Monthly Report

2024/04/16

비샬

Steel Segmentation

Training of Only E-type images

Dataset Overview:

	6 nos
	10 nos
	8 nos
Total	24 nos

SEM Image: $\mathbb{R}^{1660\times1640\times1}$

Label : $\mathbb{R}^{1660\times1640\times3}$

Train

	4 nos
	8 nos
	5 nos
Total	17 nos

Validation

	1 nos
	1 nos
	2 nos
Total	4 nos

Test

	1 nos
	1 nos
	1 nos
Total	3 nos

Steel Segmentation

Training of Only E-type images

Train		Validation		Test	
	4 nos		1 nos		1 nos
	8 nos		1 nos		1 nos
	5 nos		2 nos		1 nos
Total	17 nos	Total	4 nos	Total	3 nos

Augmentations Performed → Sliding Augmentation, Flipping, Rotation (0, 90), Magnify (0, 2.5), Intensity (0-10), Gamma (0-10), Contrast (HE)

(Same Augmentations as done in previous experiments)

Train	4480 nos
Validation	1120 nos
Test	3 nos

SEM Image: $\mathbb{R}^{800\times800\times1}$

Label : $\mathbb{R}^{800\times800\times3}$

Steel Segmentation

Training of Only E-type images

Experiments

Models	Pixel Accuracy	Dice Score
Vanilla U-Net3+	77.31	75.58
Enhanced U-Net3+	82.09	79.43
ELU-Net	81.41	79.06
Enhanced ELU-Net	86.3	84.09

3.85%

Performance Increase

5.03%

Enhancements → 7 × 7 Kernels, Dilated Convolutions, Blur Pooling
Loss → Focal + Jaccard+ MS-SSIM

Steel Segmentation

Training of Only E-type images

Experiments

Models	Pixel Accuracy		Dice Score
Vanilla U-Net3+	77.31		75.58
Enhanced U-Net3+	82.09	↑ 3.85%	79.43
Enhanced* U-Net3+	81.34	↓ 0.75%	79.02
ELU-Net	81.41		79.06
Enhanced ELU-Net	86.3	↑ 5.03%	84.09
Enhanced* ELU-Net	84.11	↓ 2.19%	81.86

Why
?
(GLU performed well in nuclei segmentation but why is it struggling here?)

Enhancements* → 7 × 7 Kernels, Dilated Convolutions, Blur Pooling, GLU Activation
Loss → Focal + Jaccard+ MS-SSIM

Steel Segmentation

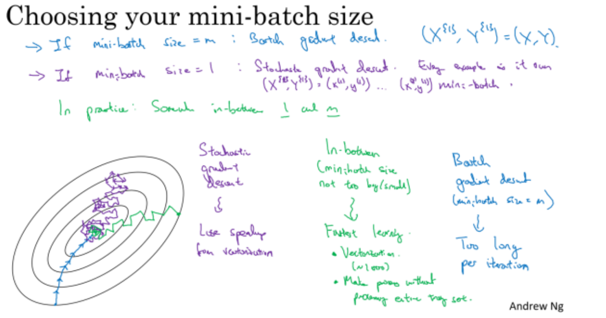
Training of Only E-type images

Why ?

Experiments

Models	Pixel Accuracy		Dice Score		
Vanilla U-Net3+	77.31		75.58	28GB, 21 Mins per epoch	Batch → 16
Enhanced U-Net3+	82.09	↑ 3.85%	79.43	44GB, 42 Mins per epoch	Batch → 16
Enhanced* U-Net3+	81.34	↓ 0.75%	79.02	47GB, 11 Hour per epoch	Batch → 2
ELU-Net	81.41		79.06		
Enhanced ELU-Net	86.3	↑ 5.03%	84.09		
Enhanced* ELU-Net	84.11	↓ 2.19%	81.86		

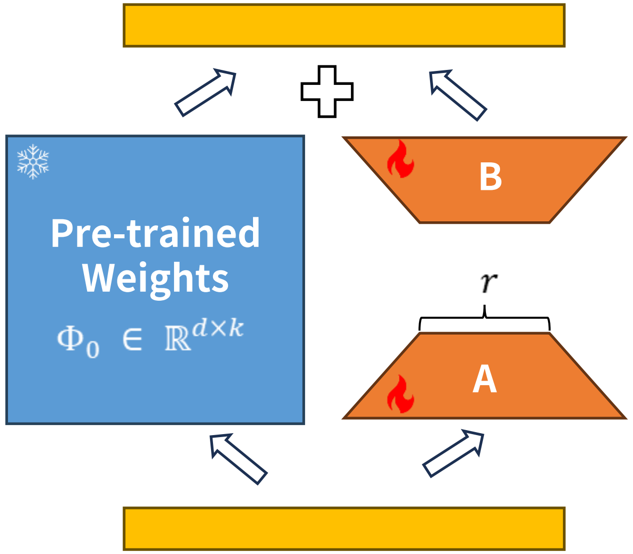
Noisy Gradient Estimates & Poor Generalization



