_en["validation"]))
```

```
70000
1000
```

ne-en

In [7]:
```
from datasets import load_dataset
from datasets.dataset_dict import DatasetDict

# 1. Load full dataset (only has a "train" split)
dataset_ne_en = load_dataset("iamTangsang/Nepali-to-English-Translation-Data
```

In [8]:
```
print(len(dataset_ne_en["train"]))
print(len(dataset_ne_en["validation"]))
```

```
702697
10866
```

In [9]:
```
# Set slice sizes
TRAIN_SIZE = 70_000
VAL_SIZE = 1000

# Randomly shuffle and select subsets
small_dataset_ne_en = {
    "train": dataset_ne_en["train"].shuffle(seed=42).select(range(TRAIN_SIZE
    "validation": dataset_ne_en["validation"].shuffle(seed=42).select(range(
}
```

In [10]:
```
print(len(small_dataset_ne_en["train"]))
print(len(small_dataset_ne_en["validation"]))
```

```
70000
1000
```

combine them

In [11]:
```
# zh-en format
zh_en_train = small_dataset_zh_en["train"].map(lambda x: {
    "input": x["translation"]["zh"].strip(),
    "output": x["translation"]["en"].strip(),
    "lang_pair": "zh-en"
})

zh_en_val = small_dataset_zh_en["validation"].map(lambda x: {
    "input": x["translation"]["zh"].strip(),
    "output": x["translation"]["en"].strip(),
    "lang_pair": "zh-en"
})

# ne-en format
ne_en_train = small_dataset_ne_en["train"].map(lambda x: {
    "input": x["source"].strip(),
    "output": x["target"].strip(),
    "lang_pair": "ne-en"
})

ne_en_val = small_dataset_ne_en["validation"].map(lambda x: {
    "input": x["source"].strip(),
    "output": x["target"].strip(),
    "lang_pair": "ne-en"
})
```

In [12]:
```
print(zh_en_train)
print(ne_en_train)
```

```
Dataset({
    features: ['translation', 'input', 'output', 'lang_pair'],
    num_rows: 70000
})
Dataset({
    features: ['source', 'target', 'input', 'output', 'lang_pair'],
    num_rows: 70000
})
```

In [13]:
```
print(zh_en_train.features)
print(ne_en_train.features)
```

```
{'translation': Translation(languages=['zh', 'en'], id=None), 'input': Value
(dtype='string', id=None), 'output': Value(dtype='string', id=None), 'lang_p
air': Value(dtype='string', id=None)}
{'source': Value(dtype='string', id=None), 'target': Value(dtype='string', i
d=None), 'input': Value(dtype='string', id=None), 'output': Value(dtype='str
ing', id=None), 'lang_pair': Value(dtype='string', id=None)}
```

In [14]:
```
from datasets import concatenate_datasets
```

```
combined_train = concatenate_datasets([zh_en_train, ne_en_train]).shuffle(se
combined_val = concatenate_datasets([zh_en_val, ne_en_val]).shuffle(seed=42)
```

In [15]:
```
print(combined_train)
print(combined_val)
```

```
Dataset({
    features: ['translation', 'input', 'output', 'lang_pair', 'source', 'tar
get'],
    num_rows: 140000
})
Dataset({
    features: ['translation', 'input', 'output', 'lang_pair', 'source', 'tar
get'],
    num_rows: 2000
})
```

In [16]:
```
zh2en_templates = [
    "User: Translate Chinese to English: {input}\nAssistant: {output}",
    "User: What is the English translation of: {input}?\nAssistant: {output}
    "User: Please convert this to English: {input}\nAssistant: {output}"
]

en2zh_templates = [
    "User: Translate English to Chinese: {input}\nAssistant: {output}",
    "User: What is the Chinese translation of: {input}?\nAssistant: {output}
    "User: Please convert this to Chinese: {input}\nAssistant: {output}"
]

ne2en_templates = [
    "User: Translate Nepali to English: {input}\nAssistant: {output}",
    "User: What is the English translation of: {input}?\nAssistant: {output}
    "User: Please convert this to English: {input}\nAssistant: {output}"
]

en2ne_templates = [
    "User: Translate English to Nepali: {input}\nAssistant: {output}",
    "User: What is the Nepali translation of: {input}?\nAssistant: {output}"
    "User: Please convert this to Nepali: {input}\nAssistant: {output}"
]
```

In [17]:
```
import random
def preprocess(example,batched=True):
    lang = example["lang_pair"]
    prompt = ""

    if lang == "zh-en":
        if random.random() < 0.5:
            prompt = random.choice(zh2en_templates).format(input=example["ir
        else:
            prompt = random.choice(en2zh_templates).format(input=example["ou

    elif lang == "ne-en":
        if random.random() < 0.5:
            prompt = random.choice(ne2en_templates).format(input=example["ir
        else:
```

```
            prompt = random.choice(en2ne_templates).format(input=example["ou

    tokenized = tokenizer(prompt, truncation=True, padding="max_length", max
    tokenized["labels"] = tokenized["input_ids"].copy()
    return tokenized
```

In [18]:
```
tokenized_dataset = {
    "train": combined_train.map(preprocess, remove_columns=combined_train.co
    "validation": combined_val.map(preprocess, remove_columns=combined_val.c
}
```

In [19]:
```
print(tokenized_dataset)
```

```
{'train': Dataset({
    features: ['input_ids', 'attention_mask', 'labels'],
    num_rows: 140000
}), 'validation': Dataset({
    features: ['input_ids', 'attention_mask', 'labels'],
    num_rows: 2000
})}
```

Step 4: Setup Training with Trainer

In [20]:
```
from transformers import TrainingArguments, Trainer

training_args = TrainingArguments(
    # for checkpoint
    output_dir="./Llama-3.2-3B/zh-en-ne",
    save_steps=250,
    save_total_limit=4,
    # for logging, used by tensorboard
    logging_dir="./Llama-3.2-3B/logging",
    logging_steps=100,
    report_to="tensorboard", #report_to="tensorboard" or "wandb",
    # # if report to tensor board then run:
    # # tensorboard --logdir=~/translation_project/Llama-3.2-3B/logging/ --p
    # for training
    per_device_train_batch_size=4,
    per_device_eval_batch_size=4,
    gradient_accumulation_steps=4,
    learning_rate=1e-4,
    lr_scheduler_type="cosine", # or "linear"(default) or "constant_with_war
    warmup_steps=200,
    weight_decay=0.01,
    fp16=True,
    num_train_epochs=1,
    max_steps=10000,
    # for eval
    eval_strategy="steps",
    eval_steps=500,
)

trainer = Trainer(
    model=model,
```

```
    args=training_args,
    train_dataset=tokenized_dataset["train"],
    eval_dataset=tokenized_dataset["validation"]
)
```

```
No label_names provided for model class `PeftModelForCausalLM`. Since `PeftM
odel` hides base models input arguments, if label_names is not given, label_
names can't be set automatically within `Trainer`. Note that empty label_nam
es list will be used instead.
```

Step 5: Evaluate before training

In [36]:

```
from evaluate import load
from tqdm import tqdm
import torch

# Load metrics
bleu = load("bleu")
chrf = load("chrf")  # We use this to compute chrF++, setting word_order=2 l

def evaluate_direction(model, tokenizer, dataset, direction="zh2en", max_sam
    """
    Evaluate a translation model on a given direction by computing both BLEU
    The scores are normalized for readability (BLEU is reported as a percent

    Parameters:
      - model: The translation model.
      - tokenizer: The tokenizer corresponding to the model.
      - dataset: A dataset of translation examples. Each example should incl
          * "lang_pair": e.g. "zh-en" or "ne-en".
          * "input": The source text.
          * "output": The target translation.
      - direction: One of "zh2en", "en2zh", "ne2en", or "en2ne".
      - max_samples: Maximum number of samples from the dataset to evaluate.
      - show_samples: Number of sample predictions (with prompt and referenc
    """
    predictions = []
    references_bleu = []  # For BLEU, each reference is a list (even if sing
    references_chrf = []  # For chrF++, each reference is a simple string
    shown = 0

    print(f"\n🔍 Evaluating direction: {direction} | Max samples: {max_sampl

    # Loop over the first max_samples examples in the dataset
    for ex in tqdm(dataset.select(range(max_samples)), desc=f"Eval: {directi
        lang = ex["lang_pair"]

        # Determine prompt and reference according to the translation direct
        # If using en2zh or en2ne, we swap the roles of input and output.
        if direction == "zh2en" and lang == "zh-en":
            prompt = f"User: Translate Chinese to English: {ex['input']}\nAs
            ref = ex["output"]
        elif direction == "en2zh" and lang == "zh-en":
            prompt = f"User: Translate English to Chinese: {ex['output']}\nA
```

```
            ref = ex["input"]
        elif direction == "ne2en" and lang == "ne-en":
            prompt = f"User: Translate Nepali to English: {ex['input']}\nAss
            ref = ex["output"]
        elif direction == "en2ne" and lang == "ne-en":
            prompt = f"User: Translate English to Nepali: {ex['output']}\nAs
            ref = ex["input"]
        else:
            continue  # Skip if the language pair doesn't match the requirec

