Reviewed by Jackson Lightfoot

"""Recon objects for combining system blocks (such as datasets and transformers), model blocks (such as CNNs and ResNets), and optimization blocks (such as conjugate gradient descent)."""

**Classes:**

**Recon**

"""An abstract class for implementing system-model-optimization (SMO) constructions.

The Recon is an abstract class which outlines common functionality for all SMO structure implementations. All of them share hyperparameter initialization, MCMRI dataset processing and loading, loss function, training step, and optimizer code. Each implementation of Recon must provide batch, forward, and get\_metadata methods in order to define how batches are created from the data, how the model performs its forward pass, and what metadata the user should be able to return. Currently, Recon automatically builds the dataset as an MultiChannelMRIDataset object; overload \_build\_data to circumvent this.

Args:

hprams (dict): Key-value pairings with parameter names as keys.

Attributes:

hprams (dict): Key-value pairings with hyperparameter names as keys.

\_loss\_fun (func): Set to use either torch.nn.MSELoss or \_abs\_loss\_fun.

D (MultiChannelMRIDataset): Holds the MCMRI dataset.

"""

**Methods:**

***batch***

"""Not implemented, should define a forward operator A and the adjoint matrix of the input x.

Args:

data (Tensor): The data which the batch will be drawn from.

Raises:

NotImplementedError: Method needs to be implemented.

"""

***forward***

"""Not implemented, should perform a prediction using the implemented model.

Args:

y (Tensor): The data which will be passed to the model for processing.

Returns:

The model’s prediction in Tensor form.

Raises:

NotImplementedError: Method needs to be implemented.

"""

***get\_metadata***

"""Accesses metadata for the Recon.

Returns:

A dict holding the Recon’s metadata.

Raises:

NotImplementedError: Method needs to be implemented.

"""

***training\_step***

"""Defines a training step solving deep inverse problems, including batching, performing a forward pass through the model, and logging data. This may either be supervised or unsupervised based on hyperparameters.

Args:

batch (tuple): Should hold the indices of data and the corresponding data, in said order.

batch\_nb (None): Currently unimplemented.

Returns:

A dict holding performance data and current epoch for performance tracking over time.

"""

***configure\_optimizers***

"""Determines whether to use Adam or SGD depending on hyperparameters.

Returns:

Torch’s implementation of SGD or Adam, depending on hyperparameters.

"""

***train\_dataloader***

"""Creates a DataLoader object, with distributed training if specified in the hyperparameters.

Returns:

A PyTorch DataLoader that has been configured according to the hyperparameters.

"""