Enhancements* → 7 × 7 Kernels, Dilated Convolutions, Blur Pooling, GLU Activation
Loss → Focal + Jaccard+ MS-SSIM

Steel Segmentation – LoRA

Training of Only E-type images



Steel Segmentation – LoRA

Training of Only E-type images

Data Loader

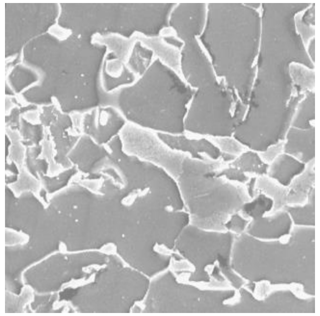
Load pre-trained SAM

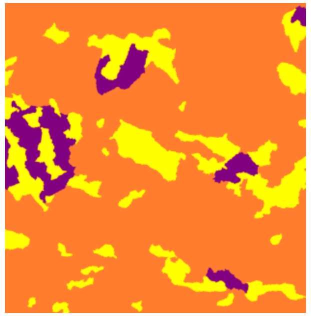
Initiate SAM_LoRA

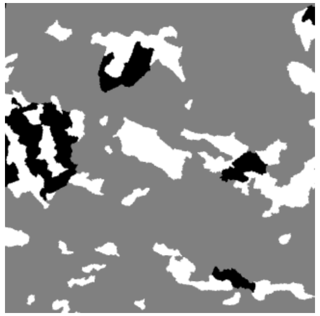
Process Input

Start Training

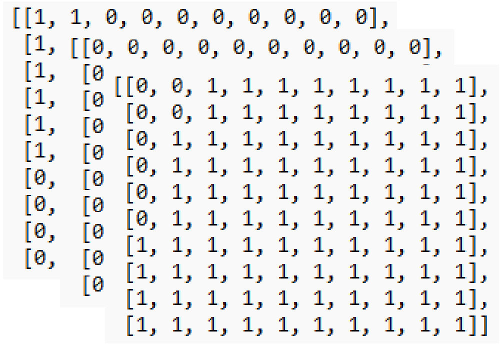
Previous Training







$\{0, 128, 255\}$



One-hot

Steel Segmentation – LoRA

Training of Only E-type images

Data Loader

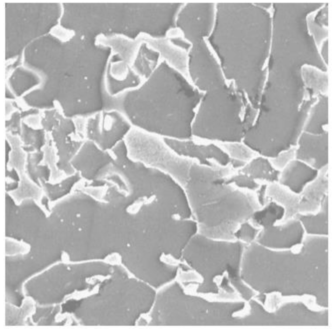
Load pre-trained SAM

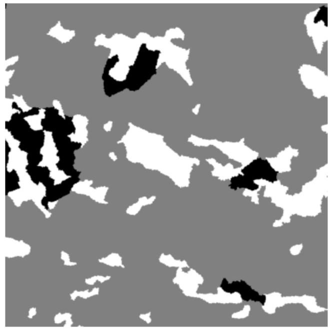
Initiate SAM_LoRA

Process Input

Start Training

Current Training





bainite ferrite martensite

opencv

Steel Segmentation – LoRA

Training of Only E-type images

Data Loader

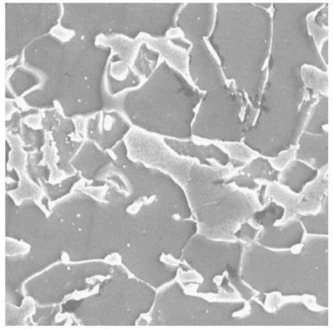
Load pre-trained SAM

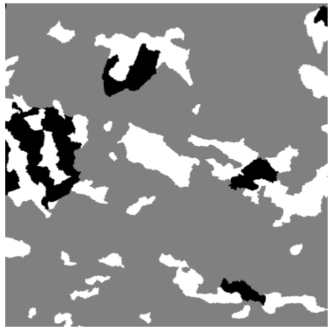
Initiate SAM_LoRA

Process Input

Start Training

Current Training





collated

The bounding boxes are all stored in a single list

