CSC 578 HW 7: Convolutional Neural Networks

Update: 5/21/2022: Updated the notebook file and pdf in Step 1 to current version of TensorFlow, and included a zip file.

- Again, much credit to Prof. Tomuro
- **Graded out of 5 points.**
- Here is a dynamic multimedia video description of the assignment
- **DO NOT include the question** in your submission. Just clearly label your submission with section headers.

Overview

The objective of this assignment is to enhance your understanding of Convolutional Neural Networks (CNNs) while gaining experience with Kaggle competitions at the same time. Kaggle is an online machine learning community. It provides a large collection of public data sets and hosts competitions to solve data science and machine learning challenges.

The assignment is to experiment with CNNs and write about your experience on the hyperparameter tuning and the analysis of results. The CNN experimentation part is made into a **Kaggle inClass competition** so that you can compete with other students in the class. Your assignment submission will consist of your code and your documentation of your journey of hyperparameter tuning along with the analysis of your results and your post-competition reflections.

The competition is to classify the CIFAR-10 image data: one of the most widely used benchmark datasets for image classification. In this assignment, you will use CNNs to classify the images. The dataset is already divided into training and test sets, and you use the training set to train a CNN model (though you further split the training set to training and validation subsets in the code). After training the model, you generate predictions for the test set and submit the predictions to the competition.

Note: The competition will close on Sunday, May 29th, at 6:59 PM CDT. This is strictly enforced by Kaggle. There can be no submissions to Kaggle after this time. The submissions for the class (described in the Submission section) will be accepted until 11:59PM Central time.

Also note that this competition is intended for **fun**. Don't be concerned or take it personally if you did not place high in the leaderboard; ranking will not affect your homework grade.

Code Development

A. You can start with the start-up code cnn_578_hw7_2022.ipynb (It was made in Colab.) Here's a zip file with the notebook. And here's the pdf for it, using TF2.)

- Note that there are three places in the code where you choose the appropriate syntax/line(s) for the version of TensorFlow (TF version 1.X or 2.Y) you are using. You can use either version of TF, but since not all TF2's syntax is backward compatible for some operations, you have to use the right syntax for your version of TF.
- For example, Google CoLab is now using version 2.8. If you prefer working on your local machine, and your machine doesn't have TensorFlow, you have to install it on your end. Or if you have access to an environment in which TensorFlow is already installed, you should first check the version and choose the appropriate lines in the code.
- Then run the code to ensure it works in your system/environment.
- **Running the code**: If you can run your code with hardware acceleration (GPU or TPU), it will **greatly increase** the speed! With Colab, you can do this with `Runtime > Change Runtime Type > Hardware Accelerator > GPU or TPU`. There is a limit on how long you can use these, and it's not a published formula, but it will make your training much faster. You might want to disconnect your session in between training runs to avoid hitting the limit. You can also sign up for "Colab Pro" (under settings). I think it's just $10/month. It provides faster hardware and more relaxed runtime restrictions.

B. Add code in the file to generate predictions for **test_images**, the held-out input data.

- The function to generate predictions is `predict()`, such as '`model.predict(...)`'. You can look at the HW#4 code for syntax.

- Save predictions in the code. So you know, they will be stored in a list of lists in Keras, where each inner list is the prediction for a test image, i.e., a list of length 10, carrying the probability of each target class. For example,

```
[[1.3942997e-06 1.3935754e-04 5.6055570e-03 6.6085613e-01 6.7078590e-04
  2.8833924e-02 3.0388865e-01 5.1890177e-09 3.2569944e-06 8.7058385e-07]
 [9.5547803e-05 4.3761898e-02 2.9793470e-12 3.3167865e-12 1.0443895e-13
  1.6440552e-17 4.0106508e-11 6.3863180e-22 9.5614260e-01 9.3374570e-09]
```

  - For a submission to Kaggle, you should create a csv (comma-separated values) file. You may name the file as you wish, but ensure that the extension is `.csv`. Also the content of the file must be formatted **exactly** to the following specifications:

    - The file must have the header on the first row, then the predictions of test images to follow.
    - The prediction of each image must start with an ID number. You can simply traverse the saved prediction list from the beginning and assign an integer sequence from 1 (NOT 0) to 10000 (the size of the test set).

- Follow the ID number with the predicted values (10 of them) for each class.
- **No extra characters, not even spaces**.
- Submit predictions for the **test set** not for the training set.
- Repeat for each test image, one per line, as shown below.

```
id,cat0,cat1,cat2,cat3,cat4,cat5,cat6,cat7,cat8,cat9
1,1.3942997e-06,0.00013935754,0.005605557,0.6608561,0.0006707859,0.
2,9.55478e-05,0.043761898,2.979347e-12,3.3167865e-12,1.0443895e-13,
...
10000,4.3202136e-15,1.2549072e-13,1.8496052e-10,2.0123039e-11,5.050
```

