CSCI 8360 Data Science Practicum
Department of Computer Science
University of Georgia

Project 3: Image Classification

DUE: Thursday, October 13 by 11:59:59pm

Out September 22, 2016

1 Overview

You jumped from four-category document classification on a dataset that could fit in memory to 10-category malware classification on a dataset that would crush most laptop SSDs. You've demonstrated remarkable creativity, be it in the specific algorithms used, the feature sets generated, or even in the coding styles and team chemistry. As you surely knew beforehand but have now seen firsthand, there's more to coding than just the code itself. You have every right to be proud of the work you've done to this point!

Now, we move from text to images. Specifically, the CIFAR-10 image dataset.

Your next task is, yet again, to design a classifier. However, the focus will be a bit more on the feature engineering; since you'll be working with a large collection of small images, the matter of generating features is a little less defined. Consequently, the realm of tools available to you will be expanded. It is still highly recommended that you stick with the Spark distributed compute engine, but you are more than welcome to use packages built and implemented on top of Spark (e.g. SparkNet, Caffe-on-Spark) in addition to Spark's built-in utilities (MLlib, GraphX, etc).

To emphasize: I *strongly* recommend you stick with Spark, but provided you are still using a cluster of machines (i.e. you cannot do this on 1 node), **you are welcome to use a non-Spark distributed framework** (e.g. h2O, Flink, Hadoop, etc).

2 Data

Each file is a PNG image; 60,000 of them, named `00000.png` through `59999.png`. Each PNG image is 32×32 pixels, with three color channels: standard RGB format. All these images are accessible through a web browser at the following URL:

`https://s3.amazonaws.com/eds-uga-csci8360/data/project3/images/<file>`

and accessible on AWS through the S3 path:

`s3n://eds-uga-csci8360/data/project3/images/<file>`

Each image belongs to one of 10 possible conceptual image categories (hence the name: CIFAR-10). There are three metadata files that are also available: `X_train.txt`, `y_train.txt`, and `X_test.txt`. The respective `X*` files contain the 5-digit photo IDs, one per line, corresponding to the PNG filenames of the images that are in the training and testing datasets. The `y_train.txt` file contains the corresponding labels for the photos in the training set; again, one per line. You will find these text files at the following browser URL:

`https://s3.amazonaws.com/eds-uga-csci8360/data/project3/metadata/<file>`

and accessible on AWS through the S3 path:

`s3n://eds-uga-csci8360/data/project3/metadata/<file>`

Your goal is to develop an image classification pipeline that aims for the highest possible accuracy on the `X_test.txt` dataset. The ground-truth labels are stored in AutoLab and will be used to grade your submissions. Make sure you generate a label text file where the labels are in the same ordering as the input files appeared.

3 Guidelines

Everyone should probably use Spark. However, this is not set in stone. Should you deviate from Spark, I expect sufficient justification in the accompanying `README.md` file. If you stick with Spark, what language API you use is entirely up to you and your team. You are also allowed (and encouraged) to use libraries that build on top of Spark, such as SparkNet or Caffe-on-Spark, in order to give yourselves a leg up on some fancier image processing.

Create a GitHub repository for you and your team to work on the project. You are more than welcome to make it a private repository, but the development must take place on GitHub, as I will be using it and the activity you record on it as part of

the grading process. I would highly recommend using the repository naming scheme of `team-name-project3` to help differentiate from other teams and future projects. Make sure your repository includes

1. a `README.md` giving an overview of your program, how to install and run it, and your design rationale (especially if you chose to use a framework other than Spark)
2. a `CONTRIBUTORS.md` file containing the names of everyone on your team and their respective contributions to the project, and
3. your code.

Exercise good software design principles. This means: adopt a consistent coding style, document your code, use the GitHub ticketing system to identify milestones and flag bugs, and provide informative and frequent `git commit` comments. If your entire team is working on the code, make sure you've split up the work accordingly to take advantage of everyone's strengths.

Use flintrock or EMR (if using Spark). You have two options here. You can either install flintrock for setting up and tearing down EC2 clusters for Spark, or you can use Amazon EMR (Elastic MapReduce) for spinning up ephemeral clusters in the AWS web dashboard.