Steel Segmentation – LoRA

Training of Only E-type images

Data Loader

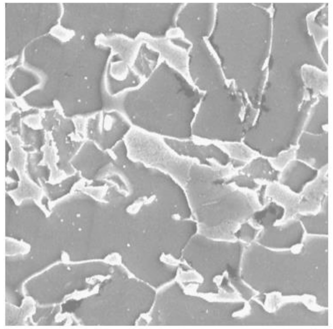
Load pre-trained SAM

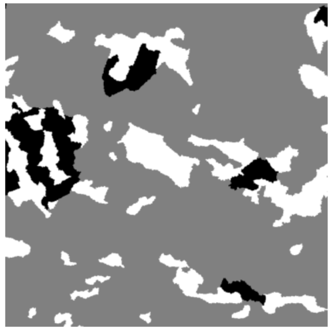
Initiate SAM_LoRA

Process Input

Start Training

Current Training





collated

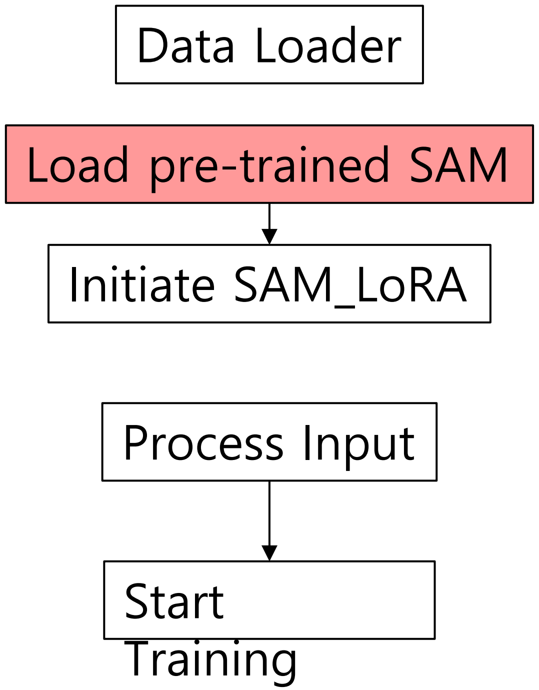
Used as Ground Truth Masks

The bounding boxes are all stored in a single list

Used as input prompt

Steel Segmentation – LoRA

Training of Only E-type images

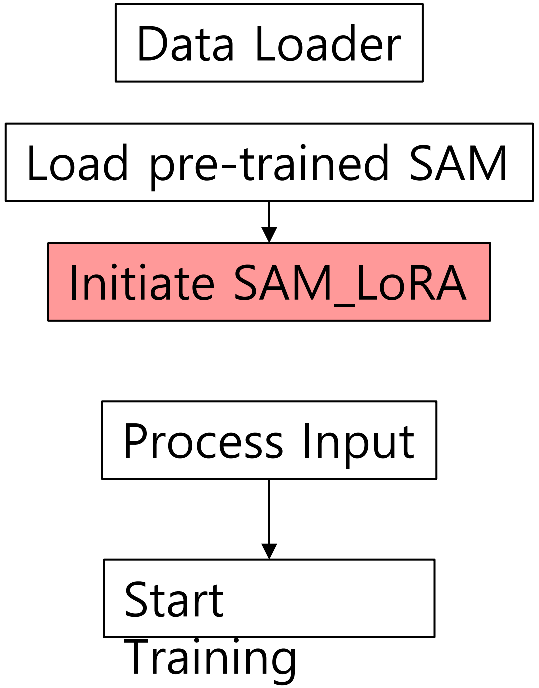


Used the base pre-trained SAM

- ViT B → 91M parameters (selected)
- ViT L → 308M parameters
- ViT H → 636M parameters

Steel Segmentation – LoRA

Training of Only E-type images



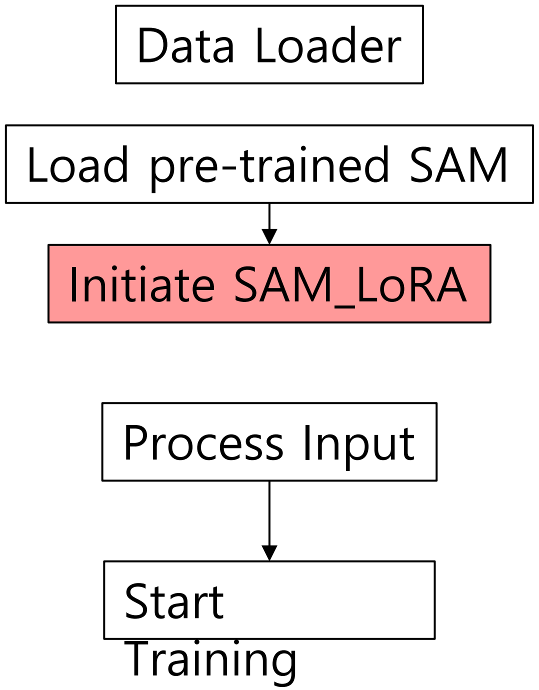
Made using Latex

Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                          ▷ SAM model instance
        rank                               ▷ Rank of the LoRA matrix
        lora_layer                         ▷ List of layers for LoRA
        A_weights, B_weights               ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v, w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

