Engine

Contents

Anomalib engine.

class **anomalib.engine.Engine**(***callbacks***=*None*, ***normalization***=*NormalizationMethod.MIN_MAX*, ***threshold***=*'F1AdaptiveThreshold'*, ***task***=*TaskType.SEGMENTATION*, ***image_metrics***=*None*, ***pixel_metrics***=*None*, ***logger***=*None*, ***default_root_dir***=*'results'*, ****kwargs***)

Bases: `object`

Anomalib Engine.

> **Note**
>
> Refer to PyTorch Lightning's Trainer for a list of parameters for details on other Trainer parameters.

Parameters:

- **callbacks** (*list[Callback]*) – Add a callback or list of callbacks.
- **normalization** (*NORMALIZATION, optional*) – Normalization method. Defaults to NormalizationMethod.MIN_MAX.
- **threshold** (*THRESHOLD*) – Thresholding method. Defaults to "F1AdaptiveThreshold".
- **task** (*TaskType, optional*) – Task type. Defaults to TaskType.SEGMENTATION.
- **image_metrics** (*list[str] | str | dict[str, dict[str, Any]] | None, optional*) – Image metrics to be used for evaluation. Defaults to None.
- **pixel_metrics** (*list[str] | str | dict[str, dict[str, Any]] | None, optional*) – Pixel metrics to be used for evaluation. Defaults to None.
- **default_root_dir** (*str, optional*) – Default root directory for the trainer. The results will be saved in this directory. Defaults to `results`.
- ****kwargs** – PyTorch Lightning Trainer arguments.

export(*model*, *export_type*, *export_root=None*, *transform=None*, *ov_args=None*, *ckpt_path=None*)

Export the model in PyTorch, ONNX or OpenVINO format.

Parameters:

- **model** (*AnomalyModule*) – Trained model.
- **export_type** (*ExportType*) – Export type.
- **export_root** (*str | Path | None, optional*) – Path to the output directory. If it is not set, the model is exported to trainer.default_root_dir. Defaults to None.
- **transform** (*Transform | None, optional*) – Input transform to include in the exported model. If not provided, the engine will try to use the transform from the datamodule or dataset. Defaults to None.
- **ov_args** (*dict[str, Any] | None, optional*) – This is optional and used only for OpenVINO's model optimizer. Defaults to None.
- **ckpt_path** (*str | Path | None*) – Checkpoint path. If provided, the model will be loaded from this path.

Returns:

Path to the exported model.

Return type:

Path

Raises:

- **ValueError** – If Dataset, Datamodule, and transform are not provided.
- **TypeError** – If path to the transform file is not a string or Path.

CLI Usage:

1. **To export as a torch `.pt` file you can run the following command.**

   ```
   `python anomalib export --model Padim --export_mode TORCH
   --data MVTec `
   ```

2. **To export as an ONNX `.onnx` file you can run the following command.**

   ```
   `python anomalib export --model Padim --export_mode ONNX --data
   Visa --input_size "[256,256]" `
   ```

3. **To export as an OpenVINO `.xml` and `.bin` file you can run the following command.**

   ```
   `python anomalib export --model Padim --export_mode OPENVINO
   --data Visa --input_size "[256,256]" `
   ```

4. **You can also overrride OpenVINO model optimizer by adding the `--mo_args.<key>` arguments.**

   ```
   `python anomalib export --model Padim --export_mode OPENVINO
   --data Visa --input_size "[256,256]"
   --mo_args.compress_to_fp16 False `
   ```

***fit*(*model*, *train_dataloaders=None*, *val_dataloaders=None*, *datamodule=None*, *ckpt_path=None*)**

Fit the model using the trainer.

Parameters:

- **model** (*AnomalyModule*) – Model to be trained.
- **train_dataloaders** (*TRAIN_DATALOADERS | None, optional*) – Train dataloaders. Defaults to None.
- **val_dataloaders** (*EVAL_DATALOADERS | None, optional*) – Validation dataloaders. Defaults to None.
- **datamodule** (*AnomalibDataModule | None, optional*) – Lightning datamodule. If provided, dataloaders will be instantiated from this. Defaults to None.
- **ckpt_path** (*str | None, optional*) – Checkpoint path. If provided, the model will be loaded from this path. Defaults to None.

