**One Circle**

**Team details**

* Jingtang Ma (Arthur): Senior in Computer Science & Minor in Math; University of Missouri -- Kansas City; Contact: [jmkqf@umsystem.edu](mailto:jmkqf@umsystem.edu)
* Molan Zhang: Graduate in Computer Science; University of Missouri -- Kansas City; [mz9kk@mail.umkc.edu](mailto:mz9kk@mail.umkc.edu)
* Qiao Yang: Senior in Information Techonogly; University of Missouri -- Kansas City; [qjy2fc@mail.umkc.edu](mailto:qjy2fc@mail.umkc.edu)
* Benjamin Nguyen: Computer Science Major; University of Missouri -- Kansas City, pdnrtv@umsystem.edu

**The story and its details**

There are people and communities in need of help: company managers, policemen, FBI agents, and CIA agents. The story can be divided into two parts: enterprises and the public security department.

For enterprises, managers are pretty busy. They are facing the highest pressures every day, they need to make those employees work most efficiently and they need to make the most important decision for the company. They do not have much more time for taking a recording of those employees’ attendance. As a result, the managers have to lighten the burden in their hands. When managers want to know the employee's monthly attendance table, the problem will take place. The model we trained can record attendance of each employee and provide a daily report to their managers on the computer screen in their office rooms. The reason why managers use One Circle is managers do not have extra time spent recording employee's attendance.

For the public security department such as police departments or FBI departments, it is hard to find criminals on surveillance video/photo by eyes. When policemen or agents try to find criminals as quickly as possible, One Circle can help them. The model we trained can track a specific face when this face appears on surveillance video/photo, One Circle will send a short report to let agents/policemen know where the person is by using Trilateration[2]. The reason why agents/policemen use One Circle is they cannot work all day all night to check each surveillance video/photo to find criminals.