Use EC2 or GCE (if not using Spark). EC2 will be your best AWS option if you're using a non-Spark distributed framework (if using Hadoop, go with EMR). Use AMIs to build a working environment once and use that to spin up clusters so you don't need to reinstall all the prerequisites every time you spin up a new EC2 cluster.

Shut down your EC2 clusters when you are finished testing. AWS EC2 instances incur costs for every hour they are turned on, even if they are not actively running any jobs. I will not hesitate to revoke AWS credentials that are racking up disproportionate expenses. Here's the EC2 dashboard you can use to double-check if you have any resources allocated: `https://quinngroup.signin.aws.amazon.com/console`

Be careful to keep your AWS credentials out of GitHub. It's very easy to hard-code AWS credentials into configuration files or even the code itself, put these files under version control, and push them to the remote repositories. If this happens, take steps *immediately* to remove the commit history from GitHub (search StackOverflow for how to do this), as we have already had one instance of AWS credentials being picked up from public GitHub repositories and used fraudulently.

4 Submissions

All submissions will go to **AutoLab**.

You can access AutoLab at `https://autolab.cs.uga.edu`. Your team will first need to create an account on AutoLab before you can be added to the `csci8360-fa16` course and submit assignments.

You can make submissions on behalf of your team to the **Project 3** assignment that is open. When you do, you will submit one file: **a text file containing the predictions of your trained classifier on the `X_test.txt` dataset, one prediction per line for each document**. For example, if you had six images in the test set to classify, your output might look like:

```
1
7
7
3
5
3
```

Your classification accuracy will be tabulated and compared against the true labels (hidden on the server) and should appear on the leaderboard.

If you think you can do better than what shows up–particularly to beat out everyone else–make another submission! There's no penalty for additional submissions, but keep in mind that swamping the server at the last minute may result in your submission being missed; AutoLab is programmed to close submissions *promptly* at 11:59pm on Sept 15, so give yourself plenty of time!

5 Grading

Given that this is a practicum, and that you're all talented graduate students, the grading for the projects is a little different, operating on a sliding scale with some stationary points that require extra effort to move beyond.

5.1 Baseline

As last time, everyone's starting grade is a **B**. To maintain this grade, you'll need at minimum–

- A simple core classification strategy (e.g., a single Decision Tree classifier) that

runs over features that are largely drawn from the raw data itself (e.g., grayscale pixel intensities flattened in long vectors).

- A submission to AutoLab that performs considerably better than random (in this case, well above 10% accuracy).
- A `README.md` file that clearly explains the contents of your code, how to run it, and explains your design rationale.
- A `CONTRIBUTORS.md` file that lists your team members and what their role in the project was.
- GitHub tickets and corresponding milestones that show a coherent and cohesive overarching project design plan.
- Well-documented, well-designed, easy-to-read source code.

Meet all these criteria, and you and your team receive a **B**. If you're happy with this, excellent! If not, read on...

5.2 Improvements

To move beyond the baseline, you'll need to demonstrate a combination of *algorithmic novelty* and *engineering best practices*. Regarding the algorithm, this can take the form of a more sophisticated learner, or better feature sets, or a combination to boost your accuracy further. For example:

- Use of deep learning.
- More sophisticated feature sets, e.g. SIFT or SURF, to better quantize images and take advantage of the RGB format.
- Ensembles of weak learners, with boosting and bagging.
- Autoencoders (unsupervised feature learning) to create compact representations of the images, before feeding into a classifier.
- Deployment of TensorFlow, Caffe, Theano, or another deep learning framework on Spark or a different distributed platform (**your program must run on a cluster!**).
- Obtaining the top AutoLab testing accuracy.
- [your great idea here!]

For the engineering, this can take the form of example scripts, install instructions, outstanding design patterns, a `LICENSE` file to attach an open license to your code, or even containerized deployments for ease of reproducibility (e.g. Docker builds).

Implementing some or all of these will push your team into the well-deserved **A** range. Good luck!

6 Reminders

- You can use any functionality that's available in Spark, that's built on top of Spark, or–*provided it utilizes clusters of machines*, and does not simply run on one node–that's entirely outside of Spark. Make sure you provide justification for your architecture!
- If you run into problems, work with your teammates. If you still run into problems, query the Slack channel. I'll be regularly checking Slack and interjecting where I can.
- This is the first time 8360 has been offered, and the first time AutoLab has been used, so there are bound to be hiccups. Please be patient! Thanks for your understanding.