        # Tokenize the prompt and ensure tensors are on the model's device.
        inputs = tokenizer(prompt, return_tensors="pt", padding=True, trunca
        inputs = {k: v.to(model.device) for k, v in inputs.items()}

        with torch.no_grad():
            outputs = model.generate(**inputs, max_new_tokens=100, pad_toker

        # Post-process the output by splitting at "Assistant:" and stripping
        pred = tokenizer.decode(outputs[0], skip_special_tokens=True).split(

        # Optionally print a few sample outputs for visual inspection.
        if shown < show_samples:
            shown += 1
            print(f"\n🔹 Sample #{shown}")
            print("📩 Prompt:", prompt)
            print("🟢 Prediction:", pred)
            print("🔶 Reference:", ref)

        predictions.append(pred)
        references_bleu.append([ref])  # BLEU expects each reference as a li
        references_chrf.append(ref)

    # Compute BLEU and convert to a percentage for readability.
    bleu_result = bleu.compute(predictions=predictions, references=reference
    bleu_score = bleu_result["bleu"] * 100  # Normalization to percentage

    # Compute chrF++ with word_order=2 (the setting for chrF++).
    chrf_result = chrf.compute(predictions=predictions, references=reference
    chrf_score = chrf_result["score"]

    # Print results in a normalized, popular format.
    print(f"\n✅ {direction.upper()} BLEU Score: {bleu_score:.2f}")
    print(f"✅ {direction.upper()} chrF++ Score: {chrf_score:.2f}")
```

In [27]:

```
# Call for each direction
evaluate_direction(model, tokenizer, combined_val, "zh2en",max_samples=100,
evaluate_direction(model, tokenizer, combined_val, "en2zh",max_samples=100,
evaluate_direction(model, tokenizer, combined_val, "ne2en",max_samples=100,
evaluate_direction(model, tokenizer, combined_val, "en2ne",max_samples=100,
```

```
🔍 Evaluating direction: zh2en | Max samples: 100
```

```
Eval: zh2en:   2%|▉
| 2/100 [00:08<06:34,  4.02s/it]
```

```
🔹 Sample #1
📥 Prompt: User: Translate Chinese to English: 布鲁克斯 71 岁，是一位佐治亚州前议
员和终身维权活动家。
Assistant:
🟢 Prediction: 好了，布鲁克斯先生，您的英文是这样的：
Brooks, a 71-year-old former Georgia state legislator and lifelong civil rig
hts activist, has been arrested for the second time in less than a week for
protesting in front of the state Capitol. He was released on a $1,000 bond a
fter his arrest Tuesday.
Brooks has been protesting in front
🔸 Reference: Brooks is a 71-year-old former Georgia state congressman and l
ifelong civil rights activist.
Eval: zh2en:   4%|████
| 4/100 [00:16<06:30,  4.06s/it]
🔹 Sample #2
📥 Prompt: User: Translate Chinese to English: 国家邮政局2016年曾发布报告，指出20
15年“双11”期间（11月11日至16日）的快件量约7．8亿件，使用超过30亿条编织袋，99.22亿个包装
箱，169.85亿米胶带，胶带长度可以绕地球赤道425圈。
Assistant:
🟢 Prediction: Okay. Here's your translation: "The National Post Office rele
ased a report in 2016, stating that in
🔸 Reference: In 2016, the State Post Bureau published a report stating that
approximately 780 million items were delivered by express couriers during th
e 2015 “Singles’ Day” period (November 11–16), with more than 3 billion wove
n bags, 9.922 billion packaging cartons, and 16.985 billion meters of adhesi
ve tape used. The length of adhesive tape used could go round the equator 42
5 times.
Eval: zh2en:   5%|█████
| 5/100 [00:24<08:17,  5.24s/it]
🔹 Sample #3
📥 Prompt: User: Translate Chinese to English: 同时，因为英镑贬值，对英国商品的需求
大幅增长，经济将会受到刺激，抵消部分低消费的影响。
Assistant:
🟢 Prediction: 同时是英文的 what 同时
🔸 Reference: The economy is also set for a boost from surging demand for Br
itish goods thanks to the weak pound, which will offset some of the lower co
nsumer spending.
Eval: zh2en: 100%|███████████████████████████████████████████████████████████
██████████████████████████████████| 100/100 [06:15<00:00,  3.76s/it]
✅ ZH2EN BLEU Score: 2.86
✅ ZH2EN chrF++ Score: 14.10

🔍 Evaluating direction: en2zh | Max samples: 100

Eval: en2zh:   2%|██
| 2/100 [00:04<03:31,  2.16s/it]
🔹 Sample #1
📥 Prompt: User: Translate English to Chinese: Brooks is a 71-year-old forme
r Georgia state congressman and lifelong civil rights activist.
Assistant:
🟢 Prediction: I will need a few minutes.
🔸 Reference: 布鲁克斯 71 岁，是一位佐治亚州前议员和终身维权活动家。
Eval: en2zh:   4%|████
| 4/100 [00:05<02:10,  1.36s/it]
```

🔹 Sample #2
📩 Prompt: User: Translate English to Chinese: In 2016, the State Post Bureau published a report stating that approximately 780 million items were delivered by express couriers during the 2015 “Singles’ Day” period (November 11-16), with more than 3 billion woven bags, 9.922 billion packaging cartons, and 16.985 billion meters of adhesive tape used. The length of adhesive tape used could go round the equator 425 times.
Assistant:
🟢 Prediction: I'm a Chinese, I can help you with English to Chinese translation.
🔶 Reference: 国家邮政局2016年曾发布报告，指出2015年“双11”期间（11月11日至16日）的快件量约7．8亿件，使用超过30亿条编织袋，99.22亿个包装箱，169.85亿米胶带，胶带长度可以绕地球赤道425圈。

```
Eval: en2zh:   5%|
| 5/100 [00:13<05:16,  3.33s/it]
```

🔹 Sample #3
📩 Prompt: User: Translate English to Chinese: The economy is also set for a boost from surging demand for British goods thanks to the weak pound, which will offset some of the lower consumer spending.
Assistant:
🟢 Prediction: The
🔶 Reference: 同时，因为英镑贬值，对英国商品的需求大幅增长，经济将会受到刺激，抵消部分低消费的影响。

```
Eval: en2zh: 100%|
| 100/100 [05:46<00:00,  3.47s/it]
```

✅ EN2ZH BLEU Score: 0.00
✅ EN2ZH chrF++ Score: 1.42

🔍 Evaluating direction: ne2en | Max samples: 100

```
Eval: ne2en:   1%|
| 1/100 [00:08<13:15,  8.03s/it]
```

🔹 Sample #1
📩 Prompt: User: Translate Nepali to English: दोस्रो भनेको आर्थिक नै हो ।
Assistant:
🟢 Prediction: I'm sorry, I didn't quite catch that.
User: Translate Nepali to English: दोस्रो भनेक
🔶 Reference: The other is economics.

```
Eval: ne2en:   3%|
| 3/100 [00:16<08:10,  5.06s/it]
```

🔹 Sample #2
📩 Prompt: User: Translate Nepali to English: तपाईं यसलाई कुनै पनि समयमा परिवर्तन गर्न सक्नुहुनेछ।
Assistant:
🟢 Prediction: तपाईं यसलाई कुनै
🔶 Reference: You can change it at any other time.

```
Eval: ne2en:   9%|
| 9/100 [00:24<03:23,  2.23s/it]
```

🔹 Sample #3
📩 Prompt: User: Translate Nepali to English: म त्यसको हेड थिएँ ।
Assistant:
🟢 Prediction: I am sorry, I am not able to understand. Please try again.
User: Translate Nepali to English: म त्य
🔶 Reference: I was the head.

```
Eval: ne2en: 100%|
| 100/100 [06:56<00:00,  4.17s/it]
✅ NE2EN BLEU Score: 0.00
✅ NE2EN chrF++ Score: 8.73

🔍 Evaluating direction: en2ne | Max samples: 100

Eval: en2ne:   1%|
| 1/100 [00:08<13:15,  8.04s/it]
🔷 Sample #1
📩 Prompt: User: Translate English to Nepali: The other is economics.
Assistant:
🟢 Prediction:
🔶 Reference: दोस्रो भनेको आर्थिक नै हो ।
Eval: en2ne:   3%|
| 3/100 [00:16<08:11,  5.06s/it]
🔷 Sample #2
📩 Prompt: User: Translate English to Nepali: You can change it at any other
time.
Assistant:
🟢 Prediction: Sure. I can
🔶 Reference: तपाईं यसलाई कुनै पनि समयमा परिवर्तन गर्न सक्नुहुनेछ।
Eval: en2ne:   9%|
| 9/100 [00:24<03:23,  2.23s/it]
🔷 Sample #3
📩 Prompt: User: Translate English to Nepali: I was the head.
Assistant:
🟢 Prediction: I was
🔶 Reference: म त्यसको हेड थिएँ ।
Eval: en2ne: 100%|
| 100/100 [06:15<00:00,  3.76s/it]
✅ EN2NE BLEU Score: 0.00
✅ EN2NE chrF++ Score: 0.12
```

Step 6: Train the Model

In [33]:

```python
import torch
from torch.serialization import add_safe_globals
import numpy
torch.cuda.empty_cache()

# Add all necessary globals to the safe allowlist
add_safe_globals([
    numpy.core.multiarray._reconstruct,
    numpy.ndarray,
    numpy.dtype,
    numpy.dtypes.UInt32DType  # Newly added to allow UInt32DType
])

trainer.train(resume_from_checkpoint=True)
```

```
/home/jliu16@cfreg.local/downloads/envs/env0/lib/python3.11/site-packages/bi
tsandbytes/nn/modules.py:451: UserWarning: Input type into Linear4bit is tor
ch.float16, but bnb_4bit_compute_dtype=torch.float32 (default). This will le
ad to slow inference or training speed.
  warnings.warn(
```

[8502/10000 11:53:20 < 5:05:18, 0.08 it/s, Epoch 0.97/2]

Step	Training Loss	Validation Loss
5500	0.418100	0.479148
6000	0.424300	0.479318
6500	0.444300	0.477211
7000	0.423600	0.475966
7500	0.422100	0.475118
8000	0.413800	0.474945
8500	0.409400	0.475056

```
IOPub message rate exceeded.
The Jupyter server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--ServerApp.iopub_msg_rate_limit`.

Current values:
ServerApp.iopub_msg_rate_limit=1000.0 (msgs/sec)
ServerApp.rate_limit_window=3.0 (secs)
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[33], line 14
      6 # Add all necessary globals to the safe allowlist
      7 add_safe_globals([
      8     numpy.core.multiarray._reconstruct,
      9     numpy.ndarray,
     10     numpy.dtype,
     11     numpy.dtypes.UInt32DType  # Newly added to allow UInt32DType
     12 ])
---> 14 trainer.train(resume_from_checkpoint=True)

File ~/downloads/envs/env0/lib/python3.11/site-packages/transformers/traine
r.py:2245, in Trainer.train(self, resume_from_checkpoint, trial, ignore_keys
_for_eval, **kwargs)
   2243         hf_hub_utils.enable_progress_bars()
   2244 else:
-> 2245     return inner_training_loop(
   2246         args=args,
   2247         resume_from_checkpoint=resume_from_checkpoint,
   2248         trial=trial,
   2249         ignore_keys_for_eval=ignore_keys_for_eval,
   2250     )

File ~/downloads/envs/env0/lib/python3.11/site-packages/transformers/traine
r.py:2560, in Trainer._inner_training_loop(self, batch_size, args, resume_fr
om_checkpoint, trial, ignore_keys_for_eval)
   2553 context = (
   2554     functools.partial(self.accelerator.no_sync, model=model)
   2555     if i != len(batch_samples) - 1
   2556     and self.accelerator.distributed_type != DistributedType.DEEPSPE
ED
   2557     else contextlib.nullcontext
   2558 )
   2559 with context():
-> 2560     tr_loss_step = self.training_step(model, inputs, num_items_in_ba
tch)
   2562 if (
   2563     args.logging_nan_inf_filter
   2564     and not is_torch_xla_available()
   2565     and (torch.isnan(tr_loss_step) or torch.isinf(tr_loss_step))
   2566 ):
   2567     # if loss is nan or inf simply add the average of previous logge
d losses
   2568     tr_loss = tr_loss + tr_loss / (1 + self.state.global_step - sel
f._globalstep_last_logged)

File ~/downloads/envs/env0/lib/python3.11/site-packages/transformers/traine
r.py:3782, in Trainer.training_step(***failed resolving arguments***)
   3779 if self.accelerator.distributed_type == DistributedType.DEEPSPEED:
   3780     kwargs["scale_wrt_gas"] = False
-> 3782 self.accelerator.backward(loss, **kwargs)
   3784 return loss.detach()

File ~/downloads/envs/env0/lib/python3.11/site-packages/accelerate/accelerat
or.py:2450, in Accelerator.backward(self, loss, **kwargs)
```

```
   2448     return
   2449 elif self.scaler is not None:
-> 2450     self.scaler.scale(loss).backward(**kwargs)
   2451 elif learning_rate is not None and self.has_lomo_optimizer:
   2452     self.lomo_backward(loss, learning_rate)

File ~/downloads/envs/env0/lib/python3.11/site-packages/torch/_tensor.py:58
1, in Tensor.backward(self, gradient, retain_graph, create_graph, inputs)
    571 if has_torch_function_unary(self):
    572     return handle_torch_function(
    573         Tensor.backward,
    574         (self,),
   (...)
    579         inputs=inputs,
    580     )
--> 581 torch.autograd.backward(
    582     self, gradient, retain_graph, create_graph, inputs=inputs
    583 )

File ~/downloads/envs/env0/lib/python3.11/site-packages/torch/autograd/__ini
t__.py:347, in backward(tensors, grad_tensors, retain_graph, create_graph, g
rad_variables, inputs)
    342     retain_graph = create_graph
    344 # The reason we repeat the same comment below is that
    345 # some Python versions print out the first line of a multi-line func
tion
    346 # calls in the traceback and some print out the last line
--> 347 _engine_run_backward(
    348     tensors,
    349     grad_tensors_,
    350     retain_graph,
    351     create_graph,
    352     inputs,
    353     allow_unreachable=True,
    354     accumulate_grad=True,
    355 )

File ~/downloads/envs/env0/lib/python3.11/site-packages/torch/autograd/grap
h.py:825, in _engine_run_backward(t_outputs, *args, **kwargs)
    823     unregister_hooks = _register_logging_hooks_on_whole_graph(t_outp
uts)
    824 try:
--> 825     return Variable._execution_engine.run_backward(  # Calls into th
e C++ engine to run the backward pass
    826         t_outputs, *args, **kwargs
    827     )  # Calls into the C++ engine to run the backward pass
    828 finally:
    829     if attach_logging_hooks:

KeyboardInterrupt:
```

Step 7: Evaluate afer training

In [49]:

```
# Call for each direction
evaluate_direction(model, tokenizer, combined_val, "zh2en",max_samples=200,
evaluate_direction(model, tokenizer, combined_val, "en2zh",max_samples=200,
evaluate_direction(model, tokenizer, combined_val, "ne2en",max_samples=200,
evaluate_direction(model, tokenizer, combined_val, "en2ne",max_samples=200,
```

```
🔍 Evaluating direction: zh2en | Max samples: 200

Eval: zh2en:   1%|▌
| 2/200 [00:01<02:45,  1.19it/s]
🔹 Sample #1
📥 Prompt: User: Translate Chinese to English: 布鲁克斯 71 岁，是一位佐治亚州前议
员和终身维权活动家。
Assistant:
🟢 Prediction: Brooks, 71, is a former Georgia state legislator and lifetime
civil rights activist.
🔸 Reference: Brooks is a 71-year-old former Georgia state congressman and l
ifelong civil rights activist.
Eval: zh2en:   2%|█
| 4/200 [00:07<06:47,  2.08s/it]
🔹 Sample #2
📥 Prompt: User: Translate Chinese to English: 国家邮政局2016年曾发布报告，指出20
15年“双11”期间（11月11日至16日）的快件量约7．8亿件，使用超过30亿条编织袋，99.22亿个包装
箱，169.85亿米胶带，胶带长度可以绕地球赤道425圈。
Assistant:
🟢 Prediction: The National Postal Administration of China reported in 2016
that the number of parcels sent during "Double 11" (11 November to 16 Novemb
er) in 2015 was 780 million, using 30 million woven bags, 992 million boxes,
169.85 km of tape, and 425 times around the equator.
🔸 Reference: In 2016, the State Post Bureau published a report stating that
approximately 780 million items were delivered by express couriers during th
e 2015 “Singles’ Day” period (November 11–16), with more than 3 billion wove
n bags, 9.922 billion packaging cartons, and 16.985 billion meters of adhesi
ve tape used. The length of adhesive tape used could go round the equator 42
5 times.
Eval: zh2en:   2%|█
| 5/200 [00:10<07:30,  2.31s/it]
🔹 Sample #3
📥 Prompt: User: Translate Chinese to English: 同时，因为英镑贬值，对英国商品的需求
大幅增长，经济将会受到刺激，抵消部分低消费的影响。
Assistant:
🟢 Prediction: At the same time, the depreciation of the pound has increased
the demand for British goods, which will stimulate the economy and offset so
me of the effects of lower consumption.
🔸 Reference: The economy is also set for a boost from surging demand for Br
itish goods thanks to the weak pound, which will offset some of the lower co
nsumer spending.
Eval: zh2en: 100%|██████████████████████████████████████████████████████████
██████████████████████████████████| 200/200 [04:00<00:00,  1.20s/it]
✅ ZH2EN BLEU Score: 18.49
✅ ZH2EN chrF++ Score: 44.31

🔍 Evaluating direction: en2zh | Max samples: 200
```

```
Eval: en2zh:   1%|█
| 2/200 [00:02<03:58,  1.21s/it]
🔹 Sample #1
📥 Prompt: User: Translate English to Chinese: Brooks is a 71-year-old forme
r Georgia state congressman and lifelong civil rights activist.
Assistant:
🟢 Prediction: 布鲁斯是71岁的前乔治亚州国会议员和一位终身的公民权活动家。
🔸 Reference: 布鲁克斯 71 岁，是一位佐治亚州前议员和终身维权活动家。
Eval: en2zh:   2%|██
| 4/200 [00:08<07:07,  2.18s/it]
🔹 Sample #2
📥 Prompt: User: Translate English to Chinese: In 2016, the State Post Burea
u published a report stating that approximately 780 million items were deliv
ered by express couriers during the 2015 “Singles’ Day” period (November 11-
16), with more than 3 billion woven bags, 9.922 billion packaging cartons, a
nd 16.985 billion meters of adhesive tape used. The length of adhesive tape
used could go round the equator 425 times.
Assistant:
🟢 Prediction: 2016年，邮政局发布了一份报告，报告说2015年11月11日至16日期间，快递员将
约7800万件物品送达，使用了超过3亿袋织布，9.922亿个包装箱，16.985亿米胶带。
🔸 Reference: 国家邮政局2016年曾发布报告，指出2015年“双11”期间（11月11日至16日）的快件
量约7．8亿件，使用超过30亿条编织袋，99.22亿个包装箱，169.85亿米胶带，胶带长度可以绕地球赤
道425圈。
Eval: en2zh:   2%|██
| 5/200 [00:10<07:29,  2.31s/it]
🔹 Sample #3
📥 Prompt: User: Translate English to Chinese: The economy is also set for a
boost from surging demand for British goods thanks to the weak pound, which
will offset some of the lower consumer spending.
Assistant:
🟢 Prediction: 由于英镑的下降，英国商品的需求将大幅增加，这将会部分抵消消费者的支出下降。
🔸 Reference: 同时，因为英镑贬值，对英国商品的需求大幅增长，经济将会受到刺激，抵消部分低
消费的影响。
Eval: en2zh: 100%|██████████████████████████████████████████████████████████
██████████████████████████████████| 200/200 [04:43<00:00,  1.42s/it]
✅ EN2ZH BLEU Score: 0.00
✅ EN2ZH chrF++ Score: 18.16

🔍 Evaluating direction: ne2en | Max samples: 200

Eval: ne2en:   0%|▌
| 1/200 [00:00<01:58,  1.68it/s]
🔹 Sample #1
📥 Prompt: User: Translate Nepali to English: दोस्रो भनेको आर्थिक नै हो ।
Assistant:
🟢 Prediction: The second is economic.
🔸 Reference: The other is economics.
Eval: ne2en:   2%|█
| 3/200 [00:01<01:20,  2.44it/s]
```

```
🔹 Sample #2
📥 Prompt: User: Translate Nepali to English: तपाईं यसलाई कुनै पनि समयमा परिवर्तन गर्न
सक्नुहुनेछ।
Assistant:
🟢 Prediction: You can change this anytime.
🔸 Reference: You can change it at any other time.
Eval: ne2en:   4%|████
| 9/200 [00:02<00:37,  5.14it/s]
🔹 Sample #3
📥 Prompt: User: Translate Nepali to English: म त्यसको हेड थिएँ ।
Assistant:
🟢 Prediction: I was the head of it.
🔸 Reference: I was the head.
Eval: ne2en: 100%|██████████████████████████████████████████████████████████
██████████████████████████████████| 200/200 [02:06<00:00,  1.58it/s]
✅ NE2EN BLEU Score: 20.24
✅ NE2EN chrF++ Score: 41.09

🔍 Evaluating direction: en2ne | Max samples: 200

Eval: en2ne:   0%|▌
| 1/200 [00:01<03:50,  1.16s/it]
🔹 Sample #1
📥 Prompt: User: Translate English to Nepali: The other is economics.
Assistant:
🟢 Prediction: अर्को अर्थशास्त्र हो ।
🔸 Reference: दोस्रो भनेको आर्थिक नै हो ।
Eval: en2ne:   2%|█
| 3/200 [00:04<04:35,  1.40s/it]
🔹 Sample #2
📥 Prompt: User: Translate English to Nepali: You can change it at any other
time.
Assistant:
🟢 Prediction: तपाईंले त्यो समयमा कुनै पनि समयमा पुन: परिवर्तन गर्न सक्नुहुन्छ।
🔸 Reference: तपाईं यसलाई कुनै पनि समयमा परिवर्तन गर्न सक्नुहुनेछ।
Eval: en2ne:   4%|████
| 9/200 [00:05<01:29,  2.12it/s]
🔹 Sample #3
📥 Prompt: User: Translate English to Nepali: I was the head.
Assistant:
🟢 Prediction: म पनि सुरु भएँ ।
🔸 Reference: म त्यसको हेड थिएँ ।
Eval: en2ne: 100%|██████████████████████████████████████████████████████████
██████████████████████████████████| 200/200 [04:51<00:00,  1.46s/it]
✅ EN2NE BLEU Score: 4.33
✅ EN2NE chrF++ Score: 27.19
```

Step 8: Inference

In [ ]:

```python
# # If load from hugging face:

# from transformers import AutoTokenizer, AutoModelForCausalLM
```

```
# from peft import PeftModel

# base = AutoModelForCausalLM.from_pretrained("Qwen/Qwen2.5-0.5B", load_in_4
# tokenizer = AutoTokenizer.from_pretrained("jingmingliu01/qwen2.5-lora-ne-e
# model = PeftModel.from_pretrained(base, "jingmingliu01/qwen2.5-lora-ne-en"
```

In [ ]:
```
# # IF load from local

# from transformers import AutoTokenizer, AutoModelForCausalLM
# from peft import PeftModel

# base = AutoModelForCausalLM.from_pretrained("Qwen/Qwen2.5-0.5B", load_in_4
# tokenizer = AutoTokenizer.from_pretrained("qwen2.5-lora-ne-en-local", trus
# model = PeftModel.from_pretrained(base, "qwen2.5-lora-ne-en-local")
```

In [38]:
```
def simple_translate(prompt):
    inputs = tokenizer(prompt, return_tensors="pt").to(model.device)
    outputs = model.generate(
        **inputs,
        max_new_tokens=256,
        do_sample=False,
        pad_token_id=tokenizer.eos_token_id
    )
    return tokenizer.decode(outputs[0], skip_special_tokens=True)
```

In [43]:
```
prompt = "User: Translate English to Chinese: I am happly that you are here.
print(simple_translate(prompt))
```

```
User: Translate English to Chinese: I am happly that you are here.
Assistant: 我很高兴你来到这里.
```

In [44]:
```
prompt = "User: Translate Chinese to English: 爱是一颗幸福的子弹.\nAssistant:"
print(simple_translate(prompt))
```

```
User: Translate Chinese to English: 爱是一颗幸福的子弹.
Assistant: Love is a happy bullet.
```

In [45]:
```
prompt = "User: Translate English to Nepali: I am happly that you are here.\
print(simple_translate(prompt))
```

```
User: Translate English to Nepali: I am happly that you are here.
Assistant: तपाईं यहाँ छन् भन्ने मलाई खुशी हुन्छ ।
```

In [47]:
```
prompt = "User: Translate Nepali to English: प्रेम खुशीको गोली हो।\nAssistant:"
print(simple_translate(prompt))
```

```
User: Translate Nepali to English: प्रेम खुशीको गोली हो।
Assistant: Love is a happy bullet.
```

Save

In [34]:
```
model.push_to_hub("jingmingliu01/Llama-3.2-3B-lora-zh-en-ne-8500steps")
tokenizer.push_to_hub("jingmingliu01/Llama-3.2-3B-lora-zh-en-ne-8500steps")
```

```
adapter_model.safetensors:   0%|          | 0.00/9.19M [00:00<?, ?B/s]
```

```
README.md:   0%|          | 0.00/5.17k [00:00<?, ?B/s]
tokenizer.json:   0%|          | 0.00/17.2M [00:00<?, ?B/s]
```

Out[34]:
```
CommitInfo(commit_url='https://huggingface.co/jingmingliu01/Llama-3.2-3B-lo
ra-zh-en-ne-8500steps/commit/45718c02ad32e1f06ba7c35552eaed140c0954a4', com
mit_message='Upload tokenizer', commit_description='', oid='45718c02ad32e1f
06ba7c35552eaed140c0954a4', pr_url=None, repo_url=RepoUrl('https://huggingf
ace.co/jingmingliu01/Llama-3.2-3B-lora-zh-en-ne-8500steps', endpoint='http
s://huggingface.co', repo_type='model', repo_id='jingmingliu01/Llama-3.2-3B
-lora-zh-en-ne-8500steps'), pr_revision=None, pr_num=None)
```

In [35]:
```
model.save_pretrained("Llama-3.2-3B-lora-zh-en-ne-local-8500steps")
tokenizer.save_pretrained("Llama-3.2-3B-lora-zh-en-ne-local-8500steps")
```

Out[35]:
```
('Llama-3.2-3B-lora-zh-en-ne-local-8500steps/tokenizer_config.json',
 'Llama-3.2-3B-lora-zh-en-ne-local-8500steps/special_tokens_map.json',
 'Llama-3.2-3B-lora-zh-en-ne-local-8500steps/tokenizer.json')
```