

We then use Matplotlib to plot the loss function for the training and validation set.

Listing 6.11 Plotting the classification loss

```

import matplotlib.pyplot as plt

def plot_values(
    epochs_seen, examples_seen, train_values, val_values,
    label="loss"):
    fig, ax1 = plt.subplots(figsize=(5, 3))

    ax1.plot(epochs_seen, train_values, label=f"Training {label}")
    ax1.plot(
        epochs_seen, val_values, linestyle="--",
        label=f"Validation {label}"
    )
    ax1.set_xlabel("Epochs")
    ax1.set_ylabel(label.capitalize())
    ax1.legend()

    ax2 = ax1.twiny()
    ax2.plot(examples_seen, train_values, alpha=0)
    ax2.set_xlabel("Examples seen")

    fig.tight_layout()
    plt.savefig(f"{label}-plot.pdf")
    plt.show()

epochs_tensor = torch.linspace(0, num_epochs, len(train_losses))
examples_seen_tensor = torch.linspace(0, examples_seen, len(train_losses))

plot_values(epochs_tensor, examples_seen_tensor, train_losses, val_losses)

```

Figure 6.16 plots the resulting loss curves.

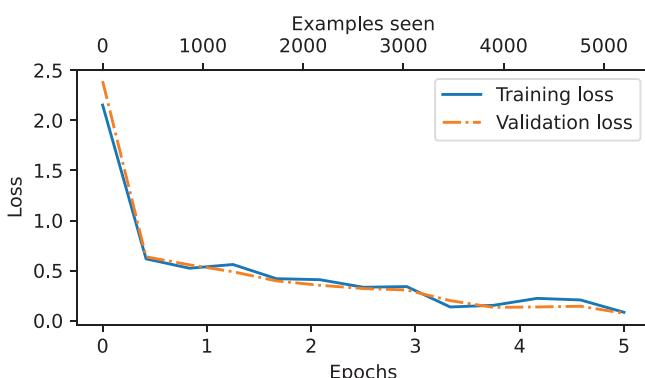


Figure 6.16 The model’s training and validation loss over the five training epochs. Both the training loss, represented by the solid line, and the validation loss, represented by the dashed line, sharply decline in the first epoch and gradually stabilize toward the fifth epoch. This pattern indicates good learning progress and suggests that the model learned from the training data while generalizing well to the unseen validation data.