GANomaly

Contents

GANomaly: Semi-Supervised Anomaly Detection via Adversarial Training.

https://arxiv.org/abs/1805.06725

***class* anomalib.models.image.ganomaly.lightning_model.Ganomaly**(*batch_size=32*, *n_features=64*, *latent_vec_size=100*, *extra_layers=0*, *add_final_conv_layer=True*, *wadv=1*, *wcon=50*, *wenc=1*, *lr=0.0002*, *beta1=0.5*, *beta2=0.999*)

Bases: `AnomalyModule`

PL Lightning Module for the GANomaly Algorithm.

Parameters:

- **batch_size** (*int*) – Batch size. Defaults to `32`.
- **n_features** (*int*) – Number of features layers in the CNNs. Defaults to `64`.
- **latent_vec_size** (*int*) – Size of autoencoder latent vector. Defaults to `100`.
- **extra_layers** (*int, optional*) – Number of extra layers for encoder/decoder. Defaults to `0`.
- **add_final_conv_layer** (*bool, optional*) – Add convolution layer at the end. Defaults to `True`.
- **wadv** (*int, optional*) – Weight for adversarial loss. Defaults to `1`.
- **wcon** (*int, optional*) – Image regeneration weight. Defaults to `50`.
- **wenc** (*int, optional*) – Latent vector encoder weight. Defaults to `1`.
- **lr** (*float, optional*) – Learning rate. Defaults to `0.0002`.
- **beta1** (*float, optional*) – Adam beta1. Defaults to `0.5`.
- **beta2** (*float, optional*) – Adam beta2. Defaults to `0.999`.

`configure_optimizers()`

Configure optimizers for each decoder.

Returns:

Adam optimizer for each decoder

Return type:

Optimizer

`property learning_type: LearningType`

Return the learning type of the model.

Returns:

Learning type of the model.

Return type:

LearningType

`on_test_batch_end(outputs, batch, batch_idx, dataloader_idx=0)`

Normalize outputs based on min/max values.

Return type:

`None`

`on_test_start()`

Reset min max values before test batch starts.

Return type:

`None`

`on_validation_batch_end(outputs, batch, batch_idx, dataloader_idx=0)`

Normalize outputs based on min/max values.

Return type:

`None`

`on_validation_start()`

Reset min and max values for current validation epoch.

Return type:

`None`

`test_step(batch, batch_idx, *args, **kwargs)`

Update min and max scores from the current step.

Return type:

`Union[Tensor, Mapping[str, Any], None]`

`property trainer_arguments: dict[str, Any]`

Return GANomaly trainer arguments.

training_step(*batch*, *batch_idx*)

Perform the training step.

Parameters:

- **batch** (*dict[str, str | torch.Tensor]*) – Input batch containing images.
- **batch_idx** (*int*) – Batch index.
- **optimizer_idx** (*int*) – Optimizer which is being called for current training step.

Returns:

Loss

Return type:

STEP_OUTPUT

validation_step(*batch*, **args*, ***kwargs*)

Update min and max scores from the current step.

Parameters:

- **batch** (*dict[str, str | torch.Tensor]*) – Predicted difference between z and z_hat.
- **args** – Additional arguments.
- **kwargs** – Additional keyword arguments.

Returns:

Output predictions.

Return type:

(STEP_OUTPUT)

Torch models defining encoder, decoder, Generator and Discriminator.

Code adapted from samet-akcay/ganomaly.

class anomalib.models.image.ganomaly.torch_model.**GanomalyModel**(*input_size*, *num_input_channels*, *n_features*, *latent_vec_size*, *extra_layers=0*, *add_final_conv_layer=True*)

Bases: `Module`

Ganomaly Model.

Parameters:

- **input_size** (*tuple[int, int]*) – Input dimension.
- **num_input_channels** (*int*) – Number of input channels.
- **n_features** (*int*) – Number of features layers in the CNNs.
- **latent_vec_size** (*int*) – Size of autoencoder latent vector.
- **extra_layers** (*int, optional*) – Number of extra layers for encoder/decoder. Defaults to `0`.
- **add_final_conv_layer** (*bool, optional*) – Add convolution layer at the end. Defaults to `True`.

forward(*batch*)

Get scores for batch.

Parameters:

batch (*torch.Tensor*) – Images

Returns:

Regeneration scores.

Return type:

Tensor

static **weights_init**(*module*)

Initialize DCGAN weights.

Parameters:

module (*nn.Module*) – [description]

Return type:

`None`

Loss function for the GANomaly Model Implementation.

class `anomalib.models.image.ganomaly.loss.`**`DiscriminatorLoss`**

Bases: `Module`

Discriminator loss for the GANomaly model.

`forward`(*pred_real*, *pred_fake*)

Compute the loss for a predicted batch.

Parameters:

- **pred_real** (*torch.Tensor*) – Discriminator predictions for the real image.
- **pred_fake** (*torch.Tensor*) – Discriminator predictions for the fake image.

Returns:

The computed discriminator loss.

Return type:

Tensor

class `anomalib.models.image.ganomaly.loss.`**`GeneratorLoss`**(*wadv=1*, *wcon=50*, *wenc=1*)

Bases: `Module`

Generator loss for the GANomaly model.

Parameters:

- **wadv** (*int, optional*) – Weight for adversarial loss. Defaults to `1`.
- **wcon** (*int, optional*) – Image regeneration weight. Defaults to `50`.
- **wenc** (*int, optional*) – Latent vector encoder weight. Defaults to `1`.

`forward`(*latent_i*, *latent_o*, *images*, *fake*, *pred_real*, *pred_fake*)

Compute the loss for a batch.

Parameters:

- **latent_i** (*torch.Tensor*) – Latent features of the first encoder.
- **latent_o** (*torch.Tensor*) – Latent features of the second encoder.
- **images** (*torch.Tensor*) – Real image that served as input of the generator.
- **fake** (*torch.Tensor*) – Generated image.
- **pred_real** (*torch.Tensor*) – Discriminator predictions for the real image.
- **pred_fake** (*torch.Tensor*) – Discriminator predictions for the fake image.

Returns:

The computed generator loss.

Return type:

Tensor

Previous
FastFlow

Next
Padim