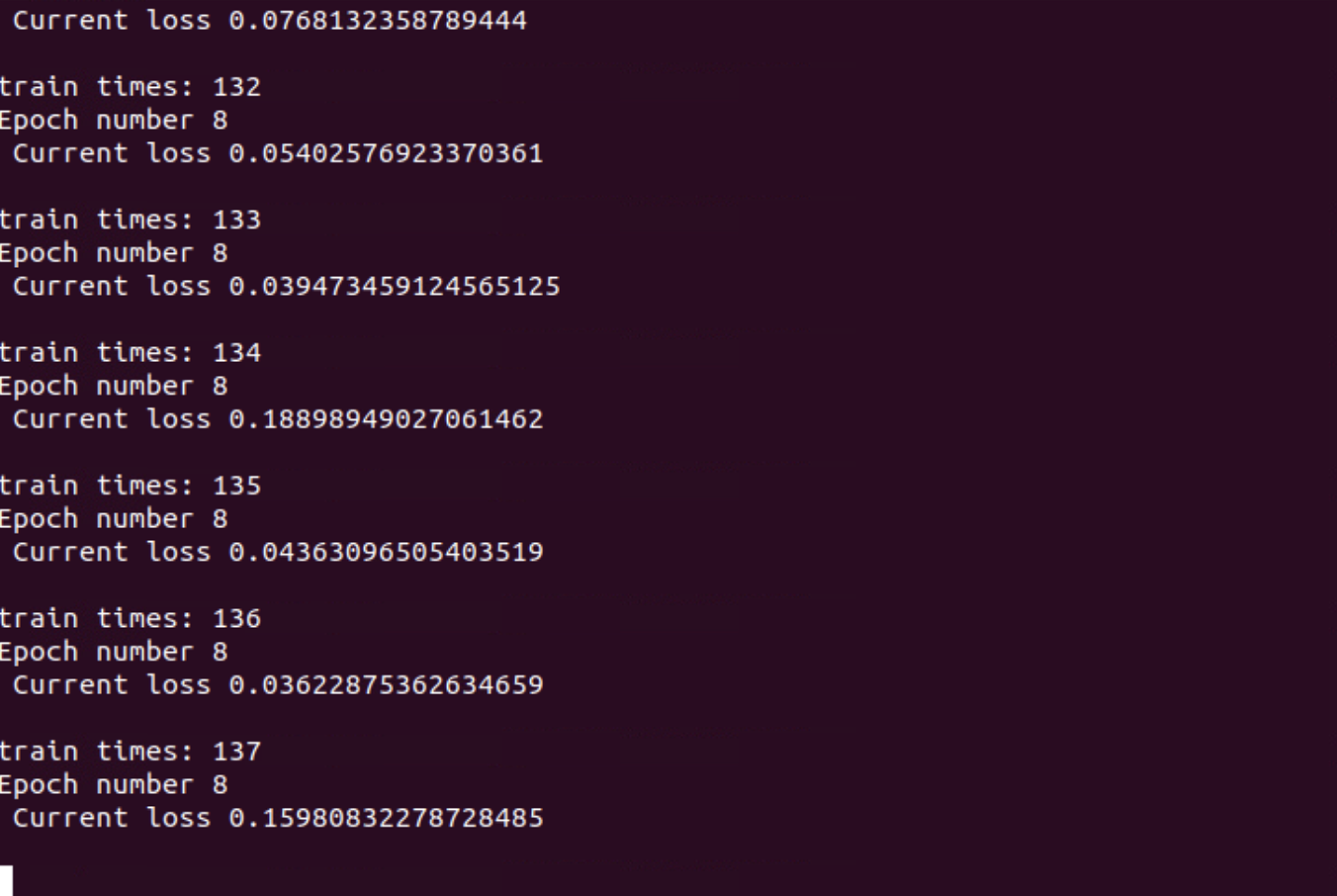
**The data and its details**

In this project I use the Labeled Faces in the Wild Dataset[1] as our training and testing dataset. This is a face photo database designed for studying the problem of face recognition in the natural environment. The dataset contains more than 13,000 images collected from the Internet. Each face is marked with the person’s name. This dataset contains a total 5749 person’s photos, of which 1,680 people have two or more different photos, which are captured by the Viola-Jones face detector.

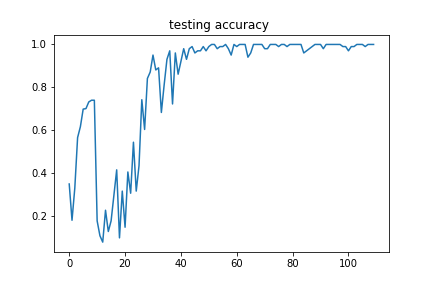
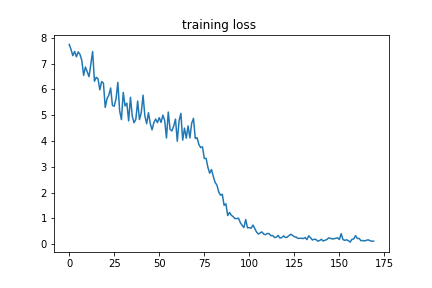
Given that the ideal application scenario of this model is the company's commuting records and the pursuit of suspects by the public security organs, the input should come from real life. For the company, the input photos of the model may be taken in the office building. For public security organs, the pictures are more from roads, shopping malls and other public places.

**Working screens from project**

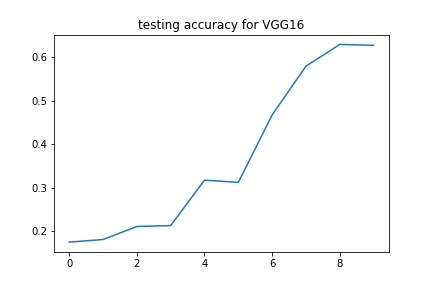
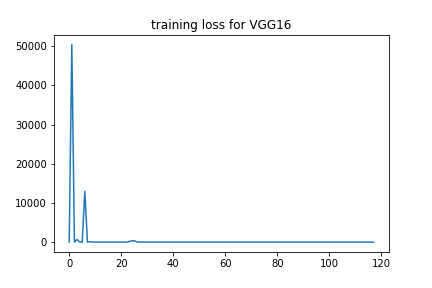
* Training process
  + This image shows the training process. Here I trained the Triplet network for 10 perches. And for the linear classification network I trained 100 epches. It takes about one hour.

