Class Project for 10-601 Machine Learning: Fall 2014

William Cohen

November 14, 2014

1 Overview

This year's project will be a competitive project based on the task of image classification. The dataset we will use will be a subsample of CIFAR-10[1], a ten-class classification task involving small color images.

We will set up several datasets, mostly based on subsamples of this data, and will use Kaggle to administer the contest.

- The *training data* contains 4000 labeled images, and will be used to build and tune your classifiers. Important: **You must not use resources on the web to extend this training set: the classifiers you submit must be trained only on the provided training set.**

- The *test data* contains 15,000 images. You will not be able to see the labels of these images, but you will be able to submit predicted labels to Kaggle to evaluate a classifier. Of these 15,000 images, only 5,000 labels will be available during the development phase: more exactly, you will submit labels all the test images, but get back accuracy numbers over the 5,000 development images.

  The final set of 10,000 images will be released evaluation 24 hours before the project is due, as a final check to make sure that you haven't drastically overfit the data.

[1] http://www.cs.toronto.edu/~kriz/cifar.html

There will be one milestone, in which each student must submit the result of training a classifier (either off the shelf or something you have written in past assignments). The final project is fairly unconstrained: the goal is simply to do something interesting that performs well on this classification task, for instance, by engineering/tuning a classifier, using ensembles of classifiers, or using any other approach you like. While we will look at the accuracy of the classifiers submitted, the project will be primarily graded on the quality of the writeup.

Project groups should be two students. If you want to work as a singleton or in a three-person team, you should get permission from Ziv or William.

2 Project Assignment: Milestone 1

2.1 Purpose and overview

Milestone 1 (MS1) will be due at 11:59PM Dec 1. The purpose is to make sure that you get an early start on processing the data, familiarize yourself with some classifier baselines, and that the class as a whole is informed about which techniques work well on this dataset (at this training-set size).

You should run one classifier per team member on the training dataset, submit a Kaggle entry for each, and write a short document describing what learners were used; how they were trained (e.g., with what parameters); approximately how long the training took, and what sort of machine you trained on; and what results were obtained, both quantitatively and qualitatively. For this milestone, we strongly recommend that you use a stable library containing several off-the-shelf classifer learners, such as WEKA.[2] You are welcome to explore multiple classifier learners if you like, and explore multiple parameter settings, but the only requirement is that each team submit one classifier run per team member.

The writeup for MS1 should be at most one paragraph per learner, with an emphasis on a classmate being able to easily reproduce what you did. Optionally you can put a pointer to a web site that contains more detail, and/or which you update as you improve your results.

For example, you might say:

For classifier one, we ran the random forest learner from Weka 3.7 (Java class weka.classifiers.trees.RandomForest) on the full training dataset, build-

[2] http://www.cs.waikato.ac.nz/ml/weka/

ing an ensemble of 200 trees trees of depth at most 20 (options -I 200 -depth 20 and using the default parameters otherwise. Training took less than 5 minutes and obtained an accuracy of 33%. Most of the errors were between similar-looking classes, such as birds and airplanes.

For the second classifier, . . .

For updates and details see http://www.cs.cmu.edu/∼ wcohen/601project/

2.2 Deliverables and grading

The data, and some starter code, is available from

https://inclass.kaggle.com/c/10-601-fall-2014-project-object-recognition

The starter code include a sample submission and some code that runs a simple linear classifier and outputs the data in the correct format. A CMU email is required to register. Questions about Kaggle should be directed to Harry or Jin (or Piazza).

Details on how to submit your MS1 writeup as a team (and the final report) will be posted by Nov 24 on the class wiki.

These writeups will be made available to the class, and are required, but no grade for them will be assigned. They will help you in deciding which techniques you want to explore more fully in the next phase of the project. Questions about the mechanics of submitting your MS1 report should be directed to Jingwei, Debjani, or Piazza.

3 Final project and report

3.1 Purpose and overview

In the next phase of the project, you will chose another method to explore more deeply. You may wish to read about what other methods people have tried on the full dataset[3], and reproduce some of their results; or you may wish to explore a method that you are personally interested in. *For the purpose of the writeup, you should describe your work as experimentally testing ideas on how to improve your baseline method(s).* These might include ideas you explored on how to tune the parameters of a classifier, or to design a new ensemble of classifiers. Its fine if you explore more than one baseline,

[3]https://www.kaggle.com/c/cifar-10

or idea for improving the classifier, but it is preferable if there is some single theme or "story" behind your project: there should be some reason why you're working as a team and submitting a joint report. We emphasize that you should be using *systematic* experimentation that's intended to improve performance on this task. However, within this constraint, you can focus on any aspect of the learning problem you wish—be creative!

For instance, you can build a new ensemble method, based on the utilities in Weka; you can combine clustering and classification; you can adopt feature-selection methods; you can explore and configure a learner based on another package like the deep-learning package Theano.[4]; you can work with a classifier you implement yourself. (Warning: we will not give credit for only partially implementing a new classifier, however, so much sure you get this finished if you go that route!)

You may also explore other representations of the data, but the focus of the project is machine learning, not image-processing so *you are not encouraged to spend any substantial amount of time on engineering the image representation.* The exception to this rule might be if you spend time using techniques discussed in class, like dimensionality reduction or k-means. (Put another way, don't expect to get much credit for work you've done on the project using techniques that fall outside the scope of this class.)

Your decisions about what to do for the project can (and should) be informed by your classmates experiences with the data, as well as your own: there's no penalty for switching classifiers after MS1. You can also conduct additional experiments with data outside the CIFAR-10 collection, if you feel like it will be helpful to test your hypotheses. However, your final CIFAR-10 submission should be based on the distributed training data.

You are also expected to describe and motivate your experiments in a format appropriate for scientific results. To do this, you will want to think carefully about the experiments you conduct: what conjecture are you testing? what is the purpose behind, say, conducting a parameter sweep, or choosing a particular learning method to explore? You will also need to keep records of your experiments, or be prepared to redo them periodically.

[4]http://deeplearning.net/software/theano/

3.2 Choosing a project topic

One possible strategy would be to start with a reasonable-sounding abstract, and some results you might hope to get, and focus your experiments around that, so it will be easier to tell your "story" in the writeup. Here are some examples of such a starting point, with the conjectured outcomes in italics.

1. Many of the top-performing methods for image classification with many examples are based on deep learning. However, deep learning methods are also computationally expensive, and require extensive parameter tuning. In this project, we compared a fixed-architecture convolutional networks with linear classifiers when trained under resource constraints. Specifically, we implemented a network learning method that supported early stopping, and limited training time to the time needed to train a tree ensemble, a relatively fast learning method, on the same data. *We showed that by carefully tuning the learning rate and capacity the network, one could obtain accuracy better than the tree ensemble, even in this resource-limited setting.*

   (Of course your actual result might be different: the final abstract might end *By carefully tuning the learning rate and capacity the network, we were able to obtain 10% higher accuracy than the baseline network learner in this setting; however, the accuracy of the resource-limited deep learner was still consistently lower than the tree ensemble. Comparable accuracy could be obtained with a time budget of triple the time needed by the tree learner.*)

2. One successful past approach to image classification has been to combine linear classifiers with a k-means clustering method to find higher-level features. In this project, we reproduced this work, and explored several variants: in particular, we explored combining k-means clustering with feature selection methods, and also explored using the k-means clustering representation with several other learning methods. *We showed that using feature selection methods, about 2/3 of the k-means clusters could be discarded without significantly impacting accuracy, leading to a much smaller overall model. We also showed improvements of 10% over the baseline k-means/linear combination, using an ensemble method to replace the linear classifier.*

3. Although convolutional networks are widely used for image classification, they require extensive parameter tuning to be effective, and the procedures used for parameter tuning are not well-established. We explored three systematic approaches to tuning parameters for convolutional networks, and report on their effectiveness for CIFAR-10 and two additional image-classification datasets, compared to a baseline tuning strategy that performs a grid search over three parameters. *The best tuning strategy achieved 10% higher accuracy, while requiring only 1/4 the total training time.*

4. In this project, we explore the bias-variance tradeoff for several learning methods on an image-classification task using different-sized training sets. Based on these experiments, we configured a particular ensemble method which was predicted to perform well on the CIFAR-10 data with 4000 training examples, *and demonstrate that the ensemble indeed has 5% higher accuracy than the same ensemble with WEKA's default parameter settings.*

3.3 Deliverables

At noon 12/9, you should submit the code you are using (which may be scripts invoking external learners) in your experiments to Autolab, along with a draft of your final project report. You should not submit code you and your team-mates haven't written: if you're using third-party code like Weka, don't send that to us. The main purpose of this is so that we have a record of the code you're writing.

At 1:00pm 12/9 we will release a final test set on Kaggle. You are only allowed to submit one set of predictions against that final test set. The purpose of this is to ensure you have not overfit the development data with your experiments. You have until 11:59PM 12/10 to complete this run, but you are encouraged to do it earlier: there should be no change to your learning system at this point.

By 11:59pm 12/10, you should also submit your writeup. **This is a hard deadline and there will be no extensions**, since we're running right up against the time that grades need to be turned in. The format is specified below.

3.4 Format for the writeup

The writeup is important - it will be read and graded. It should be at least three pages and at most five, not counting the appendix and bibliography, and should be written using the style format used by ACM: see

http://www.acm.org/sigs/publications/proceedings-templates

This is not a *complete* report of everything you did - you should be writing a report that will help a reader (say, one of your fellow students) reproduce your results. The paper should also give enough information for someone *not* in the class to understand why what you did is useful (if indeed you feel it is), or else, what the goal was, and what if anything was learned in your attempts to accomplish that goal.

The structure of the report is as follows (but the lengths are just suggestions, not requirements).

- **Title and other information.** Use a descriptive title, such as "Using A Logistic Model Tree Learning for Image Classification" or "A Comparison of Margin-Based Learning Methods on the CIFAR-10 Task". Include the name of all project members and their Andrew id's and/or Andrew emails.

- **Abstract.** Briefly summarize the motivation for your technique, the problem you're addressing, and the experimental results you obtained, in the format suggested by the sample abstracts above.

- **Introduction**. Write an expanded version of the abstract, where you give more detail, in your own words, about the motivation for the work you did, and expand on the brief summaries given in the abstract. (About half a page).

- **Background**. Write a summary of the baseline algorithm you extended, including its full name and briefly how it works, including a citation to an appropriate papers. You may want to discuss other closely related algorithms as well. Also include a similar summary of any other techniques (e.g., ensemble methods, filters, etc) that you used. Include in your description the sort of implementation details you used in MS1 (e.g., Weka classnames and option), but do *not* make this the only description. (About 1-2 pages).

- **Methods**. (You may want to chose a more informative title here.) Describe how you extended or adopted the baseline, as clearly as possible. Again, include in your description the details, but do not make this the only description. If you considered more than one interesting technique in your experiments, describe each of them. (About 1-2 pages).

- **Results**.

  - **Preliminary Experiments**. Include a description of some key tuning/comparison experiments you performed in the course of your project.
  - **Development and Evaluation Data**. Include (at least) a table showing accuracy of your baseline and improved baseline on the two Kaggle runs. Discuss them briefly.

  (About 1-2 pages).

- **Conclusions**. Summarize the motivation briefly, the technique you introduced, and your results. If appropriate, discuss intuitions behind the techniques you proposed, or make a conjectures about why they performed as they did. (About half a page).

- **Bibliography**. Include a bibliography with complete citations to all the papers you discuss, in a consistent format.

- **Appendix**. (a) Summarize briefly which parts of the project were performed by which team members. (b) Summarize briefly what you learned in this project, and how you'd change it if it were to be used in a later version of 10-601. (At most one page).

In the writing, *don't* use a lot of lists and bullet points—use narrative text as much as possible. Do proof your paper carefully. Don't use jargon from the assignments and milestones themselves. The report should read like a machine learning paper, and should be comprehensible students that took 10-601 a different semester, or at different school—so they know about as much about machine learning as you, but nothing about the specific assignments from this class.

The writeups will be graded based on their completeness in addressing all points mentioned above, their clarity, and their correctness. An good writeup

will propose significant and interesting techniques, and provide insightful and plausible explanations for their performance.

When you are preparing your report, you should consider the following criteria:

1. Coherence of the outline (assumed if you follow the suggested one above).

2. Clarity, accuracy and completeness of the claims made in the abstract and/or conclusions.

3. Clarity and correctness of the descriptions of baseline learner, and appropriateness of the choice of baseline relative to the motivations, and clarity, accuracy and completeness of the discussion any other background.

4. Clarity and correctness of the improvements to the baseline.

5. Clarity and correctness of the descriptions of the experiments performed.

6. Completeness of the descriptions of the experiments and the implementation: in particular, could the work be re-implemented by a hypothetical reader who has taken 10-601 a different year (so knows the same material, but not necessarily course-specific jargon), and is not familiar with the toolkits used.

7. Readability of the results as presented in the tables/graphics of the writeup, as well as their correctness (as compared to the autolab run results).

8. Correctness and appropriateness of the bibliography. (You can use any consistent format you like, but the bibliography should be contain full citations in a format appropriate for a scientific paper.)

9. How significant/interesting the techniques explored were (do they seem to be plausible ways to improve performance), and how insightful and plausible the explanations for their performance were.

10. Writing style, and correctness of the grammar, spelling, etc.