Steel Segmentation – LoRA

Training of Only E-type images

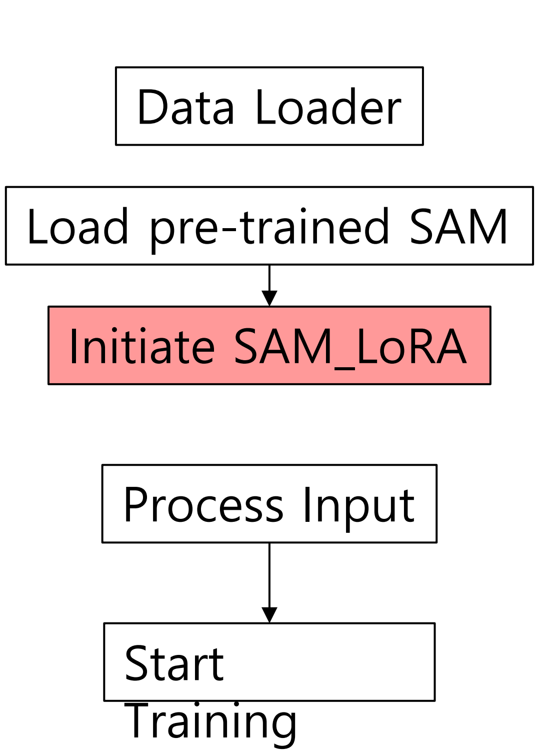


Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                              ▷ SAM model instance
        rank                                   ▷ Rank of the LoRA matrix
        lora_layer                             ▷ List of layers for LoRA
        A_weights, B_weights                   ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v,
w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

Steel Segmentation – LoRA

Training of Only E-type images



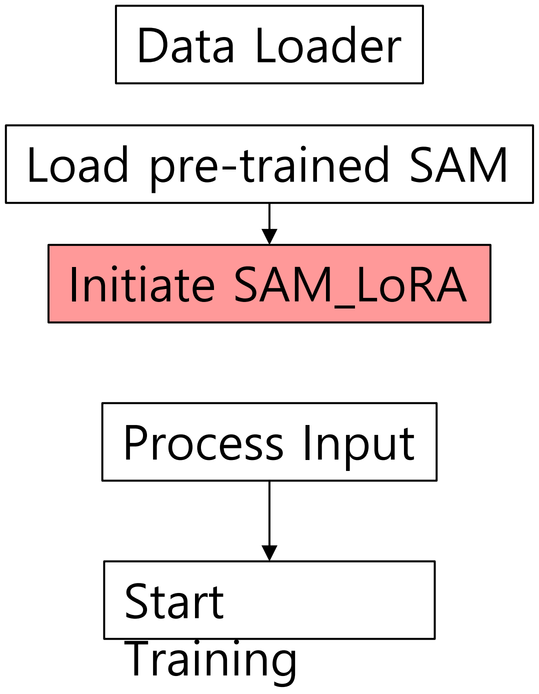
Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                                   ▷ SAM model instance
        rank                                        ▷ Rank of the LoRA matrix
        lora_layer                                  ▷ List of layers for LoRA
        A_weights, B_weights                        ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v, w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

