softmax-classifier

September 26, 2019

0.1 PyTorch data

PyTorch comes with a nice paradigm for dealing with data which we'll use here. A PyTorch `Dataset` knows where to find data in its raw form (files on disk) and how to load individual examples into Python datastructures. A PyTorch `DataLoader` takes a dataset and offers a variety of ways to sample batches from that dataset.

Take a moment to browse through the `CIFAR10 Dataset` in `2_pytorch/cifar10.py`, read the `DataLoader` documentation linked above, and see how these are used in the section of `train.py` that loads data. Note that in the first part of the homework we subtracted a mean CIFAR10 image from every image before feeding it in to our models. Here we subtract a constant color instead. Both methods are seen in practice and work equally well.

PyTorch provides lots of vision datasets which can be imported directly from `torchvision.datasets`. Also see `torchtext` for natural language datasets.

0.2 Softmax Classifier in PyTorch

In PyTorch Deep Learning building blocks are implemented in the neural network module `torch.nn` (usually imported as `nn`). A PyTorch model is typically a subclass of `nn.Module` and thereby gains a multitude of features. Because your logistic regressor is an `nn.Module` all of its parameters and sub-modules are accessible through the `.parameters()` and `.modules()` methods.

Now implement a softmax classifier by filling in the marked sections of `models/softmax.py`.

The main driver for this question is `train.py`. It reads arguments and model hyperparameter from the command line, loads CIFAR10 data and the specified model (in this case, softmax). Using the optimizer initialized with appropriate hyperparameters, it trains the model and reports performance on test data.

Complete the following couple of sections in `train.py`: 1. Initialize an optimizer from the torch.optim package 2. Update the parameters in model using the optimizer initialized above

At this point all of the components required to train the softmax classifer are complete for the softmax classifier. Now run

```
$ run_softmax.sh
```

to train a model and save it to `softmax.pt`. This will also produce a `softmax.log` file which contains training details which we will visualize below.

Note: You may want to adjust the hyperparameters specified in `run_softmax.sh` to get reasonable performance.

0.3 Visualizing the PyTorch model

[6]:
```
# Assuming that you have completed training the classifer, let us plot the␣
 ↪training loss vs. iteration. This is an
# example to show a simple way to log and plot data from PyTorch.

# we neeed matplotlib to plot the graphs for us!
import matplotlib
# This is needed to save images
matplotlib.use('Agg')
import matplotlib.pyplot as plt
%matplotlib inline
```

```
//anaconda3/envs/cs7643/lib/python3.6/site-packages/ipykernel_launcher.py:7:
UserWarning:
This call to matplotlib.use() has no effect because the backend has already
been chosen; matplotlib.use() must be called *before* pylab, matplotlib.pyplot,
or matplotlib.backends is imported for the first time.

The backend was *originally* set to 'module://ipykernel.pylab.backend_inline' by
the following code:
  File "//anaconda3/envs/cs7643/lib/python3.6/runpy.py", line 193, in
_run_module_as_main
    "__main__", mod_spec)
  File "//anaconda3/envs/cs7643/lib/python3.6/runpy.py", line 85, in _run_code
    exec(code, run_globals)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/ipykernel_launcher.py", line 16, in <module>
    app.launch_new_instance()
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/traitlets/config/application.py", line 658, in launch_instance
    app.start()
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/ipykernel/kernelapp.py", line 563, in start
    self.io_loop.start()
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/tornado/platform/asyncio.py", line 148, in start
    self.asyncio_loop.run_forever()
  File "//anaconda3/envs/cs7643/lib/python3.6/asyncio/base_events.py", line 421,
in run_forever
    self._run_once()
  File "//anaconda3/envs/cs7643/lib/python3.6/asyncio/base_events.py", line
1425, in _run_once
    handle._run()
  File "//anaconda3/envs/cs7643/lib/python3.6/asyncio/events.py", line 127, in
_run
    self._callback(*self._args)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-packages/tornado/ioloop.py",
```

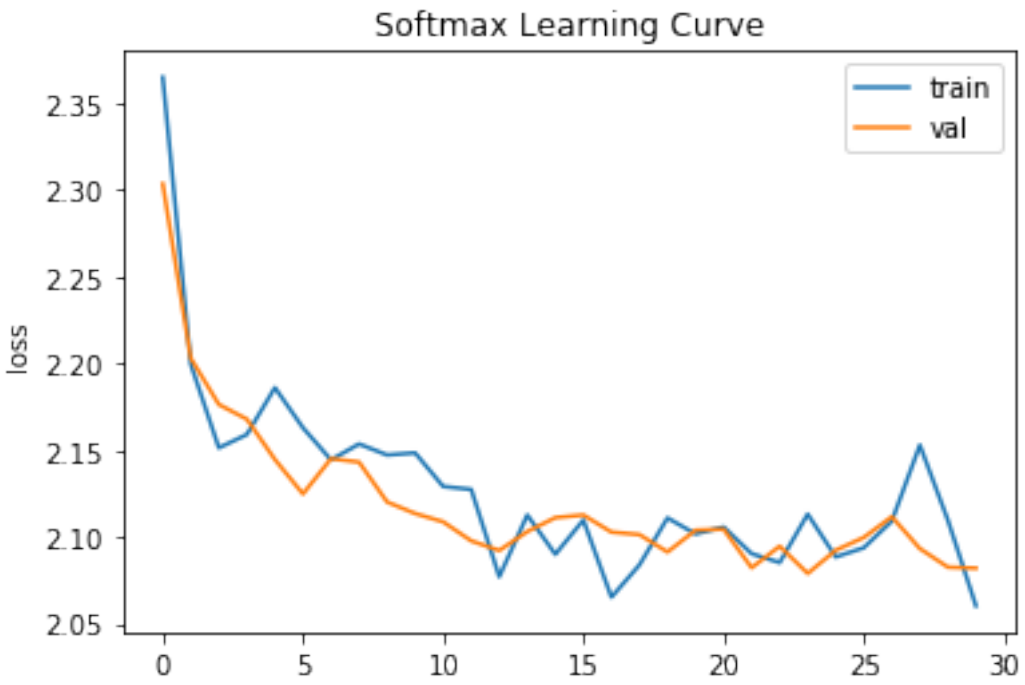
```
line 690, in <lambda>
    lambda f: self._run_callback(functools.partial(callback, future))
  File "//anaconda3/envs/cs7643/lib/python3.6/site-packages/tornado/ioloop.py",
line 743, in _run_callback
    ret = callback()
  File "//anaconda3/envs/cs7643/lib/python3.6/site-packages/tornado/gen.py",
line 787, in inner
    self.run()
  File "//anaconda3/envs/cs7643/lib/python3.6/site-packages/tornado/gen.py",
line 748, in run
    yielded = self.gen.send(value)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/ipykernel/kernelbase.py", line 365, in process_one
    yield gen.maybe_future(dispatch(*args))
  File "//anaconda3/envs/cs7643/lib/python3.6/site-packages/tornado/gen.py",
line 209, in wrapper
    yielded = next(result)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/ipykernel/kernelbase.py", line 272, in dispatch_shell
    yield gen.maybe_future(handler(stream, idents, msg))
  File "//anaconda3/envs/cs7643/lib/python3.6/site-packages/tornado/gen.py",
line 209, in wrapper
    yielded = next(result)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/ipykernel/kernelbase.py", line 542, in execute_request
    user_expressions, allow_stdin,
  File "//anaconda3/envs/cs7643/lib/python3.6/site-packages/tornado/gen.py",
line 209, in wrapper
    yielded = next(result)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/ipykernel/ipkernel.py", line 294, in do_execute
    res = shell.run_cell(code, store_history=store_history, silent=silent)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/ipykernel/zmqshell.py", line 536, in run_cell
    return super(ZMQInteractiveShell, self).run_cell(*args, **kwargs)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/IPython/core/interactiveshell.py", line 2855, in run_cell
    raw_cell, store_history, silent, shell_futures)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/IPython/core/interactiveshell.py", line 2881, in _run_cell
    return runner(coro)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/IPython/core/async_helpers.py", line 68, in _pseudo_sync_runner
    coro.send(None)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/IPython/core/interactiveshell.py", line 3058, in run_cell_async
    interactivity=interactivity, compiler=compiler, result=result)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
```

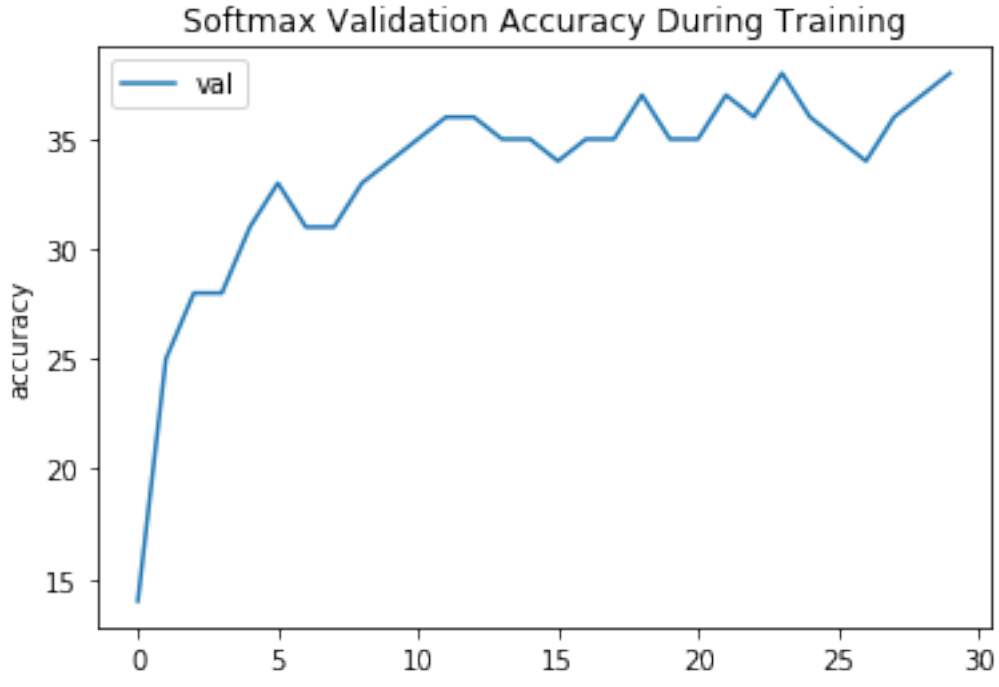
```
packages/IPython/core/interactiveshell.py", line 3249, in run_ast_nodes
    if (await self.run_code(code, result,  async_=asy)):
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/IPython/core/interactiveshell.py", line 3326, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
  File "<ipython-input-4-b4795f515c11>", line 9, in <module>
    get_ipython().run_line_magic('matplotlib', 'inline')
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/IPython/core/interactiveshell.py", line 2314, in run_line_magic
    result = fn(*args, **kwargs)
  File "</anaconda3/envs/cs7643/lib/python3.6/site-
packages/decorator.py:decorator-gen-108>", line 2, in matplotlib
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/IPython/core/magic.py", line 187, in <lambda>
    call = lambda f, *a, **k: f(*a, **k)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/IPython/core/magics/pylab.py", line 99, in matplotlib
    gui, backend = self.shell.enable_matplotlib(args.gui.lower() if
isinstance(args.gui, str) else args.gui)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/IPython/core/interactiveshell.py", line 3414, in enable_matplotlib
    pt.activate_matplotlib(backend)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/IPython/core/pylabtools.py", line 314, in activate_matplotlib
    matplotlib.pyplot.switch_backend(backend)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/matplotlib/pyplot.py", line 231, in switch_backend
    matplotlib.use(newbackend, warn=False, force=True)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/matplotlib/__init__.py", line 1410, in use
    reload(sys.modules['matplotlib.backends'])
  File "//anaconda3/envs/cs7643/lib/python3.6/importlib/__init__.py", line 166,
in reload
    _bootstrap._exec(spec, module)
  File "//anaconda3/envs/cs7643/lib/python3.6/site-
packages/matplotlib/backends/__init__.py", line 16, in <module>
    line for line in traceback.format_stack()


  import sys
```

[8]:
```
# Parse the train and val losses one line at a time.
import re
# regexes to find train and val losses on a line
float_regex = r'[-+]?(\d+(\.\d*)?|\.\d+)([eE][-+]?\d+)?'
train_loss_re = re.compile('.*Train Loss: ({})'.format(float_regex))
val_loss_re = re.compile('.*Val Loss: ({})'.format(float_regex))
```

```
val_acc_re = re.compile('.*Val Acc: ({})'.format(float_regex))
# extract one loss for each logged iteration
train_losses = []
val_losses = []
val_accs = []
# NOTE: You may need to change this file name.
with open('softmax.log', 'r') as f:
    for line in f:
        train_match = train_loss_re.match(line)
        val_match = val_loss_re.match(line)
        val_acc_match = val_acc_re.match(line)
        if train_match:
            train_losses.append(float(train_match.group(1)))
        if val_match:
            val_losses.append(float(val_match.group(1)))
        if val_acc_match:
            val_accs.append(float(val_acc_match.group(1)))
```

[9]:
```
fig = plt.figure()
plt.plot(train_losses, label='train')
plt.plot(val_losses, label='val')
plt.title('Softmax Learning Curve')
plt.ylabel('loss')
plt.legend()
fig.savefig('softmax_lossvstrain.png')

fig = plt.figure()
plt.plot(val_accs, label='val')
plt.title('Softmax Validation Accuracy During Training')
plt.ylabel('accuracy')
plt.legend()
fig.savefig('softmax_valaccuracy.png')
```


Softmax Learning Curve
train
val
loss
2.35
2.30
2.25
2.20
2.15
2.10
2.05
0
5
10
15
20
25
30


Softmax Validation Accuracy During Training
val
accuracy
35
30
25
20
15
0
5
10
15
20
25
30

[ ]: