北京邮电大学 本科毕业设计（论文）中期进度报告

Project Mid-term Progress Report

学院 School	International School	**专业 Programme**	**Internet of Things Engineering**		
姓 Family name	Wang	**名 First Name**	Lin		
BUPT 学号 BUPT number	2015213432	**QM 学号 QM number**	151007051	**班级 Class**	2015215120
论文题目 Project Title	App design and realization for Photo identification of circuit				

是否完成任务书中所定的中期目标？Targets met (as set in the Specification)?
YES

已完成工作 Finished work:

[1] What material was read or researched?

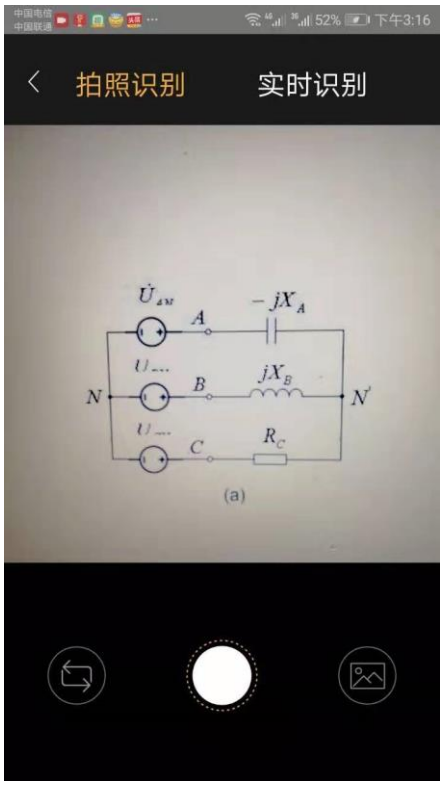
- App program
  "First Code--Android" *--GuoLin*
- Photo Recognition
  "Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library" -- Adrian Kaehler
  "Learning OpennCV: Computer Vision with the OpenCV *Library"*
- Python
  "Python programming: from entry to practice"
- Regular expression
  "Regular expressions must know"
- DeepLearn
  Learning TensorFlow *-- O'Reilly Media*
- Some GitHub Project
  ImageAI-- https://github.com/Robinatp/Tensorflow_Lite_Demo
  Tensorflow_Lite_Demo-- https://github.com/Robinatp/Tensorflow_Lite_Demo
  OpenCVForAndroid-- https://github.com/kongqw/OpenCVForAndroid

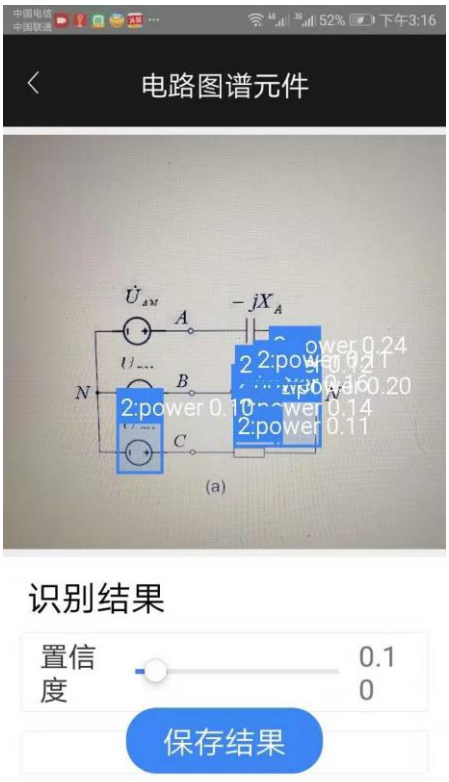
[2] What work was done?

- **App Program Module**

Before the mid-term, I continued to improve and enhance the UI of the App based on the results of the early-term. Adding the functions with the app including

1. By taking photo to achieve the recognition of component of circuit sketch.
2. Load the existing photo album, to recognize the component of circuit
3. Could adjust the precision ratio, to get different the result of the recognition.
4. Real-time recognition, without taking photo, and align the lens to the target image, we can get the real-time result.





Function Description

In this app, I integrated the training model of python+TensorFlow, which is mainly implemented taking photo to achieve the recognition of component of circuit sketch and Real-time recognition, without taking photo function.

Take a photo to identify I call the camera interface of the phone on the app side, then pass the image into the integrated and trained model to achieve the purpose of recognition.

In real-time recognition, the main object-oriented is a fast moving object. At this time, the user has no way to take a picture of the target all the time. In the real-time recognition mode, it actually takes a screenshot of the screen at a certain time, and then puts this as a cache, the image is directly transferred to the training model for recognition. Since everything is operated in the background, the recognition speed is relatively fast.

In addition, I also set the threshold as the standard for object detection to meet the user's detection of the target for different requirements. In particular, when the threshold is 0.2, the overall recognition rate of the model is the highest, but for different tags, different thresholds need to be adjusted.

And when you choose to save the results, it will be saved locally in the form of json for subsequent record viewing.

Encapsulation

1. Tectonic neural network
2. Training neural network model
3. Export the trained model as a pb file
4. Load the pb model for calculation on Android

```
def dump_graph_to_pb(pb_path):
    with tf.Session() as sess:
        check_point = tf.train.get_checkpoint_state("./model/")
        if check_point:
            saver = tf.train.import_meta_graph(check_point.model_checkpoint_path + '
            saver.restore(sess, check_point.model_checkpoint_path)
        else:
            raise ValueError("Model load failed from {}".format(check_point.model_ch

        graph_def = tf.graph_util.convert_variables_to_constants(sess, sess.graph.as

        with tf.gfile.GFile(pb_path, "wb") as f:
            f.write(graph_def.SerializeToString())
```

After we train the model, save the training results to the model folder. This result contains the parameter information obtained by the training model. It generates a pb file by the above code. Then the pb file can be used on android. Then we deploy the TensorFlow api in Android and load the pb file directly for calculation.

- **The Recognition Module**

Project Feature

First of all, when I started the building of this recognition module, I considered the following **features** of this project.

1. The recognized images are all black and white, easy to convert to 2-value images, avoiding the pre-processing of color segmentation
2. The identified parts are geometrical figures with relatively regular borders and relatively simple shapes, and the outlines are easily extracted.

Based on the above considerations, I initially considered using the HOG+SVM network framework to achieve this goal, only using opencv to process images, bypassing deep learning to build a network model, and logically processing the target image to achieve the goal of target recognition.

Problems

However, in practice, I found that there are two problems:

- One is the problem of positioning the target. Because this project's requirement is different from most other examples like face recognition models, the goals are different. This project requires much in amount and accurate with similar object, such as the schematic diagram of the capacitor and the power supply. The difference is very small, and the feature acquisition is difficult to get. Especially when the data set is small, which makes it difficult to achieve
- The second problem is the logical judgment about the circuit connection. For the next stage of the project, that is, after the component identification completed, I have to complete the simple connection relationship identification of circuits, such as series connection and parallel connection.

Only using the HOG+SVM framework, it is very difficult to achieve this goal. Using logical judgment alone consumes a large amount of memory for each recognition, and time. Especially for non-traditional connections, such as hybrids connection, which are not effective.

Summary

In summary, after considering many factors, I decided to use TensorFlow combined with the Faster R-CNN model for image recognition.