If *rank* = 0, then its finetuning

Steel Segmentation – LoRA

Training of Only E-type images



Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

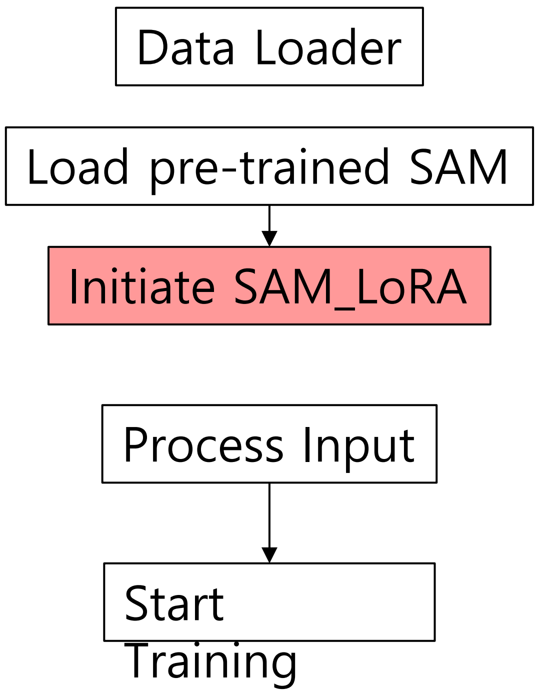
```
Class SAM_LoRA
    Properties:
        sam_model                                   ▷ SAM model instance
        rank                                        ▷ Rank of the LoRA matrix
        lora_layer                                  ▷ List of layers for LoRA
        A_weights, B_weights                        ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v,
w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

Aim is to add LoRA weights to the attention blocks

Each block of SAM has one attention block

Steel Segmentation – LoRA

Training of Only E-type images

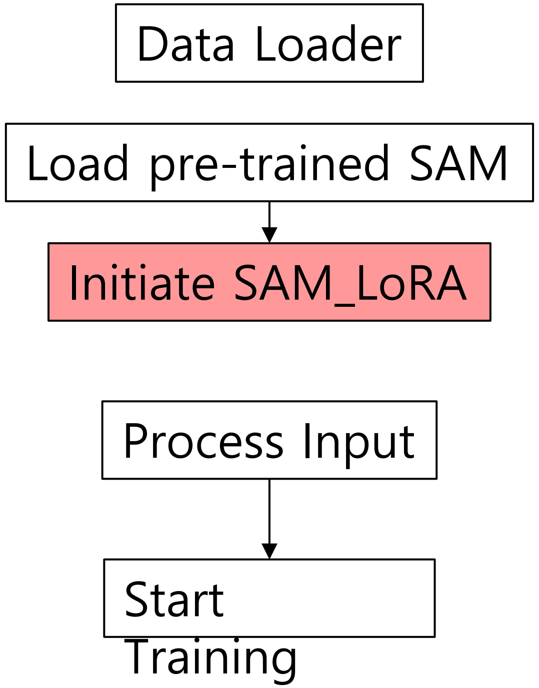


Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                                   ▷ SAM model instance
        rank                                        ▷ Rank of the LoRA matrix
        lora_layer                                  ▷ List of layers for LoRA
        A_weights, B_weights                        ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list                          Initialize low rank matrices
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v, w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

Steel Segmentation – LoRA

Training of Only E-type images

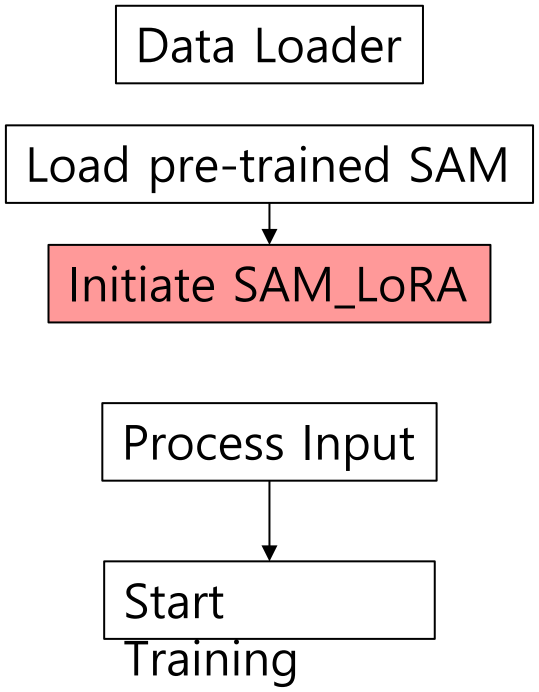


Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                                  ▷ SAM model instance
        rank                                       ▷ Rank of the LoRA matrix
        lora_layer                                 ▷ List of layers for LoRA
        A_weights, B_weights                       ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v, w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

Steel Segmentation – LoRA

Training of Only E-type images



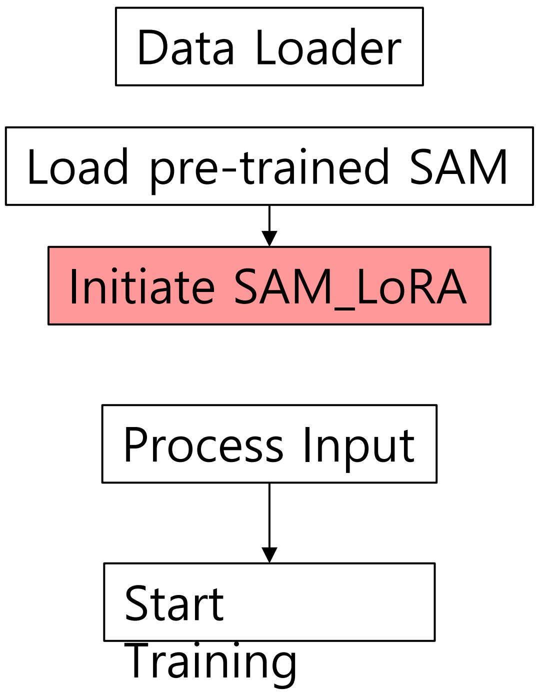
Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                          ▷ SAM model instance
        rank                               ▷ Rank of the LoRA matrix
        lora_layer                         ▷ List of layers for LoRA
        A_weights, B_weights               ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v,
w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

t_layer_i is the index of the current block,
blk is the block itself

Steel Segmentation – LoRA

Training of Only E-type images



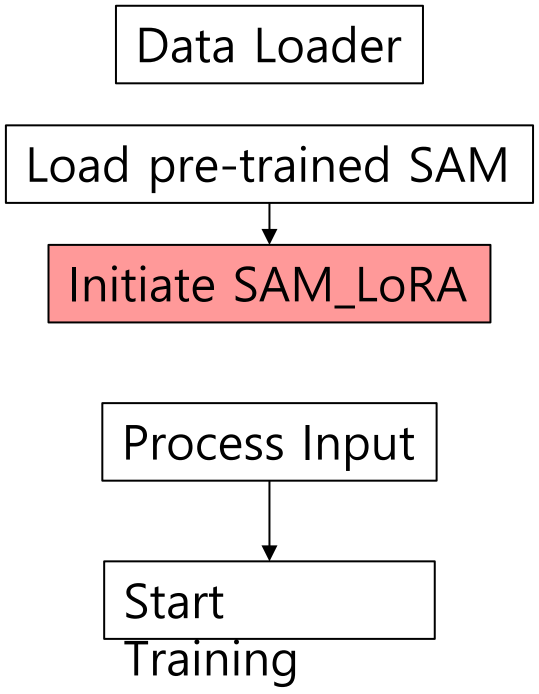
Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                                ▷ SAM model instance
        rank                                     ▷ Rank of the LoRA matrix
        lora_layer                               ▷ List of layers for LoRA
        A_weights, B_weights                     ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v, w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

Check if current block is in LoRA layer else skip

Steel Segmentation – LoRA

Training of Only E-type images



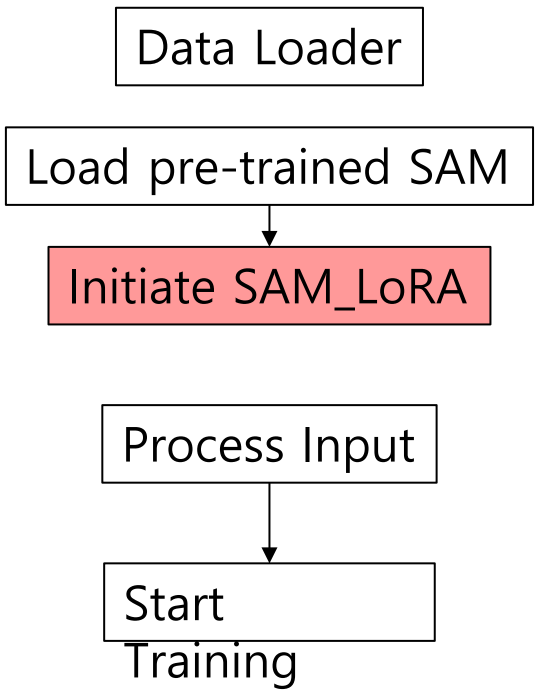
Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                                    ▷ SAM model instance
