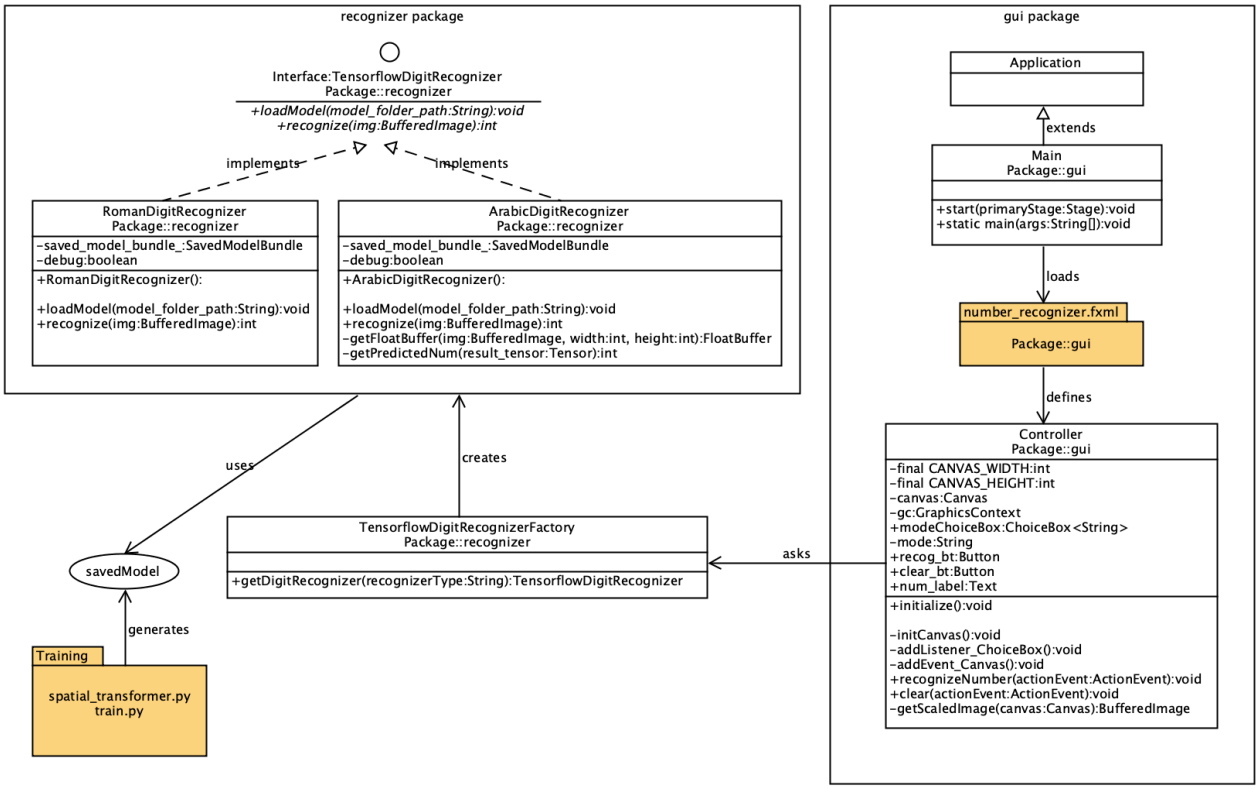
Summary

Project name： Number Recognizer
Project member: Wenwen Ni, Zhenghui Li
Member responsibility:
Wenwen Ni:
- Train the model
- Create the canvas and add EventHandlers
- Fetch the Input and output
- Create the interface and factory class
- Write comments and project description

Zhenghui Li
- Create the GUI by javafx and FXML
- Complete the comments
- Complete the factory patterns
- Test and write documentation(including class diagram, output txt, test screenshot, project description)

Class design:



Details

Training

There are two files related to training.

1. ***train.py***, the script to do the actual training.
2. ***spatial_transformer.py***, the utility function for spatial transformation layer in the neural network. It is called by ***train.py***.

spatial_transformer.py is copied from official Github page of tensorflow. https://github.com/tensorflow/models/blob/master/research/transformer/spatial_transformer.py

train.py

Input: official MNIST dataset (stored under ./MNIST_data/ directory).
Each image in the training data is 28 * 28. It uses a neural network with six layers (See the comments in train.py). We use mini batches for training and have the batch size as 50, that means each training step will use 50 images to train and update the parameters in the network.

Output: the model with parameters of the neural network (stored under **'./model_TRAINING_BATCH_NUM_batches_with_size_TRAINING_BATCH_SIZE/**, for example, if you use 10000 as the total number of training batches and 50 as batch size, the output will be under ./model_10000_batches_with_size_50/)

Application

We read the input image from canvas whose size is set to 400 * 400, convert it to a 28 * 28 image and get a BufferedImage object. Then we create a recognizer and load the saved model from our training.
In **ArabicDigitRecognizer.java**, we pass the image data and an array of keep probabilities to the model, run through the model and get the output. The output is an array of size 10, each number corresponds to the predicted confidence for each digit (from 0 - 9). The largest one will be the predicted output and the index will be the predicted number.

NumberRecognizer Interface

We create an interface **TensorflowDigitRecognizer** for instantiate recognizer in different mode(for example, Arabic number or Roman number). This interface requires to override two methods: ***loadModel()*** and ***recognize()***.
Then we use the **TensorflowDigitRecognizerFactory** class to generate different recognizers corresponding to the parameter. This is a factory design pattern.
We also wrap up the recognizing process into DigitRecognizer classes to implement the concept of encapsulation, and the Main class extends Application, which implements the concept of inheritance.