KohakuBlueleaf 2 months ago

168 lines (133 loc) · 7.06 KB

Preview Code Blame

LyCORIS - Lora beYond Conventional methods, Other Rank adaptation Implementations for Stable diffusion.

(This image is generated by the model trained in Hadamard product representation)

A project for implementing different algorithm to do parameter-efficient finetuning on stable diffusion or more.

This project is started from LoCon(see archive branch).

What we have now

See Algo.md or Demo.md for more example and explanation

Conventional LoRA

- Include Conv layer implementation from LoCon
- recommended settings
  - dim <= 64
  - alpha = 1 (or lower, like 0.3)

LoRA with Hadamard Product representation (LoHa)

- Ref: FedPara Low-Rank Hadamard Product For Communication-Efficient Federated Learning
- designed for federated learning, but has some cool property like rank<=dim^2 so should be good for parameter-efficient finetuning.
  - Conventional LoRA is rank<=dim
- recommended settings
  - dim <= 32
  - alpha = 1 (or lower)

WARNING: You are not supposed to use dim>64 in LoHa, which is over sqrt(original_dim) for almost all layer in SD

High dim with LoHa may cause unstable loss or just goes to NaN. If you want to use high dim LoHa, please use lower lr

WARNING-AGAIN: Use parameter-efficient algorithim in parameter-unefficient way is not a good idea

(IA)^3

- Ref: Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning
- You can try this algo with dev version package(or install from source code) and set algo=ia3
- This algo need much higher lr (about 5e-3~1e-2)
- This algo is good at learning style, but hard to transfer(You can only get reasonable result on the model you trained on).

- This algo produce very tiny file(about 200~300KB)
- **Experimental**

LoKR

- Basically same idea of LoHA, but use kronecker product
- This algo is quite sensitive and you may need to tune the lr
- This algo can learn both character and style, but since it is small (auto factor, full rank, 2.5MB), it is also hard to transfer.
- This algo produce relatively small file(auto factor: 900~2500KB)
- Use smaller factor will produce bigger file, you can tune it if you think 2.5MB full rank is not good enough.

DyLoRA

- Ref DyLoRA: Parameter Efficient Tuning of Pre-trained Models using Dynamic Search-Free Low Rank Adaptation
- Basically a training trick of lora.
- Every step, only update one row/col of LoRA weight.
- When we want to update k row/col, we only use 0~k row/col to rebuild the weight (0<=k<=dim)
- You can easily resize DyLoRA to target and get similar or even better result than LoRA trained at target dim. (And you don't need to train lot loras with different dim to check which is better)
- You should use large dim with alpha=dim/4~dim (1 or dim is not very recommended)
  - Example: dim=128, alpha=64
- Since we only update 1 row/col each step, you will need more step to get reasonable result. If you want to train it with few steps, you may need to set block_size (update multiple row/col every step) to higer value (default=0)

---

usage

You can just use these training scripts.

- derrian-distro/LoRA_Easy_Training_Scripts
- Linaqruf/kohya-trainer
- bmaltais/kohya_ss
- hollowstrawberry/kohya-colab

For kohya script

Activate sd-scripts' venv and then install this package

```
source PATH_TO_SDSCRIPTS_VENV/Scripts/activate
```

or

```
PATH_TO_SDSCRIPTS_VENV\Scripts\Activate.ps1 # or .bat for cmd
```

And then you can install this package:

- through pip

```
pip install lycoris_lora
```

- from source

```
git clone https://github.com/KohakuBlueleaf/LyCORIS
cd LyCORIS
pip install .
```

Finally you can use this package's kohya module to run kohya's training script

```
python3 sd-scripts/train_network.py \
  --network_module lycoris.kohya \
  --network_dim "DIM_FOR_LINEAR" --network_alpha "ALPHA_FOR_LINEAR"\
  --network_args "conv_dim=DIM_FOR_CONV" "conv_alpha=ALPHA_FOR_CONV" \
  "dropout=DROPOUT_RATE" "algo=locon" \
```

to train lycoris module for SD model

- algo list:
  - locon: Conventional Methods
  - loha: Hadamard product representation introduced by FedPara
  - lokr: Kronecker product representation
  - ia3 : (IA)^3
- Tips:
  - Use network_dim=0 or conv_dim=0 to disable linear/conv layer
  - LoHa/LoKr/(IA)^3 doesn't support dropout yet.

For a1111's sd-webui

download Extension into sd-webui, and then use LyCORIS model in the extra netowrks tabs.

Not For Kohya-ss' Additional Network

Extract LoCon

You can extract LoCon from a dreambooth model with its base model.

```
python3 extract_locon.py <settings> <base_model> <db_model> <output>
```

Use --help to get more info

```
$ python3 extract_locon.py --help
usage: extract_locon.py [-h] [--is_v2] [--device DEVICE] [--mode
MODE] [--safetensors] [--linear_dim LINEAR_DIM] [--conv_dim CONV_DIM]
                        [--linear_threshold LINEAR_THRESHOLD] [--
conv_threshold CONV_THRESHOLD] [--linear_ratio LINEAR_RATIO] [--
conv_ratio CONV_RATIO]
                        [--linear_percentile LINEAR_PERCENTILE] [--
conv_percentile CONV_PERCENTILE]
                        base_model db_model output_name
```

Example and Comparing for different algo

see Demo.md and Algo.md

Change Log

For full log, please see Change.md

2023/04/08 Update for 0.1.5

- Add (IA)^3 algorithm
- Add lokr algorithm

Todo list

- [ ] Module and Document for using LyCORIS in any other model, Not only SD.
- [x] Proposition3 in FedPara
  - also need custom backward to save the vram
- [ ] Low rank + sparse representation
  - [x] For extraction
  - [ ] For training

- [ ] Support more operation, not only linear and conv2d.
- [ ] Configure varying ranks or dimensions for specific modules as needed.
- [ ] Automatically selecting an algorithm based on the specific rank requirement.
- [ ] Explore other low-rank representations or parameter-efficient methods to fine-tune either the entire model or specific parts of it.
- [ ] More experiments for different task, not only diffusion models.

Citation

```
@misc{LyCORIS,
  author       = "Shih-Ying Yeh (Kohaku-BlueLeaf), Yu-Guan Hsieh, Zhido
  title        = "LyCORIS - Lora beYond Conventional methods, Other Ran
  howpublished = "\url{https://github.com/KohakuBlueleaf/LyCORIS}",
  month        = "March",
  year         = "2023"
}
```