Return type:

`None`

CLI Usage:

1. **you can pick a model, and you can run through the MVTec dataset.**

   `` `python anomalib fit --model anomalib.models.Padim ` ``

2. **Of course, you can override the various values with commands.**

   `` `python anomalib fit --model anomalib.models.Padim --data <CONFIG | CLASS_PATH_OR_NAME> --trainer.max_epochs 3 ` ``

4. **If you have a ready configuration file, run it like this.**

   `` `python anomalib fit --config <config_file_path> ` ``

property* model: *AnomalyModule

Property to get the model.

Raises:

UnassignedError – When the model is not assigned yet.

Returns:

Anomaly model.

Return type:

AnomalyModule

property **normalization_callback**: *NormalizationCallback | None*

The `NormalizationCallback` callback in the trainer.callbacks list, or `None` if it doesn't exist.

Returns:

Normalization callback, if available.

Return type:

NormalizationCallback | None

Raises:

ValueError – If there are multiple normalization callbacks.

predict(*model=None*, *dataloaders=None*, *datamodule=None*, *dataset=None*, *return_predictions=None*, *ckpt_path=None*)

Predict using the model using the trainer.

Sets up the trainer and the dataset task if not already set up. Then validates the model if needed and a validation dataloader is available. Finally, predicts using the model.

Parameters:

- **model** (*AnomalyModule* | *None, optional*) – Model to be used for prediction. Defaults to None.
- **dataloaders** (*EVAL_DATALOADERS* | *None, optional*) – An iterable or collection of iterables specifying predict samples. Defaults to None.
- **datamodule** (*AnomalibDataModule* | *None, optional*) – A `AnomalibDataModule` that defines the `predict_dataloader` hook. The datamodule can also be a dataset that will be wrapped in a torch Dataloader. Defaults to None.
- **dataset** (*Dataset* | *PredictDataset* | *None, optional*) – A `Dataset` or `PredictDataset` that will be used to create a dataloader. Defaults to None.
- **return_predictions** (*bool* | *None, optional*) – Whether to return predictions. `True` by default except when an accelerator that spawns processes is used (not supported). Defaults to None.
- **ckpt_path** (*str* | *None, optional*) – Either `"best"`, `"last"`, `"hpc"` or path to the checkpoint you wish to predict. If `None` and the model instance was passed, use the current weights. Otherwise, the best model checkpoint from the previous `trainer.fit` call will be loaded if a checkpoint callback is configured. Defaults to None.

Returns:

Predictions.

Return type:

_PREDICT_OUTPUT | None

CLI Usage:

1. **you can pick a model.**

   `` `python anomalib predict --model anomalib.models.Padim anomalib predict --model Padim --data datasets/MVTec/bottle/test/broken_large ` ``

2. **Of course, you can override the various values with commands.**

   `` `python anomalib predict --model anomalib.models.Padim --data <CONFIG | CLASS_PATH_OR_NAME> ` ``

4. **If you have a ready configuration file, run it like this.**

```
`python anomalib predict --config <config_file_path>
--return_predictions `
```

5. **You can also point to a folder with image or a single image instead of passing a dataset.**

```
`python anomalib predict --model Padim --data
<PATH_TO_IMAGE_OR_FOLDER> --ckpt_path <PATH_TO_CHECKPOINT> `
```

`test`(*model=None*, *dataloaders=None*, *ckpt_path=None*, *verbose=True*, *datamodule=None*)

Test the model using the trainer.

Sets up the trainer and the dataset task if not already set up. Then validates the model if needed and finally tests the model.

Parameters:

- **model** (*AnomalyModule* | *None, optional*) – The model to be tested. Defaults to None.
- **dataloaders** (*EVAL_DATALOADERS* | *None, optional*) – An iterable or collection of iterables specifying test samples. Defaults to None.
- **ckpt_path** (*str* | *None, optional*) – Either `"best"`, `"last"`, `"hpc"` or path to the checkpoint you wish to test. If `None` and the model instance was passed, use the current weights. Otherwise, the best model checkpoint from the previous `trainer.fit` call will be loaded if a checkpoint callback is configured. Defaults to None.
- **verbose** (*bool, optional*) – If True, prints the test results. Defaults to True.
- **datamodule** (*AnomalibDataModule* | *None, optional*) – A `AnomalibDataModule` that defines the `test_dataloader` hook. Defaults to None.

Returns:

A List of dictionaries containing the test results. 1 dict per dataloader.