’

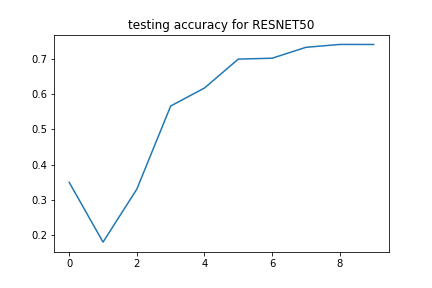
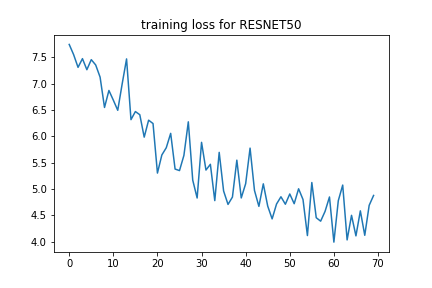
* Training loss rate and Validation accuracy rate



* Test result analysis
  + Pre-trained VGG16



* + Pre-trained Resnet50



Here I compared our model with two pre-trained models: one is VGG16 and the other is ResNet50. The table below shows that after 100 epoches training process on the same dataset. All of three models can get more than 50% accuracy, which is not bad on this dataset with about 6,000 classes. But we can still find that the triplet net performances best due to the accuracy, precision, recall, and F1-score.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | accuracy | precision | recall | F1-score |
| VGG16 | 0.681 | 0.809 | 0.690 | 0.699 |
| ResNet50 | 0.740 | 0.843 | 0.753 | 0.753 |
| **Tri-VGG16** | **0.857** | **0.913** | **0.849** | **0.881** |

* Conclusion

In one word, based on the above table, we believe that the model can extract meaningful facial features and filter out invalid information, such as environment, clothing, hairstyle, etc. Since each person has only 5-10 photos in this dataset, which shows that the model can quickly learn effective information from a small and complex data set. It is a successful model.

**Improvement from the previous increment**

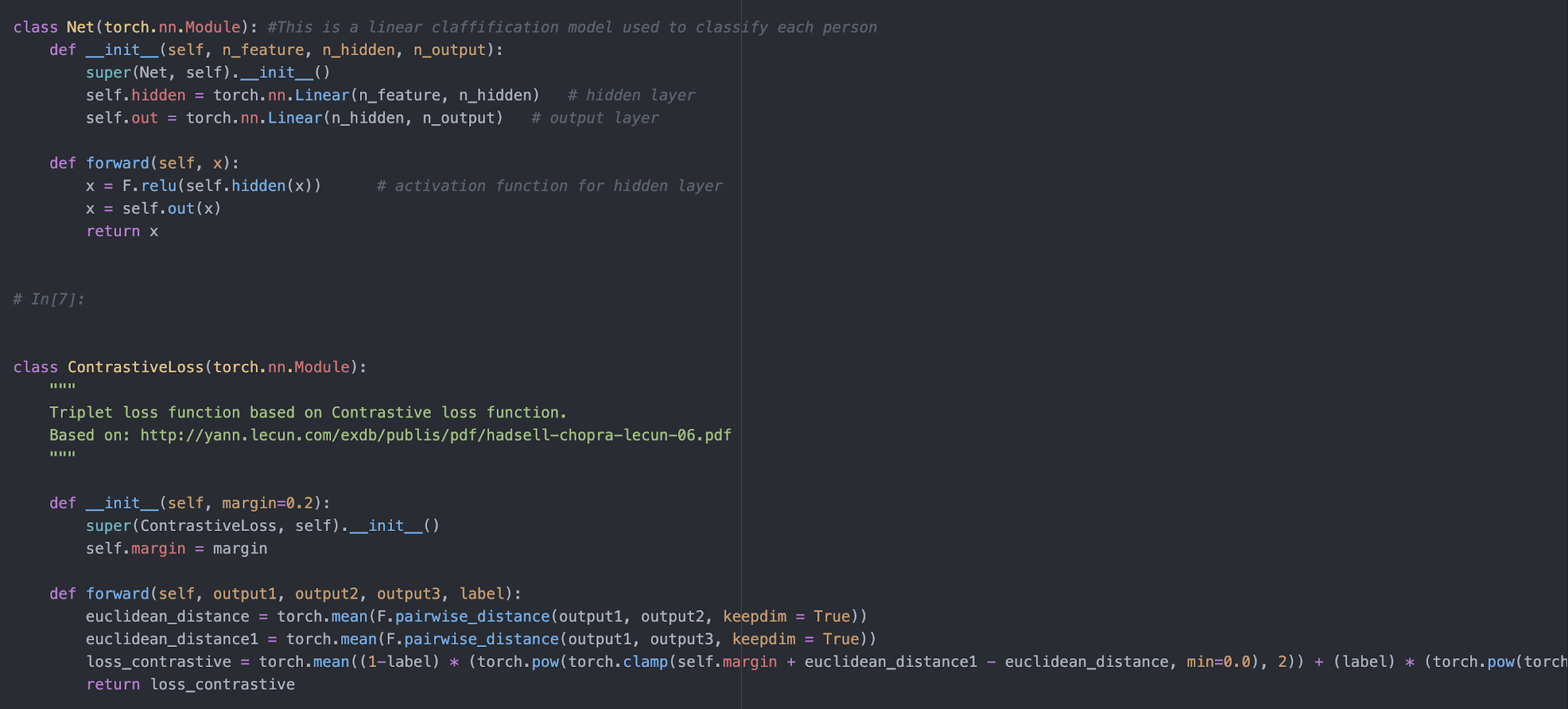
In this increment, we improved performance of the model, and we completed training and evaluation of all models. Then, we have done comparison with Tri-VGG16, VGG16, and ResNet50. Finally, we can conclude that Tri-VGG16 has the best performance in the others depending on the results. For coding, we improved the readability of the code. Considering modularity and cohesion, the code is decomposability and composability, the models should be more reusable and more testable, and we improved the code to high cohesion which means the elements inside the module have a high degree of connectedness. For documentation, I have captured screenshots to make the report understandable, and done the final conclusion.