        rank                                         ▷ Rank of the LoRA matrix
        lora_layer                                   ▷ List of layers for LoRA
        A_weights, B_weights                         ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v, w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

Get the q-k-v values from SAM

Steel Segmentation – LoRA

Training of Only E-type images



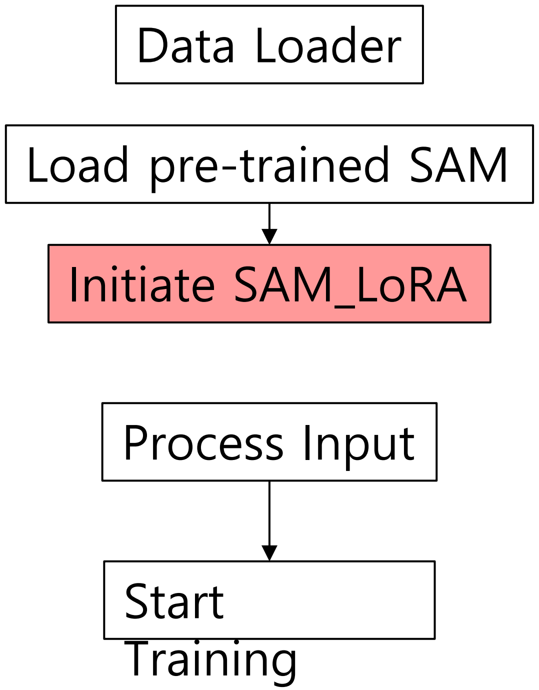
Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                                   ▷ SAM model instance
        rank                                        ▷ Rank of the LoRA matrix
        lora_layer                                  ▷ List of layers for LoRA
        A_weights, B_weights                        ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v, w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

Create Linear layers to train

Steel Segmentation – LoRA

Training of Only E-type images

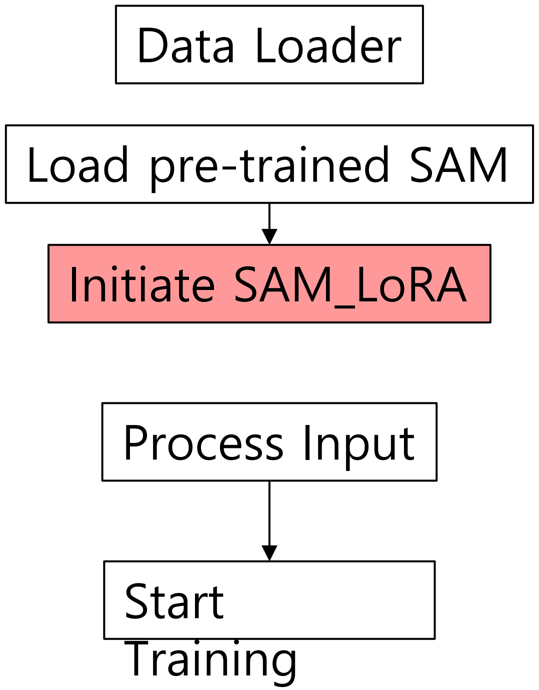


Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                                   ▷ SAM model instance
        rank                                        ▷ Rank of the LoRA matrix
        lora_layer                                  ▷ List of layers for LoRA
        A_weights, B_weights                        ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v, w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

Steel Segmentation – LoRA

Training of Only E-type images



Algorithm 1 LoRA adaptation for SAM (Segment Anything Model)