Return type:

_EVALUATE_OUTPUT

Examples

fit and test a one-class model >>> from anomalib.data import MVTec >>> from anomalib.models import Padim >>> from anomalib.engine import Engine

```
>>> datamodule = MVTec()
>>> model = Padim()
>>> model.learning_type
<LearningType.ONE_CLASS: 'one_class'>
```

```
>>> engine = Engine()
>>> engine.fit(model, datamodule=datamodule)
>>> engine.test(model, datamodule=datamodule)
```

Test a zero-shot model >>> from anomalib.data import MVTec >>> from anomalib.models import Padim >>> from anomalib.engine import Engine

```
>>> datamodule = MVTec(image_size=240, normalization="clip")
>>> model = Padim()
>>> model.learning_type
<LearningType.ZERO_SHOT: 'zero_shot'>
```

```
>>> engine = Engine()
>>> engine.test(model, datamodule=datamodule)
```

CLI Usage:

1. **you can pick a model.**

   `python anomalib test --model anomalib.models.Padim `

2. **Of course, you can override the various values with commands.**

   `python anomalib test --model anomalib.models.Padim --data <CONFIG | CLASS_PATH_OR_NAME> `

4. **If you have a ready configuration file, run it like this.**

   `python anomalib test --config <config_file_path> `

property **threshold_callback**: *_ThresholdCallback | None*

The `ThresholdCallback` callback in the trainer.callbacks list, or `None` if it doesn't exist.

Returns:

Threshold callback, if available.

Return type:

_ThresholdCallback | None

Raises:

ValueError – If there are multiple threshold callbacks.

`train(model, train_dataloaders=None, val_dataloaders=None, test_dataloaders=None, datamodule=None, ckpt_path=None)`

Fits the model and then calls test on it.

Parameters:

- **model** (*AnomalyModule*) – Model to be trained.
- **train_dataloaders** (*TRAIN_DATALOADERS | None, optional*) – Train dataloaders. Defaults to None.
- **val_dataloaders** (*EVAL_DATALOADERS | None, optional*) – Validation dataloaders. Defaults to None.
- **test_dataloaders** (*EVAL_DATALOADERS | None, optional*) – Test dataloaders. Defaults to None.
- **datamodule** (*AnomalibDataModule | None, optional*) – Lightning datamodule. If provided, dataloaders will be instantiated from this. Defaults to None.
- **ckpt_path** (*str | None, optional*) – Checkpoint path. If provided, the model will be loaded from this path. Defaults to None.

Return type:

`List[Mapping[str, float]]`

CLI Usage:

1. **you can pick a model, and you can run through the MVTec dataset.**

   ```
   `python anomalib train --model anomalib.models.Padim --data MVTec `
   ```

2. **Of course, you can override the various values with commands.**

   `python anomalib train --model anomalib.models.Padim --data <CONFIG | CLASS_PATH_OR_NAME> --trainer.max_epochs 3 `

4. **If you have a ready configuration file, run it like this.**

   `python anomalib train --config <config_file_path> `

property **trainer**: *Trainer*

Property to get the trainer.

Raises:

UnassignedError – When the trainer is not assigned yet.

Returns:

Lightning Trainer.

Return type:

Trainer

validate(*model=None*, *dataloaders=None*, *ckpt_path=None*, *verbose=True*, *datamodule=None*)

Validate the model using the trainer.

Parameters:

- **model** (*AnomalyModule* | *None, optional*) – Model to be validated. Defaults to None.
- **dataloaders** (*EVAL_DATALOADERS* | *None, optional*) – Dataloaders to be used for validation. Defaults to None.
- **ckpt_path** (*str* | *None, optional*) – Checkpoint path. If provided, the model will be loaded from this path. Defaults to None.
- **verbose** (*bool, optional*) – Boolean to print the validation results. Defaults to True.
- **datamodule** (*AnomalibDataModule* | *None, optional*) – A `datamodule` `AnomalibDataModule` that defines the `val_dataloader` hook. Defaults to None.

Returns:

Validation results.

Return type:

_EVALUATE_OUTPUT | None

CLI Usage:

1. **you can pick a model.**

   `python anomalib validate --model anomalib.models.Padim `

2. **Of course, you can override the various values with commands.**

   `python anomalib validate --model anomalib.models.Padim --data <CONFIG | CLASS_PATH_OR_NAME> `

4. **If you have a ready configuration file, run it like this.**

   `python anomalib validate --config <config_file_path> `

Previous
AI VAD

Next
Metrics