UBAI PRACTICE - T2T Generation

hs.hwang

2025-02-21

Table of contents

1 CNN

.

Text-to-text Generation , input output (text) .

Transformer (Language Model) GPT-2 .

, Transformer .

Transformer

Transformer 2017 “Attention Is All You Need” . (NLP) , RNN, LSTM .

Transformers Hugging Face (NLP) . AI (: BERT, GPT, T5) .

1.

.

(job) , `filename.sh` . , Shell `t2t.sh` .

`.sh` .

```
#!/bin/bash
#SBATCH --job-name=t2t_g
#SBATCH --output=./output/t2t_%n_%j.out
#SBATCH --error=./output/t2t_%n_%j.err
#SBATCH --nodes=1
#SBATCH --partition=gpu4
#SBATCH --gres=gpu:4
#SBATCH --cpus-per-task=16
#SBATCH --mem=128G
#SBATCH --time=24:00:00

echo "start at:" `date` #
echo "node: $HOSTNAME" #
echo "jobid: $SLURM_JOB_ID" # jobid
```

```
# Load modules (cuda  )
module load cuda/11.8 #

# Load env (python  )
source ~/miniconda3/bin/activate ubai
pip install --upgrade pip  # pip
pip install -r requirements.txt #

#
python t2t.py
```

. .

STDOUT , STDERR . (directory)
.

.

#SBATCH --nodes=1

- , .
- nodes=1 .

#SBATCH --partition=gpu4

- Partition .

#SBATCH --cpus-per-task=14

- .
- n , 1 CPU/GPU .

#SBATCH --gres=gpu:1

- GPU .
- CPU Partition . GPU , .

#SBATCH --job-name=UBAIJOB

- .

echo "start at:" 'date'

- .

echo "node: $HOSTNAME"

- .

```
echo "jobid: $SLURM_JOB_ID"
```

- jobid .

```
module ~
```

- Linux .
- CUDA/11.2.2 . GPU , unload , load .
- GPU , GPU (CPU Partition) .

```
source ~/miniconda3/bin/activate ubai
```

- ubai python .
  - ubai , .
- , activate conda .

```
pip install -r requirements.txt
```

- python , .
- conda , , .

```
python t2t.py
```

- Python .
- .py .

2.

, Python .

, terminal `sbatch` . (job) .

(job) ID .

```
sbatch t2t.sh    # sbatch filename.sh
```

(job) , STDOUT OUT .

OUT , Partition (job) . terminal `squeue` , ID .

n001, n002 … , (*Resources, Priority*) .

, .

Partition Partition cpus-per-task, gpu Partition (job) .
STDOUT OUT .

2 t2t.py

```
from transformers import GPT2LMHeadModel, GPT2Tokenizer #

model_name = 'gpt2' #
tokenizer = GPT2Tokenizer.from_pretrained(model_name) #
model = GPT2LMHeadModel.from_pretrained(model_name) #       GPT-2

input_text = "I want to eat some delicious food." #

inputs = tokenizer(input_text, return_tensors = 'pt') # pytorch
outputs = model.generate(**inputs, max_length=50) #

print("   ")
print(tokenizer.decode(outputs[0], skip_special_tokens=True)) #
```