

Training a network

Function Signature

```
def train_model(  
    train_inputs: np.ndarray,  
    train_targets: np.ndarray,  
    test_inputs: np.ndarray,  
    test_targets: np.ndarray,  
    batch_size: int = 36,  
    epochs: int = 500,  
    learning_rate: float = 0.1,  
    momentum: float = 0.6,  
    early_stop: bool = False,  
    verbose: bool = True  
) -> None:
```

Parameters

- `train_inputs` (np.ndarray): The input data to use for training. This should be a numpy array of shape `(num_samples, input_shape)`.
- `train_targets` (np.ndarray): The target data to use for training. This should be a numpy array of shape `(num_samples, output_shape)`.
- `test_inputs` (np.ndarray): The input data to use for testing. This should be a numpy array of shape `(num_samples, input_shape)`.
- `test_targets` (np.ndarray): The target data to use for testing. This should be a numpy array of shape `(num_samples, output_shape)`.
- `batch_size` (int, default=36): The batch size to use during training.
- `epochs` (int, default=500): The number of epochs to train the model for.
- `learning_rate` (float, default=0.1): The learning rate to use during training.
- `momentum` (float, default=0.6): The momentum to use during training.
- `early_stop` (bool, default=False): Whether to use early stopping during training. If True, the training will stop when the validation loss stops improving.
- `verbose` (bool, default=True): Whether to print training progress during training.

Return Value

This function does not return anything. It trains the `deeprai.models.FeedForward` instance on the given data and saves the updated weights.

The `train_model` function trains the `deeprai.models.FeedForward` instance on the given training data using the specified hyperparameters. It also evaluates the model on the test data after each epoch and prints the training progress if `verbose=True`.

The `batch_size` parameter specifies the batch size to use during training. The `epochs` parameter specifies the number of epochs to train the model for. The `learning_rate` and `momentum` parameters specify the learning rate and momentum to use during training, respectively.

The `early_stop` parameter specifies whether to use early stopping during training. If `early_stop=True`, the training will stop when the validation loss stops improving.

Examples

Here's an example of how to use the `train_model` function:

```
from deeprai.models import FeedForward

model = FeedForward()
model.add_dense(784)
model.add_dense(128, activation='relu')
model.add_dense(64, activation='relu')
model.add_dense(10, activation='sigmoid')

train_inputs = ...
train_targets = ...
test_inputs = ...
test_targets = ...

model.train_model(
    train_inputs=train_inputs,
    train_targets=train_targets,
    test_inputs=test_inputs,
    test_targets=test_targets,
```

```
batch_size=32,  
epochs=1000,  
learning_rate=0.1,  
momentum=0.6,  
early_stop=True,  
verbose=True  
)
```

This code creates a `FeedForward` model with an input shape of `(784,)`, adds three dense layers with ReLU and softmax activation functions, sets

Revision #1

Created 7 April 2023 07:29:55 by Kieran Carter

Updated 7 April 2023 07:37:12 by Kieran Carter