```
Class SAM_LoRA
    Properties:
        sam_model                                   ▷ SAM model instance
        rank                                        ▷ Rank of the LoRA matrix
        lora_layer                                  ▷ List of layers for LoRA
        A_weights, B_weights                        ▷ LoRA weights
procedure INITIALIZE(sam_model, rank, lora_layer = None)
    assert rank > 0
    if lora_layer = None then
        lora_layer ← range(len(sam_model.image_encoder.blocks))
    end if
    A_weights ← empty list
    B_weights ← empty list
    Freeze parameters in sam_model.image_encoder
    for t_layer_i, blk in enumerate(sam_model.image_encoder.blocks) do
        if t_layer_i not in lora_layer then
            continue
        end if
        w_qkv_linear ← blk.attn.qkv
        Create LoRA layers: w_a_linear_q, w_b_linear_q, w_a_linear_v, w_b_linear_v
        Append to A_weights and B_weights
        Replace blk.attn.qkv with a new LoRA_qkv instance
    end for
    Call reset_parameters
    self.sam ← sam_model
    self.lora_vit ← sam_model.image_encoder
end procedure
```

Replace updated weights to SAM

Steel Segmentation – LoRA

Training of Only E-type images

Data Loader

Load pre-trained SAM

Initiate SAM_LoRA

Process Input

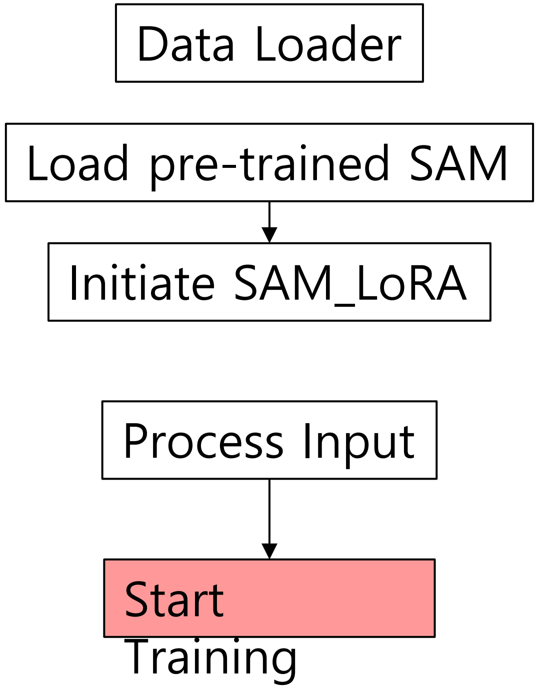
Start Training

- Image is preprocessed based on SAM input requirement
- Bounding Boxes are encoded using SAM’s prompt encoder

The inputs into the model – (Image, prompt, target) is processed into tensors

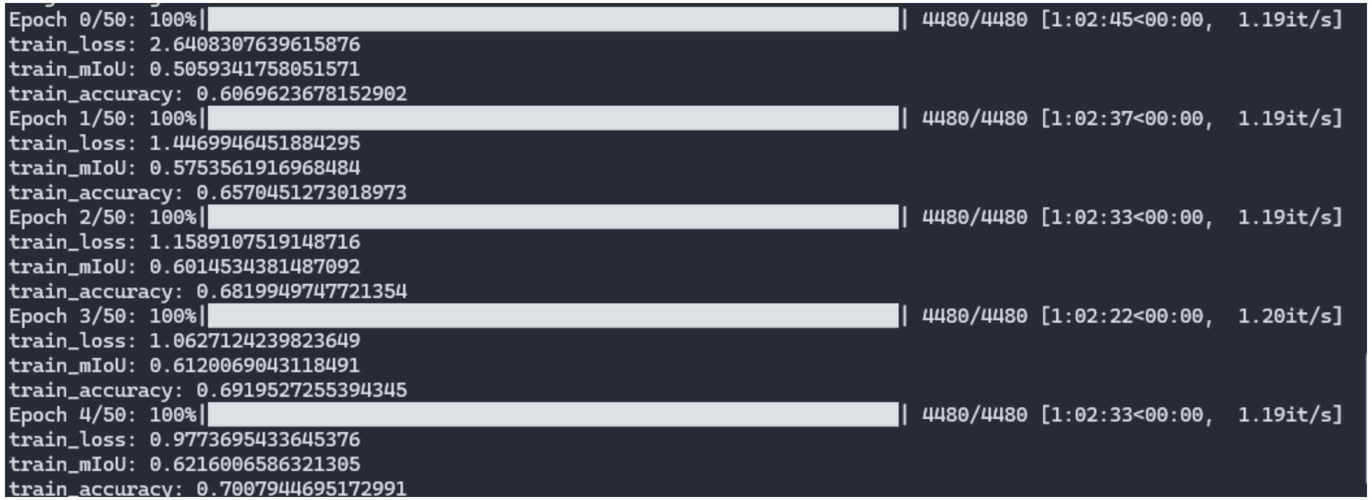
Steel Segmentation – LoRA

Training of Only E-type images



- Training is started with rank = 512 (I will test different rank values)

Initial training results



```
Epoch 0/50: 100%|████████████████████████████████| 4480/4480 [1:02:45<00:00,  1.19it/s]
train_loss: 2.6408307639615876
train_mIoU: 0.5059341758051571
train_accuracy: 0.6069623678152902
Epoch 1/50: 100%|████████████████████████████████| 4480/4480 [1:02:37<00:00,  1.19it/s]
train_loss: 1.4469946451884295
train_mIoU: 0.5753561916968484
train_accuracy: 0.6570451273018973
Epoch 2/50: 100%|████████████████████████████████| 4480/4480 [1:02:33<00:00,  1.19it/s]
train_loss: 1.1589107519148716
train_mIoU: 0.6014534381487092
train_accuracy: 0.6819949747721354
Epoch 3/50: 100%|████████████████████████████████| 4480/4480 [1:02:22<00:00,  1.20it/s]
train_loss: 1.0627124239823649
train_mIoU: 0.6120069043118491
train_accuracy: 0.6919527255394345
Epoch 4/50: 100%|████████████████████████████████| 4480/4480 [1:02:33<00:00,  1.19it/s]
train_loss: 0.9773695433645376
train_mIoU: 0.6216006586321305
train_accuracy: 0.7007944695172991
```