1. Problem specification:

This is a simple Chatbot project by using Neural network and NLP. The chatbot model is trying to recognize the intents of sentences from users, and then make static responds(randomly choose an appropriately static response). Using data-driven machine learning can effectively help the model to predict the intents from users and make correct responses.

2. Model specification:

I use CNN model in Pytorch. Input are my model and training dataset, and the output are the trained model with its weights. The loss function I use at here is: CrossEntropy

3. Specify features and pre-processing:

Since the original data are a bunch of text, I need to transfer these text to numerical values for implementing machine learning. So firstly, I use nltk library to split sentences to individual words, and then convert words to numerical values for machine learning training dataset. In the end, I shuffle the dataset before I start to use it.

4. Specify data split strategy:

I split the original dataset as 80% for training, 20% for testing by using sklearn model_selection to randomly split dataset.

5. Specify the hyper parameter search space:

For the CNN model, I manually tried all odd values of hidden_size from [1, 3, … , 9].
And initial values of learning rate from [0.001, 0.002, 0.003, …, 0.01]

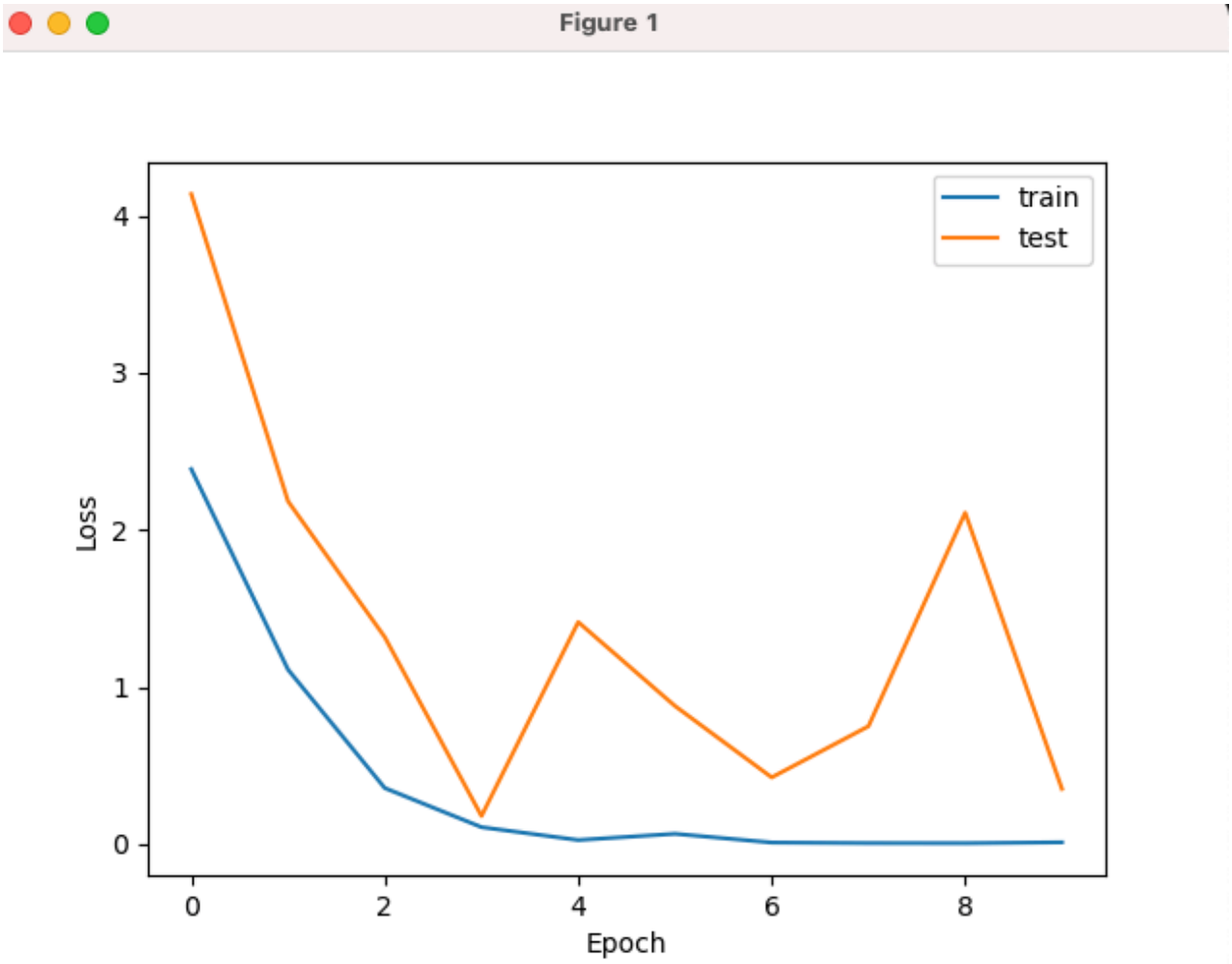
6. Explain how hyperparameters were optimized:

I use Pytorch optim.Adam function to optimize the learning rate during the training process.
And tried all values for hidden layout size, the model with less loss after training was selected.

7. Evaluate model on clean test set:

On the clean test dataset, my model has 0.3511 loss, which is worse than the performance of training set.

```
(base) tianhuizhou@Bobbys-MBP chatbot-python % python training.py
**********
epoch 20/200, loss_trainingSet = 2.3869, loss_testSet = 4.1387
epoch 40/200, loss_trainingSet = 1.1104, loss_testSet = 2.1826
epoch 60/200, loss_trainingSet = 0.3563, loss_testSet = 1.3177
epoch 80/200, loss_trainingSet = 0.1071, loss_testSet = 0.1784
epoch 100/200, loss_trainingSet = 0.0252, loss_testSet = 1.4131
epoch 120/200, loss_trainingSet = 0.0649, loss_testSet = 0.8789
epoch 140/200, loss_trainingSet = 0.0093, loss_testSet = 0.4240
epoch 160/200, loss_trainingSet = 0.0058, loss_testSet = 0.7497
epoch 180/200, loss_trainingSet = 0.0052, loss_testSet = 2.1094
epoch 200/200, loss_trainingSet = 0.0107, loss_testSet = 0.3511
final loss: loss_trainingSet = 0.0107, loss_testingSet = 0.3511
training complete. file saved to data.pth
```



8. Explain any differences in the train/test datasets:
I expect the model is able to predict the intents of sentences(from users). I think the reason that the performance might decrease is the original dataset is too small. There is a possibility that when I split data to training and test datasets, the intents do not distribute fairly. For example, I split a whole intent to test set, but the training set does not contain anything about this part. In this case, the model can have bad performance on test dataset(have a higher loss!).