C. Notes on hyperparameter tuning.

- The model in the start-up code has three convolution layers (and two maxpooling layers) and one fully-connected layer in addition to the input and output layers. Some hyperparameters are already plugged in such as activation functions. You can play with any parameter, and as many as you like. You may want to start with those mentioned in the previous assignment HW#4. Some suggestions:
  - Change the number of filters.
  - Change the size of filters (e.g. 5*5, 7*7).
  - Change the size of stride – although you must choose the sizes that 'work'.
  - Add more Convolution layers.
  - Add more Fully Connected layers.
  - If you think the network is overfitting, try adding Dropout layers. Other ideas include adding regularization and removing layers (to make the model simpler).
  - Add BatchNormalization layers. Look at the Keras Documentation to learn about it.
- As for automatic optimal parameter search such as GridSearch, you can do so for the purpose of the competition, but I recommend you play with one parameter at a time manually and observe the effect to develop your understanding of the parameter. Automatic parameter search is knowledge-free, therefore you don't learn much.
- Also keep in mind the performance can be measured by other aspects besides the competition evaluation metric, such as computational speed and learning stability. Another thing to consider is the model complexity vs. performance trade-off. Try to think from a higher point of view while you do your experiment.
- One more. You may want to look into an advanced technique called *Data Augmentation* – which inject distorted images to make the model less overfitting and more robust.
- HOWEVER, you must NOT use a pre-trained network (such as VGG net or MobileNet) in this assignment or competition.

- If you want to train longer (i.e. more epochs), you can save the current model and load it later to resume training. Look at the Keras tutorial code 'Save and Load' for reference.

D. Notes on the Kaggle competition.

- You are allowed to submit a maximum of 4 submissions in a day. The number is intentionally set to a small size in order to force you to practice using the validation set to monitor training, rather than having the Kaggle results (for the held-out test set) influence your model development.
- Note that you will see the 'public' leaderboard during competition. Then after the competition closes, the 'private' leaderboard becomes available. The way it works is the test set is divided (by Kaggle) into public and private subsets (when the competition is created), and the public leaderboard shows the ranking based on the performance for the public test set. This is also to prevent overfitting to the competition ranking (during competition). The final ranking will be the one for the private test set, so you know.
- You must make a successful submission to the competition. Failure to do so will have a significant impact on the points that you can earn. Also, this should go without saying, but you should **NOT** submit the sample_submission.csv file which is used as the baseline in the competition, though you may look at it to compare the formatting.
- Your submission to the Kaggle competition will be evaluated with the 'LogLoss' function). LogLoss is defined as:

$$LogLoss = -\frac{1}{N}\sum_{i}^{N}\sum_{j}^{M} y_{ij} \cdot \ln(p_{ij})$$

where N is the number of images in the test set, M is the number of image class labels and ln is the natural logarithm. y_{ij} is 1 if observation i belongs to class j and 0 otherwise, and p_{ij} is the predicted probability that observation i belongs to class j. Since each image is labeled with exactly one true class in this dataset, the metric is equivalent to LogLikelihood. With some effort, you may be able to achieve somewhere in the 0.6 range (or even lower) for this CIFAR-10 dataset.

E. Create a document that describes/reports your model development journey, your results, and your conclusions and reflections. See below.

Submission

Submit these **two items**:

A. A Jupyter notebook code file and a pdf version of it. Be sure to add your name and course/section number at the top of the code file.

- Show at least the best model and its full trace of training with charts. You can leave other models in the code too, but make sure you **clearly** indicate the best model. (Make

it easy to find for someone who is looking at dozens of these!)

- In general, organize the code file well, by inserting section headers and comments/mark-downs. Code with no comments will be subject to points deduction.
- **Please note:** The code file is submitted for verification and spot-checking. Please don't assume that the grader will search for graphs in your code. If you want something seen, put it in your report.

B. Report

- In pdf.
- Minimum 600 words / 2 full pages of text plus visuals.
- Be sure to add your name, course/section number and the assignment name at the top of the file.
- Also at the top of the file, write, **in bold**, your Kaggle user name (as displayed in the leaderboard) and ranking/score (public or private, or both; at the time of writing).
- **Content** (clearly labeled, professionally organized and presented):
  A. **Description of your best model** (the one that produced the best competition performance).
  B. **Description of your journey of hyperparameter tuning.**
    - Starting from the initial model, describe in depth:
      - the hyperparameters and values you tried,
      - **why** you tried them (including your expections),
      - and the results you got and your reaction to them.
    - Write insightful comments, as much as you can. A large portion of the grade will be placed on this part.
    - In the "journey description", describe one **non-best model** that caught your attention or that you found noteworthy. Discuss its performance results and say why you found it interesting.
  C. **Your final conclusions** on the best model.
  D. **Your reaction and reflection** on this assignment overall (e.g. difficulty level, challenges you had).
- **Note 1**: Reports which are nicely organized and well written, with a sufficient amount of comments and presentable graphs/charts will receive a higher grade. Ones with terse, minimal content will be considered insufficient and receive a lower grade.
- **Note 2**: Do not repeat the questions, but do include brief headers corresponding to the numbered sections above, e.g., **Description of Best Model**

DO NOT ZIP YOUR FILES. SUBMIT EACH FILE SEPARATELY.

Assessment

Submissions will be scored based on:

- 1.5 points: Making a reasonable kaggle submission which goes a good faith effort to produce good performance on the task.
- 2 points: Excellent description of best model and hyperparameter journey
- 1.5 point: Other aspects of assignment, including: non-best model description, conclusions, reactions, overall clarity, charts of results.

Spring 2022 () 2022-05-21 Sat 15:03