모델 학습을 위한 데이터 전처리

- 데이터 로드

```python
def load_dataset(self, dataset_id, tokenizer):
    def tokenization(examples):
        sources = []
        targets = []
        prompt = self.PROMPT_TEMPLATE
        for instruction, input, output in zip(examples['instruction'],examples['input'],examples['output']):
            if input is not None and input !="":
                instruction = instruction+'\n'+input
            source = prompt.format_map({'instruction':instruction})
            target = f"{output}{tokenizer.eos_token}"

            sources.append(source)
            targets.append(target)

        tokenized_sources = tokenizer(sources,return_attention_mask=False)
        tokenized_targets = tokenizer(targets,return_attention_mask=False,add_special_tokens=False)

        all_input_ids = []
        all_labels = []
        for s,t in zip(tokenized_sources['input_ids'],tokenized_targets['input_ids']):
            input_ids = torch.LongTensor(s + t)[:self.MAX_SEQ_LEN]
            labels = torch.LongTensor([-100] * len(s) + t)[:self.MAX_SEQ_LEN]
            assert len(input_ids) == len(labels)
            all_input_ids.append(input_ids)
            all_labels.append(labels)

        results = {'input_ids':all_input_ids, 'labels': all_labels}
        return results

    all_datasets = []

    raw_dataset = load_dataset(dataset_id)
    tokenization_func = tokenization
```

Hugging Face Hub Model load

- 토큰화

```python
def load_dataset(self, dataset_id, tokenizer):
    def tokenization(examples):
        sources = []
        targets = []
        prompt = self.PROMPT_TEMPLATE
        for instruction, input, output in zip(examples['instruction'],examples['input'],examples['output']):
            if input is not None and input !="":
                instruction = instruction+'\n'+input
            source = prompt.format_map({'instruction':instruction})
            target = f"{output}{tokenizer.eos_token}"

            sources.append(source)
            targets.append(target)

        tokenized_sources = tokenizer(sources,return_attention_mask=False)
        tokenized_targets = tokenizer(targets,return_attention_mask=False,add_special_tokens=False)

        all_input_ids = []
        all_labels = []
        for s,t in zip(tokenized_sources['input_ids'],tokenized_targets['input_ids']):
            input_ids = torch.LongTensor(s + t)[:self.MAX_SEQ_LEN]
            labels = torch.LongTensor([-100] * len(s) + t)[:self.MAX_SEQ_LEN]
            assert len(input_ids) == len(labels)
            all_input_ids.append(input_ids)
            all_labels.append(labels)

        results = {'input_ids':all_input_ids, 'labels': all_labels}
        return results

    all_datasets = []

    raw_dataset = load_dataset(dataset_id)
    tokenization_func = tokenization
```

Tokenizer Text to Number ID

GitHub 주소: https://github.com/EDDI-RobotAcademy/SI-Follow-LLAMA/tree/main/dataset

모델링 및 배포

- 모델 로드 및 설정

```python
    def load_model(
        self, model_id, model_config, quantize_config: Optional[QuantizeArgs] = None
    ):
        model = AutoModelForCausalLM.from_pretrained(
            model_id,
            torch_dtype=model_config.torch_dtype,
            low_cpu_mem_usage=True,
            use_cache=False,
            device_map=0,
            max_length=model_config.max_length,
            quantization_config=quantize_config,
        )
        model.gradient_checkpointing_enable()
        model.enable_input_require_grads()
        model.is_parallelizable = False
        model.model_parallel = False
        if quantize_config is not None:
            model = prepare_model_for_kbit_training(model)

        return model
```

모델을 로드하고 양자화

메모리 최적화

```python
    def get_lora_target(self, lora_target_config):
        return lora_target_config.targets

    def get_lora_config(self, lora_config, lora_targets: List):
        lora_config = LoraConfig(
            task_type=TaskType.CAUSAL_LM,
            inference_mode=False,
            r=lora_config.r,
            lora_alpha=lora_config.alpha,
            lora_dropout=lora_config.dropout,
            target_modules=lora_targets,
            bias="none",
        )
        return lora_config
```

LoRA 설정 정의

- Custom 모델 학습

```python
    def get_train_args(self, train_args):
        args = TrainingArguments(
            output_dir=train_args.output_dir,
            run_name=train_args.run_name,
            logging_steps=train_args.log_interval,
            num_train_epochs=train_args.epochs,
            per_device_train_batch_size=train_args.batch_size,
            per_device_eval_batch_size=train_args.batch_size,
            gradient_accumulation_steps=train_args.gradient_accumulation_steps,
            dataloader_num_workers=train_args.num_workers,
            learning_rate=train_args.lr,
            weight_decay=train_args.weight_decay,
            warmup_ratio=train_args.warmup_ratio,
            lr_scheduler_type=train_args.lr_scheduler_type,
            save_steps=train_args.save_steps,
            eval_steps=train_args.eval_steps,
            fp16=train_args.fp16,
            evaluation_strategy=train_args.evaluation_strategy,
            remove_unused_columns=False,
            report_to=train_args.report_to,
        )
        return args
```

훈련에 사용할 파라미터 정의

```python
    def get_trainer(
        self,
        model,
        tokenizer,
        train_args,
        train_dataset,
        eval_dataset,
        callbacks,
        data_collator,
    ):
        trainer = Trainer(
            model=model,
            tokenizer=tokenizer,
            args=train_args,
            train_dataset=train_dataset,
            eval_dataset=eval_dataset,
            callbacks=callbacks,
            data_collator=data_collator,
        )
        return trainer
```

메서드를 호출해 학습

GitHub 주소: https://github.com/EDDI-RobotAcademy/SI-Follow-LLAMA/tree/main/train

모델링 및 배포

- 학습된 모델 저장 및 배포

```
134        def train(self, trainer):
135            trainer.train()  학습 완료된 모델 경로에 저장
136
137        def save_model(self, model, save_path):
138            model.save_pretrained(save_path, from_pt=True)
139
140        def model_to_huggingface(self, model, model_id):
141            model.push_to_hub(model_id)
142
143        def tokenizer_to_huggingface(self, tokenizer, model_id):
144            tokenizer.push_to_hub(model_id)
```

Huggingface Hub에 모델과 토크나이저를 업로드 하고 경로에 배포
다른 사용자도 Hub에서 Custom한 모델을 다운로드해서 사용 할 수 있음

GitHub 주소: https://github.com/EDDI-RobotAcademy/SI-Follow-LLAMA/tree/main/train