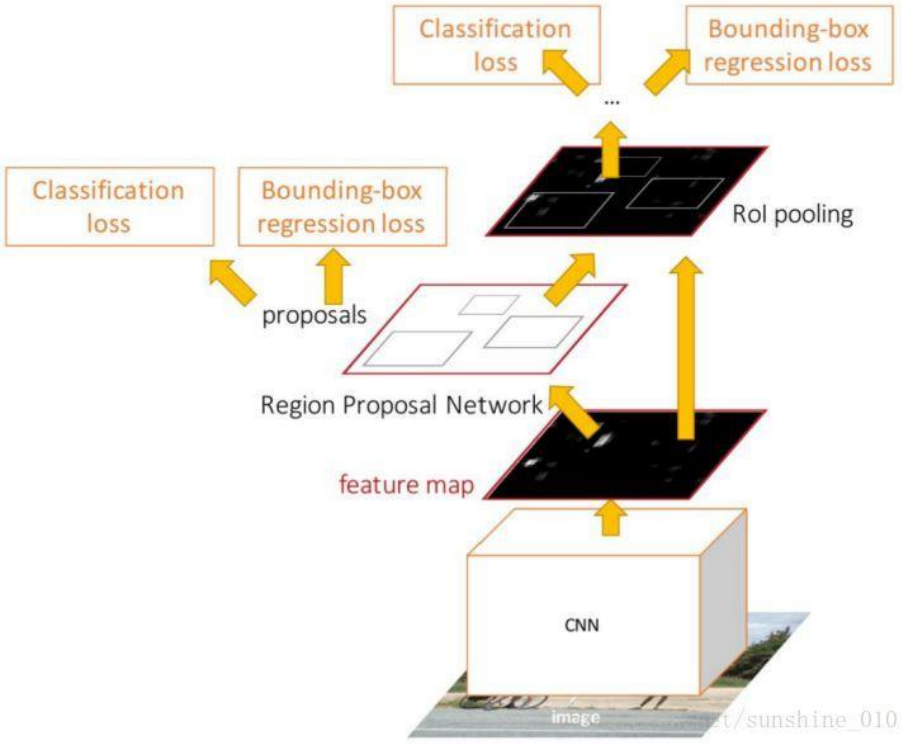
Faster R-CNN is a new and improved algorithm based on Fast RCNN. The algorithm uses Region Proposal Networks (RPN) to generate target candidate domains.

The RPN is a full convolutional neural network. The previous layer is a feature map of any size. Generally, the candidate region generation strategy is to perform a sliding window operation on the feature map of the last convolution layer of the network.

For each window, k target candidate regions are generated at the same time, and each candidate region has different sizes and proportions, and the k candidate regions become anchors. Each window is mapped to a low-dimensional vector, which is then passed into two sub-networks: the border classification network and the border regression network.

The border classification network outputs the probability that each anchor belongs to the target or background, and the border regression network outputs the value of the translation scaling of each anchor.

The principle shown as follow



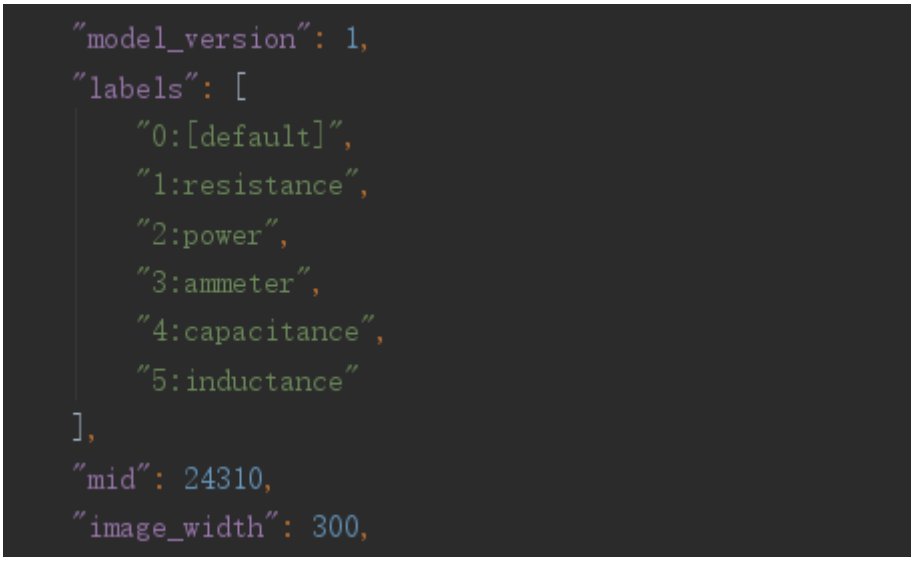
Operating procedures

Step1: I found a trained model initialization model on the Internet, and collected data sets to custom an RPN network;

Step 2: Train a Fast RCNN using the region proposal as input, which is generated by the RPN in step one.

Step 3: Initialize the RPN network, with the network parameters of the Fast RCNN network trained in step two, but only adjust the unique RPN network layer。

Step 4: Use the classifier to determine whether it belongs to a specific class, then output

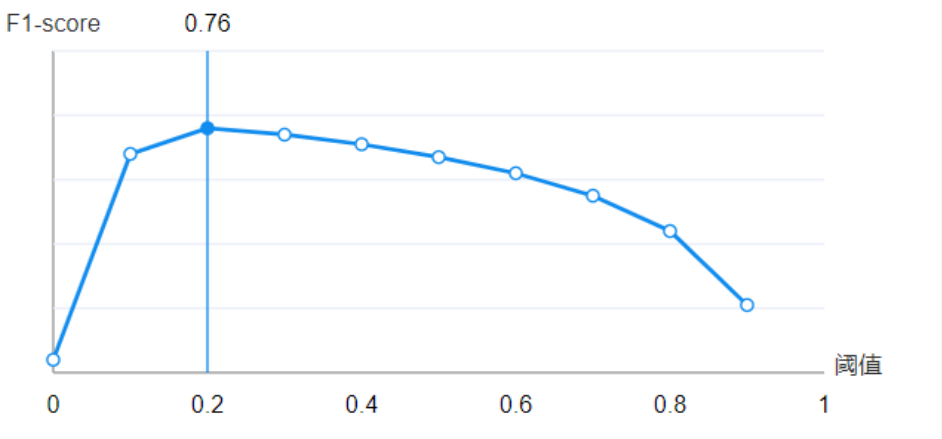


As shown in the figure, the label information in Android and the kernel information, I have defined five sets of labels as target recognition. Resistance, Ammeter ,Power ,Inductance ,Capacitance as the target.

- **Result Analysis**

Technical terminology

1. TP (True Positive): predicting the correct answer
2. FP (False Positive): wrong to predict other classes as this class
3. FN (False Negative): This type of label is predicted to be other types of labels.mAP (mean average precision)
4. Precision: refers to the proportion of positive samples in the positive example determined by the classifier. $precision_k = \frac{TP}{TP + FP}$
5. Recall rate: refers to the proportion of the total positive case that is predicted to be positive. $recall_k = \frac{TP}{TP + FN}$
6. Accuracy: represents the correct weight of the classifier for the entire sample. $accuracy = \frac{TP + TN}{TP + TN + FP + FN}$
7. F1-score: F1-score is the harmonic mean of the average accuracy and recall rate for each category. $f1_k = \frac{2 \cdot precision_k \cdot recall_k}{precision_k + recall_k}$
8. mAP (mean average precision) :For object detection tasks, each type of object can calculate its precision and recall rate. It can be calculated multiple times under different thresholds. Each class can get a P-R curve, the area under the curve is mAp



When different thresholds are set for processing, the F1-score is the highest when the threshold is 0.20, and the identification rate of each label is

Resistance: 94%.
Ammeter 76%.
Power 81%
Inductance 84%
Capacitance 39%
And, the whole mAp is 77.3%

In the above figure, we can clearly see that the model can maintain a relatively high accuracy when the threshold is between 0.1 and 0.2, but the accuracy is lower as the threshold increases. But then as my data set continues to add, I believe that the overall accuracy will continue to improve.

尚需完成的任务 Work to do:

- After completing the identification of the circuit components, I also need to complete the logical identification of the connection of the circuit.
- When calling the mobile phone camera, because of the shooting angle and other issues, it can't be the same as the ideal test data. The angular error of the deflection will also cause the recognition rate to drop. I need to identify it and find a way to reduce this error.
- During the identification process, the ideal test picture is different from the actual actual picture taken. We need to improve the recognition accuracy of the mobile phone shooting end.
- After completing the logic recognition, I also need to complete the logic solution based on the logic identification, including simple rules, including Ohm's law, kcl, kvl.
- Finally, we need to do design recommendation algorithms on the basis of each type of topic to recommend similar topics.

存在问题 Problems:

1. Only using the HOG+SVM framework, the subsequent logical recognition cannot be completed.

2. There is a difference between the test result of the pc-side ide and the test result after the package is applied to the app, and the recognition accuracy after the package is applied to the app is less than the accuracy under the ideal state.

拟采取的办法 Solutions:

1. The Faster R-CNN network framework is used for identification.

2. It may be that when the phone is running, the image is compressed, causing the recognition rate to drop, and the input problem of the mobile phone is handled.

论文结构 Structure of the final report: