Project: a Simple Run of Stochastic Gradient

Last updated: January 1, 2020
Goal

- A basic understanding of how stochastic gradient is used to train CNN
- Familiar with our simpleNN package
Download (or clone) the code simpleNN at https://github.com/cjlin1/simpleNN

We will use the Python part for this project.

Use CPU!!

Later we will do timing comparisons with MATLAB/Octave code on CPU

To use the package you need to install Tensorflow

Follow, for example, instructions at https://www.tensorflow.org/install/pip?lang=python3
Make sure first you can run the training and prediction examples shown in README

Consider the following sets

- MNIST
- CIFAR10

from the following directory:
https://www.csie.ntu.edu.tw/~cjlin/courses/optdl2020/slides/proj_data/

These sets are in the MATLAB format but our python code can read them.

Training and test sets are available.
Run these two sets by the simple stochastic gradient algorithm
That is, we use SGD rather than Adam
For the architecture, let’s do

<table>
<thead>
<tr>
<th>Our symbol</th>
<th>filter size $h \times h \times d^m$</th>
<th>#filters $d^{m+1}$</th>
<th>stride s</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv 1</td>
<td>$5 \times 5 \times 3$</td>
<td>32</td>
<td>1</td>
</tr>
<tr>
<td>pool 1</td>
<td>$2 \times 2$</td>
<td>-</td>
<td>2</td>
</tr>
<tr>
<td>conv 2</td>
<td>$3 \times 3 \times 32$</td>
<td>64</td>
<td>1</td>
</tr>
<tr>
<td>pool 2</td>
<td>$2 \times 2$</td>
<td>-</td>
<td>2</td>
</tr>
<tr>
<td>conv 3</td>
<td>$3 \times 3 \times 64$</td>
<td>64</td>
<td>1</td>
</tr>
<tr>
<td>pool 3</td>
<td>$2 \times 2$</td>
<td>-</td>
<td>2</td>
</tr>
<tr>
<td>full 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This network has been implemented as the one called **CNN_31ayers**

Thus you don’t need to handle the network at this moment.

To see details you can trace `net/net.py`. But you can do the project without knowing details now.

For other options, use the default values.

This is a project but not a homework. We don’t expect the same result.
A minor difference in your code or your settings may cause your results slightly different from those of others

- Try different random seeds to see the variance of test accuracy
For the padding size, we avoid the shrinkage of the output image in each convolutional layer by

\[ a^m_{\text{conv}} = a^m. \]  \hspace{1cm} (1)

For the convolution operation, we enlarge \( a^m \) to \( a^m_{\text{pad}} \) so that

\[ a^m = a^m_{\text{conv}} = \left\lfloor \frac{a^m_{\text{pad}} - h}{s} \right\rfloor + 1. \]
Thus

\[ a^m = \left\lfloor \frac{2p + a^m - h}{s} \right\rfloor + 1. \]

Because \( s = 1 \) in our setting, we can let the padding size be

\[ p = \frac{h - 1}{2} \]

so that (1) holds.
For activation function, use
\[ \sigma(x) = \max(x, 0) \] \hspace{1cm} (2)
for convolution layers and use a linear function for the last full layer
\[ \sigma(x) = x. \] \hspace{1cm} (3)
Write a report with ≤ 2 pages in pdf

In your report, beside the training/prediction results, you may discuss the following issues:

- Your environment and any difficulties on installation
- Difficulties in using the package? Which part you think is not very friendly?
- Runing time
- and anything you think is interesting

No need to write lots of things. What I will check are
insight of your observations
whether your argument is clear and logical
THose writing a clear report often get better scores than those getting better accuracy or lots of results without good analyses
Another note is that you want to well organize your code directory as in this course you will do many projects
Students with the following IDs (last three digits): ??
please do a ??-minute presentation (??-minute the contents and ??-minute Q&A)

- Please submit your presentation slides before the class
- People not chosen for presentation do not need to prepare/submit slides
- You can use your computer for the presentation, but please use the submitted version of slides
- You may want to have your code available in your computer. To answer some our questions showing the code is easier