微调实战-1： Embedding 微调

第1步：创建虚拟环境

```
conda create -n swift_venv python=3.12 -y
conda init bash && source /root/.bashrc
conda activate swift_venv
conda install ipykernel
ipython kernel install --user --name=swift_venv
```

第2步：安装 ms-swift

```
pip install git+https://github.com/modelscope/ms-swift.git
```

第3步：安装其他依赖

```
pip install deepspeed liger-kernel
pip install scikit-learn
pip install -U sentence-transformers

# 建议科学上网后，再执行下面的命令👇
pip install flash-attn --no-build-isolation
```

第4步：下载训练数据集

```
# 设置下面命令，无需科学上网
export HF_ENDPOINT=https://hf-mirror.com

pip install --upgrade huggingface_hub

huggingface-cli download --repo-type dataset --resume-download microsoft/ms_marco --local-dir /root/autodl-tmp/dataset/stsb
```

数据格式：

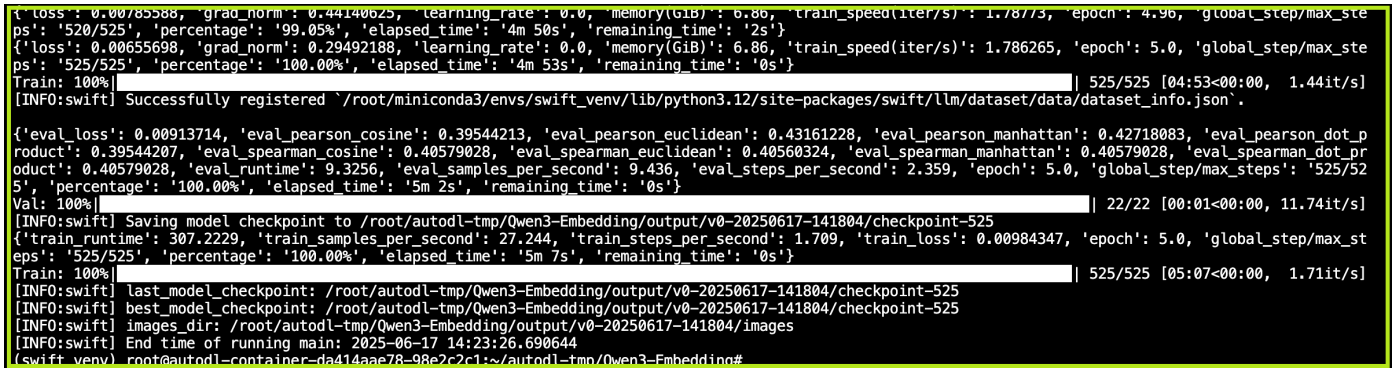
```
{
    "sentence1": "A plane is taking off.",
    "sentence2": "An air plane is taking off.",
    "score": 1.0
}
```

第5步：开始 embedding 模型训练

方法-1:全参数训练

```
# 每个节点使用的GPU数量
nproc_per_node=1
NPROC_PER_NODE=$nproc_per_node \
swift sft \
    # 预训练模型路径
    --model /root/autodl-tmp/models/Qwen3-Embedding-0.6B \
    # 任务类型为embedding训练
    --task_type embedding \
    # 模型架构类型
    --model_type qwen3_emb \
    # 全参数微调
    --train_type full \
    # 训练数据集
    --dataset /root/autodl-tmp/dataset/stsb:positive  \
    # 将5%的训练数据用作验证集
    --split_dataset_ratio 0.05 \
    # 按训练步数进行评估
    --eval_strategy steps \
    # 指定模型输出目录
    --output_dir output \
    #  每200步保存一次模型检查点
    --eval_steps 200 \
    # 训练轮次
    --num_train_epochs 5 \
    # 每200步保存一次模型检查点
    --save_steps 200 \
    # 每个GPU的训练批次大小
    --per_device_train_batch_size 8 \
    # 每个GPU的评估批次大小
    --per_device_eval_batch_size 4 \
    # 梯度累积步数
    --gradient_accumulation_steps 2 \
    # 学习率
    --learning_rate 6e-6 \
    # 损失函数类型，需要label字段（相似度分数）
    --loss_type cosine_similarity  \
    # 指定标签字段名称
```

```
    --label_names labels \
    # 是否丢弃最后一个不完整的批次，保证批次大小一致
    --dataloader_drop_last true
    # 多卡分布式
    # --deepspeed zero3
```



✨ **损失函数类型对比：**

MS-SWIFT 框架支持多种损失函数，用于训练 Embedding。选择合适的损失函数对模型性能至关重要，不同的损失函数适用于不同的数据格式和训练目标。

1. InfoNCE 损失

`--loss_type infonce`

原理：

- 对比学习损失函数，最大化正样本对相似度，最小化负样本对相似度
- 使用批内对比学习策略，将同批次内其他样本作为负样本

数据格式：

```
# 不含负样本
{"query": "sentence1", "response": "sentence1-positive"}

# 含多个负样本
{"query": "sentence1", "response": "sentence1-positive", "rejected_response":
["sentence1-negative1", "sentence1-negative2"]}
```

环境变量配置：

```
export INFONCE_TEMPERATURE=0.01          # 温度参数，控制相似度分布锐利度
export INFONCE_USE_BATCH=True            # 使用批内负样本
export INFONCE_HARD_NEGATIVES=False      # 困难负样本数量标准化
export INFONCE_MASK_FAKE_NEGATIVE=False # 屏蔽假负样本
```

2. 余弦相似度损失

`--loss_type cosine_similarity`

原理:

- 直接优化预测相似度与真实相似度标签的差异
- 使用 MSE 损失计算 `||input_label - cosine_sim(u,v)||_2`

数据格式:

```
{"query": "sentence1", "response": "sentence2", "label": 0.8}
```

3. 对比学习损失

`--loss_type contrastive`

原理:

- 经典的对比学习损失，正样本拉近，负样本推远
- 需要设置 margin 参数

数据格式:

```
{"query": "sentence1", "response": "sentence2", "label": 1}  # 1表示正样本，0表示负样本
```

4. 在线对比学习损失

`--loss_type online_contrastive`

原理:

- 对比学习的改进版本，选择困难正样本和困难负样本
- 通常比标准对比学习效果更好

数据格式:

```
{"query": "sentence1", "response": "sentence2", "label": 1}  # 1表示正样本，0表示负样本
```

---

5. 训练数据类型对应的损失函数

数据类型	推荐损失函数	原因
只有正样本对	`infonce`	利用批内负样本，训练信号强
正负样本对	`contrastive` 或 `online_contrastive`	直接利用标注的负样本
连续相似度分数	`cosine_similarity`	直接优化相似度预测

方法-2: LoRA微调

```
# （1）LoRA 训练
nproc_per_node=1
NPROC_PER_NODE=$nproc_per_node \
swift sft \
    --model /root/autodl-tmp/models/Qwen3-Embedding-0.6B \
    --task_type embedding \
    --model_type qwen3_emb \
    --train_type lora \
    --lora_rank 16 \
    --lora_alpha 32 \
    --lora_dropout 0.1 \
    --dataset sentence-transformers/stsb:positive  \
    --split_dataset_ratio 0.05 \
    --eval_strategy steps \
    --output_dir output \
    --eval_steps 200 \
    --num_train_epochs 5 \
    --save_steps 200 \
    --per_device_train_batch_size 8 \
    --per_device_eval_batch_size 4 \
    --gradient_accumulation_steps 2 \
    --learning_rate 6e-6 \
    --loss_type cosine_similarity  \
    --label_names labels \
    --dataloader_drop_last true
```

```
# （2）合并 LoRA 权重
swift export --adapters /root/autodl-tmp/Qwen3-Embedding/output/v1-20250619-
131635/checkpoint-525 --merge_lora true --output_dir /root/autodl-tmp/Qwen3-
Embedding/output/merged_model --safe_serialization true  --exist_ok true
```

```
[INFO:swift] Start time of running main: 2025-06-17 14:51:29.329702
[INFO:swift] swift.__version__: 3.6.0.dev0
[INFO:swift] merge_device_map: None
[INFO:swift] Loading the model using model_dir: /root/autodl-tmp/models/Qwen3-Embedding-0.6B
[INFO:swift] model_kwargs: {'device_map': 'cuda:0'}
[2025-06-17 14:51:30,894] [INFO] [real_accelerator.py:254:get_accelerator] Setting ds_accelerator to cuda (auto detect)
[2025-06-17 14:51:32,266] [INFO] [logging.py:107:log_dist] [Rank -1] [TorchCheckpointEngine] Initialized with serialization = False
[INFO:swift] default_system: None
[INFO:swift] max_length: 2048
[INFO:swift] response_prefix: ''
[INFO:swift] agent_template: react_en
[INFO:swift] Merge LoRA...
[INFO:swift] Saving merged weights...
[INFO:swift] Successfully merged LoRA and saved in /root/autodl-tmp/Qwen3-Embedding/output/merged_model.
[INFO:swift] End time of running main: 2025-06-17 14:51:34.805782
(swift_venv) root@autodl-container-da414aae78-98e2c2c1:~/autodl-tmp/Qwen3-Embedding#
```

```
# （3）测试合并后的模型

# 1️⃣ 修改脚本：examples/qwen3_embedding_transformers.py 中模型加载路径：
model_path = "/root/autodl-tmp/Qwen3-Embedding/output/merged_model"

# 2️⃣ 执行命令
python examples/qwen3_embedding_transformers.py
```

```
(swift_venv) root@autodl-container-da414aae78-98e2c2c1:~/autodl-tmp/Qwen3-Embedding# python examples/qwen3_embedding_transformers.py
query outputs:
 tensor([[-0.0296, -0.0708, -0.0015,  ...,  0.0666,  0.0197,  0.0002],
        [-0.0249, -0.0555, -0.0026,  ..., -0.0049, -0.0028, -0.0042]],
       device='cuda:0', dtype=torch.float16, grad_fn=<DivBackward0>)
doc outputs:
 tensor([[-0.0030, -0.0471,  0.0038,  ...,  0.0720,  0.0311, -0.0100],
        [-0.0258, -0.0739, -0.0027,  ...,  0.0163, -0.0275,  0.0188]],
       device='cuda:0', dtype=torch.float16, grad_fn=<DivBackward0>)
相似度得分: [[75.3125, 21.5], [17.46875, 84.0625]]
(swift_venv) root@autodl-container-da414aae78-98e2c2c1:~/autodl-tmp/Qwen3-Embedding#
```

微调实战-2： ReRanker 微调

```
# 请参考:        Qwen3-Reranker-SFT
```