**Important code snippets from the project**

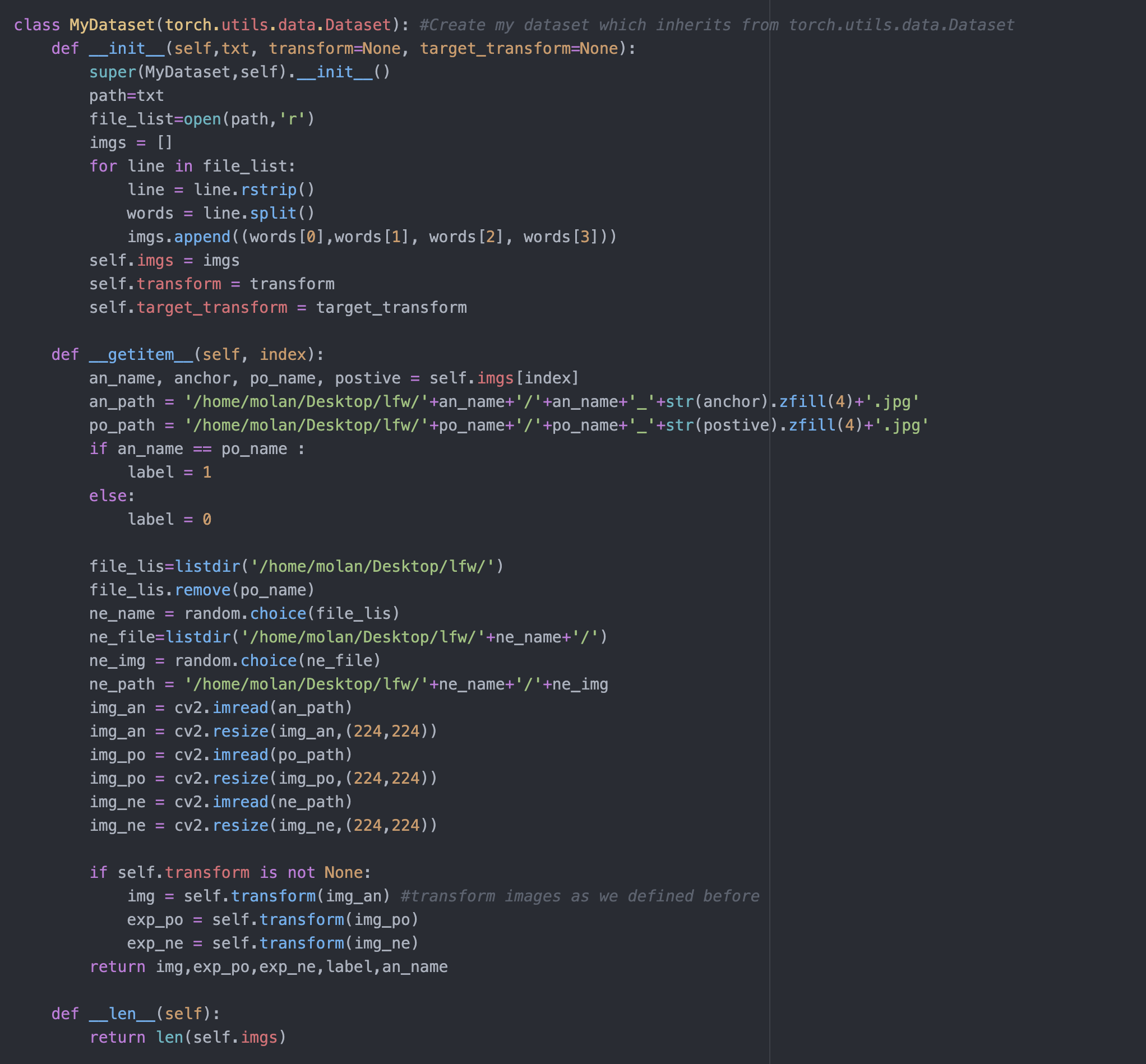
* Define tri-VGG model:



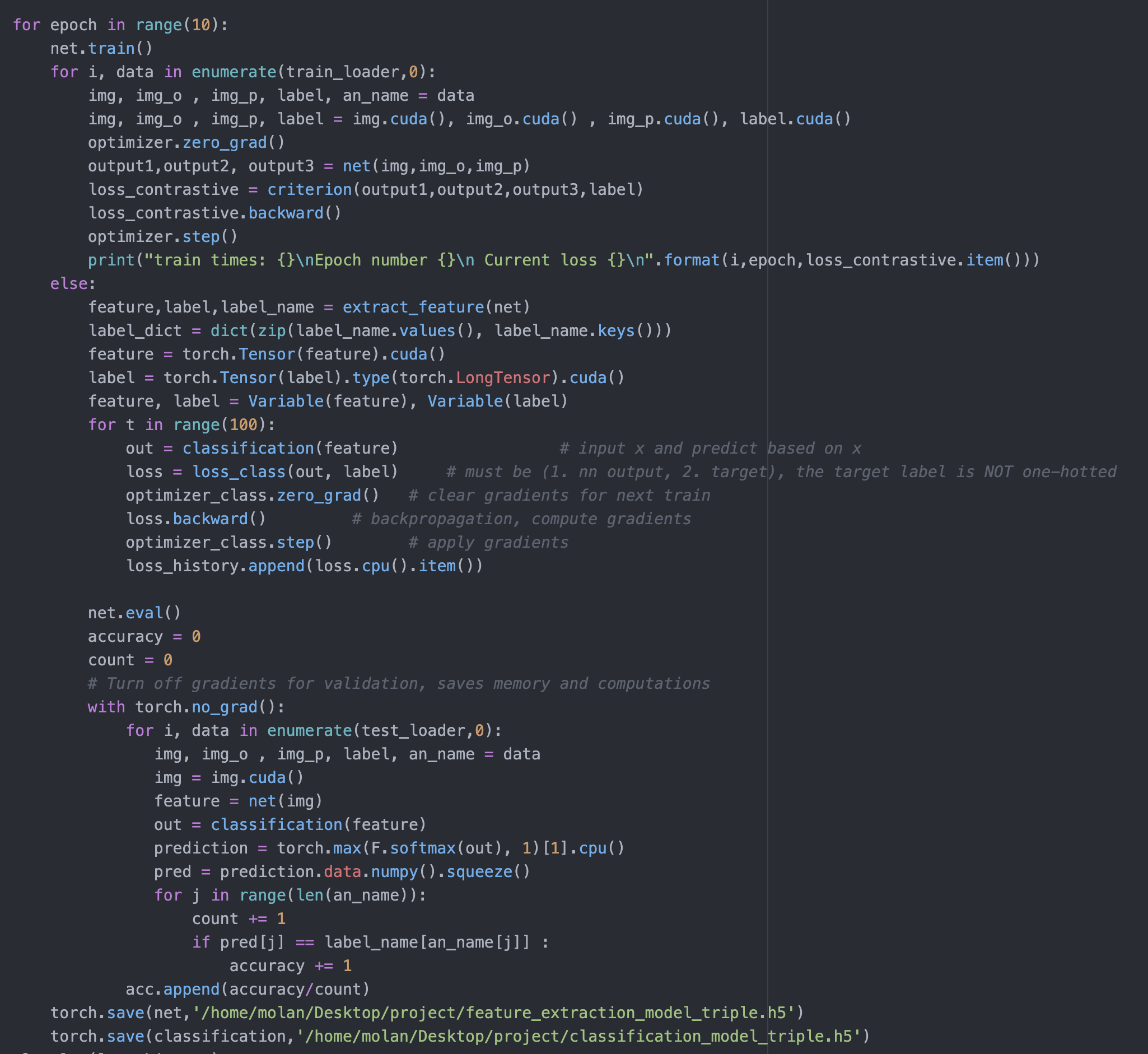


****

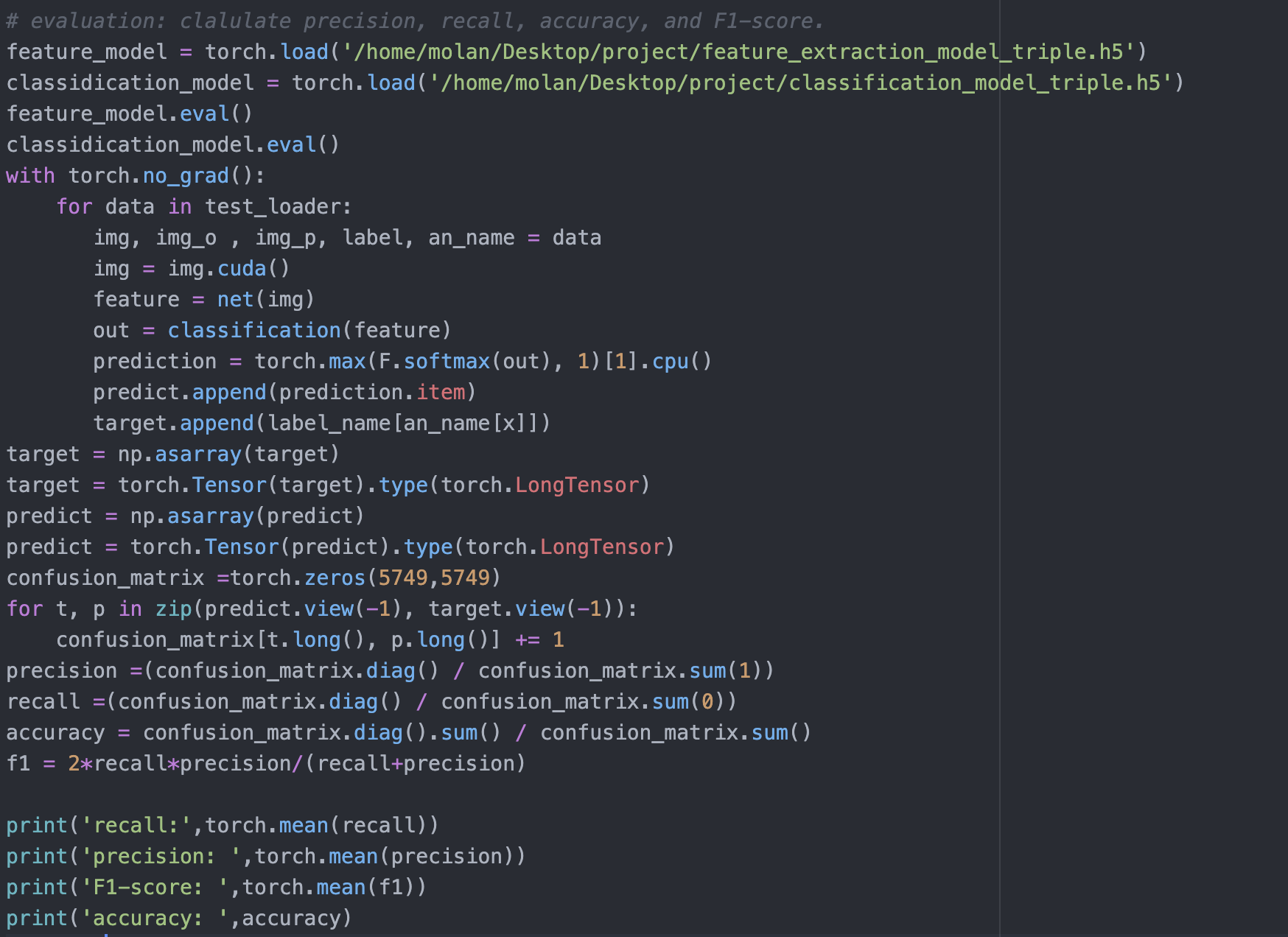
* create our own dataset:



* training process:



* evaluation:



**Work sharing/Module sharing between teammates**

* Jingtang Ma (Arthur): Work with the story and its details; Help Molan evaluating the model; Communication with the instructor; Come up with the agenda and communicate with the team.
* Molan Zhang: Work with the data and its details; Work with the code; design model and improve it; evaluating model; record video.
* Qiao Yang: organizing the report, code review, and testing
* Benjamin Nguyen: Finding issues, blockages during projects (in reality and program), running, testing, reviewing the code to collect issues. Contribution in story and details.

**Any issues, blockages with the project**

In reality, we have to face privacy problems. Security and privacy are issues we need to concern. Need to find a way to make sure these data of face we record are used in the safe and right way and doesn’t leak the data because information is very important. In addition, the program can be used for some bad purposes such as spying or selling dataset, this violates the privacy permission. Our program cannot guarantee that the result is great but we make it as similar as possible with the realistic model because there is probably the possibility of misrecognition, which leads to wrong decisions. It is supposed to be dangerous if government or police agents make wrongful convictions.

While working with the code, one of the problems that we face is there are too many packages that are used such as pytorch (including torchvision, cuda, optim), it took time to find out the way to install and import. Second, the problem with CUDA enabled and the torch, it requires a GPU that is strong enough to process so it may be hard to run with a weak GPU. Moreover, CUDA version should be compatible with torch, it means we need to check the CUDA version before installing torch. It will catch errors if CUDA enabled a different version with torch used. Third, training and testing data took a significant time till the final result during each epoch, but if we want the most accurate result, we need to use enough needed epochs to catch a good result. Another issue, it is not important, but using VGG-16 may not change the accuracy score significantly (as we can see in the graph), and it makes the model’s weight heavy, increasing interference time. It obviously catches good results eventually.

For this model usage, we also need to consider the loss cost for misclassification. For example, the cost of missing a criminal is larger (false positive) than identifying a wrong person as a criminal by mistake (false negative). In the result, we might use the BoostCost classifier as the final classification model which can create our own cost function. Also, due to the splical edition for Python and some packages, it is hard to transfer this model to the other different environments. We also plan to put this model in a docker when we finish the training process.

**GitHub link for your project**

* https://github.com/CS5590-0001-Projject/CS5590-0001-Project

**Video link for your project**

* https://drive.google.com/file/d/1W6G0cTNPVfmUWEC-hjcQO6eVN\_50rDon/view?usp=sharing

**Presentation link for your project**

* https://docs.google.com/presentation/d/1DE-fRGCHNRRMo3HlhanvjF3cJAemH43ancbfO\_nHEnw/edit?usp=sharing

**References**

* Huang, G. B., Mattar, M., Berg, T., & Learned-Miller, E. (2008, October). *Labeled faces in the wild: A database for studying face recognition in unconstrained environments*.
* L. A. Martínez Hernández, S. Pérez Arteaga, G. Sánchez Pérez, A. L. Sandoval Orozco and L. J. García Villalba, "Outdoor Location of Mobile Devices Using Trilateration Algorithms for Emergency Services," in IEEE Access, vol. 7, pp. 52052-52059, 2019, doi: 10.1109/ACCESS.2019.2911058.