Deep Learning (Homework 2)

Due date : 5/18/2018

- High-level API are forbidden in this homework, such as Keras, slim, TFLearn, etc. You should implement the forward computation by yourself.
- Homework submission – Please zip each of your source code and report into a single compress file and name the file using this format : HW2_StudentID_StudentName.zip (rar, 7z, tar.gz, ... etc are *not* acceptable)

1 Convolutional Neural Network for Image Recognition

In this exercise, you will construct a Convolutional Neural Network (CNN) for image recognition by using Food-11 dataset. The dataset contains 16,643 images grouped into 11 categories, which basically cover the major types of food that people consume in daily life. The 11 categories are *Bread*, *Dairy product*, *Dessert*, *Egg*, *Fried food*, *Meat*, *Noodles/Pasta*, *Rice*, *Seafood*, *Soup*, and *Vegetable/Fruit*. The figure below is example food images of the 11 categories.

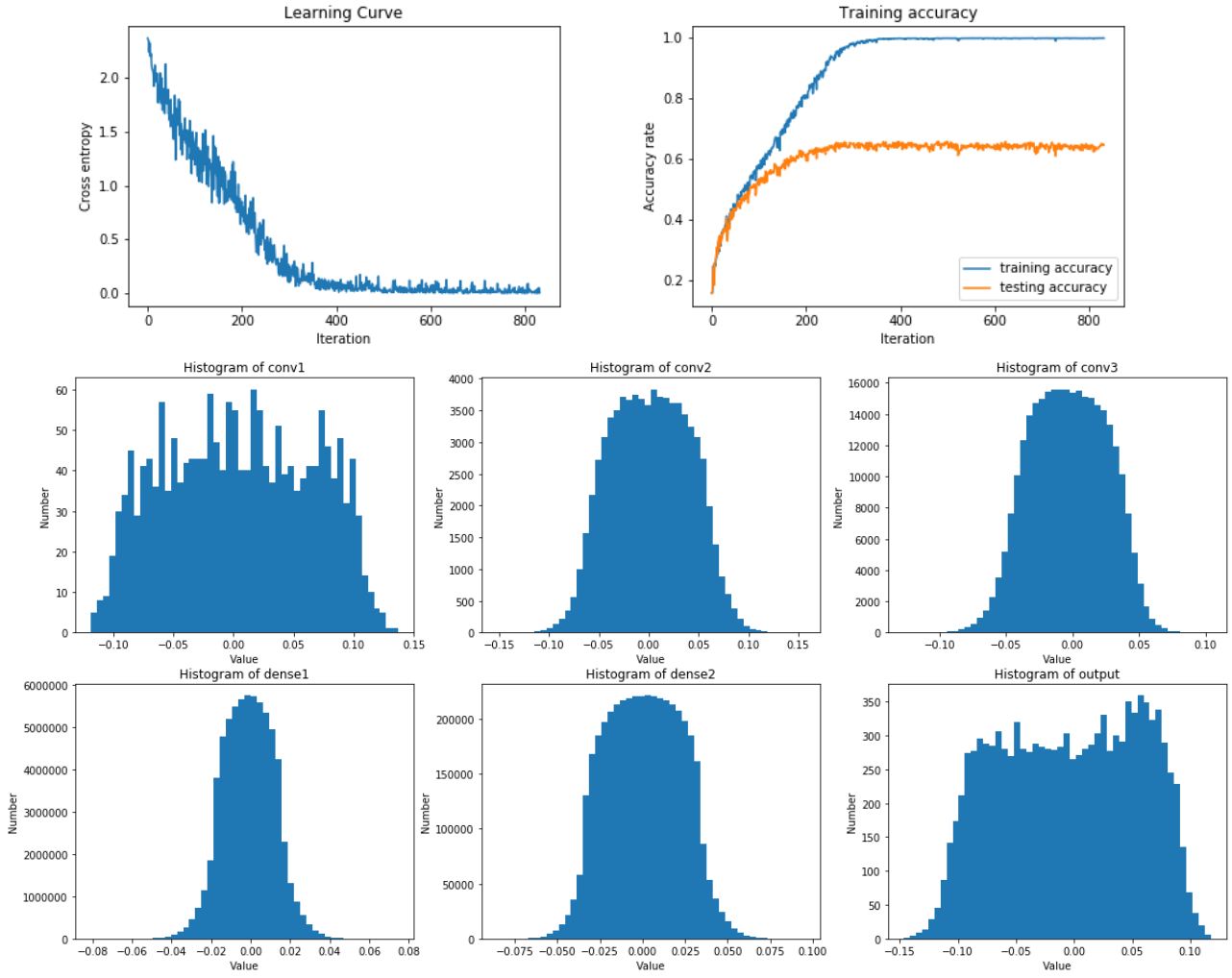
The concrete example food items in each category, and the number of images for each subset are listed below:

Here are the download page and related paper. You should preprocess the images such as resizing or cropping by yourself before implementation.

i. Please describe in details how to preprocess images because of the different resolution images in Food-11 dataset and explain why. You have to submit your preprocessing code.

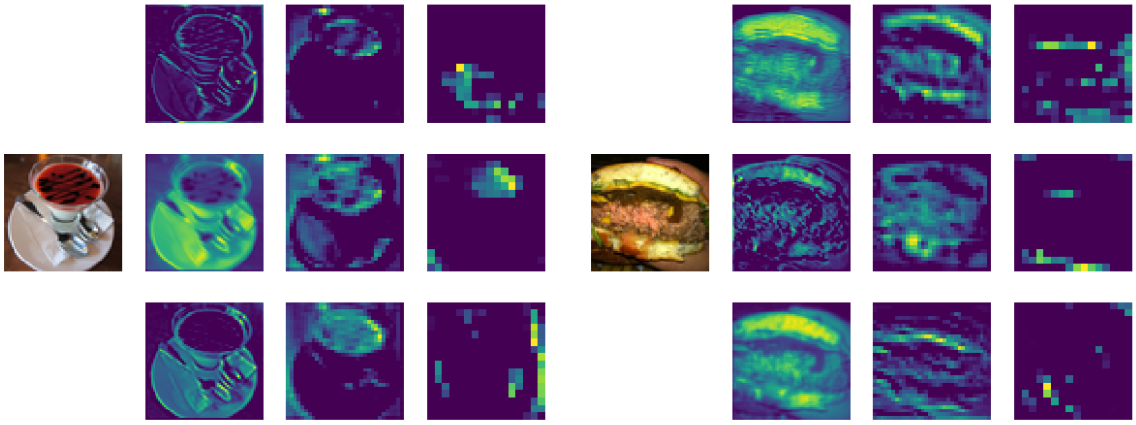
Category	Example items	Training	Validation	Evaluation
Bread	Bread, burger, pizza, pancakes, etc.	994	362	368
Dairy products	Milk, yogurt, cheese, butter, etc.	429	144	148
Dessert	Cakes, ice cream, cookies, chocolates, etc.	1500	500	500
Egg	Boiled and fried eggs, and omelette.	986	327	335
Fried food	French fries, spring rolls, fried calamari, etc.	848	326	287
Meat	Raw or cooked beef, pork, chicken, duck, etc.	1325	449	432
Noodles/Pasta	Flour/rice noodle, ramen, and spaghetti pasta.	440	147	147
Rice	Boiled and fried rice.	280	96	96
Seafood	Fish, shellfish, and shrimp; raw or cooked.	855	347	303
Soup	Various kinds of soup.	1500	500	500
Vegetable/Fruit	Fresh or cooked vegetables, salad, and fruits.	709	232	231
Total		9866	3430	3347

ii. Please implement a CNN for image recognition by using Food-11. You need to design the network architecture and analyze the effect of different settings including stride size and filter size. Plot learning curve, accuracy rate of training and test sets, and distribution of weights and biases.



iii. Show some examples of detected and undetected food images and discuss about your results.

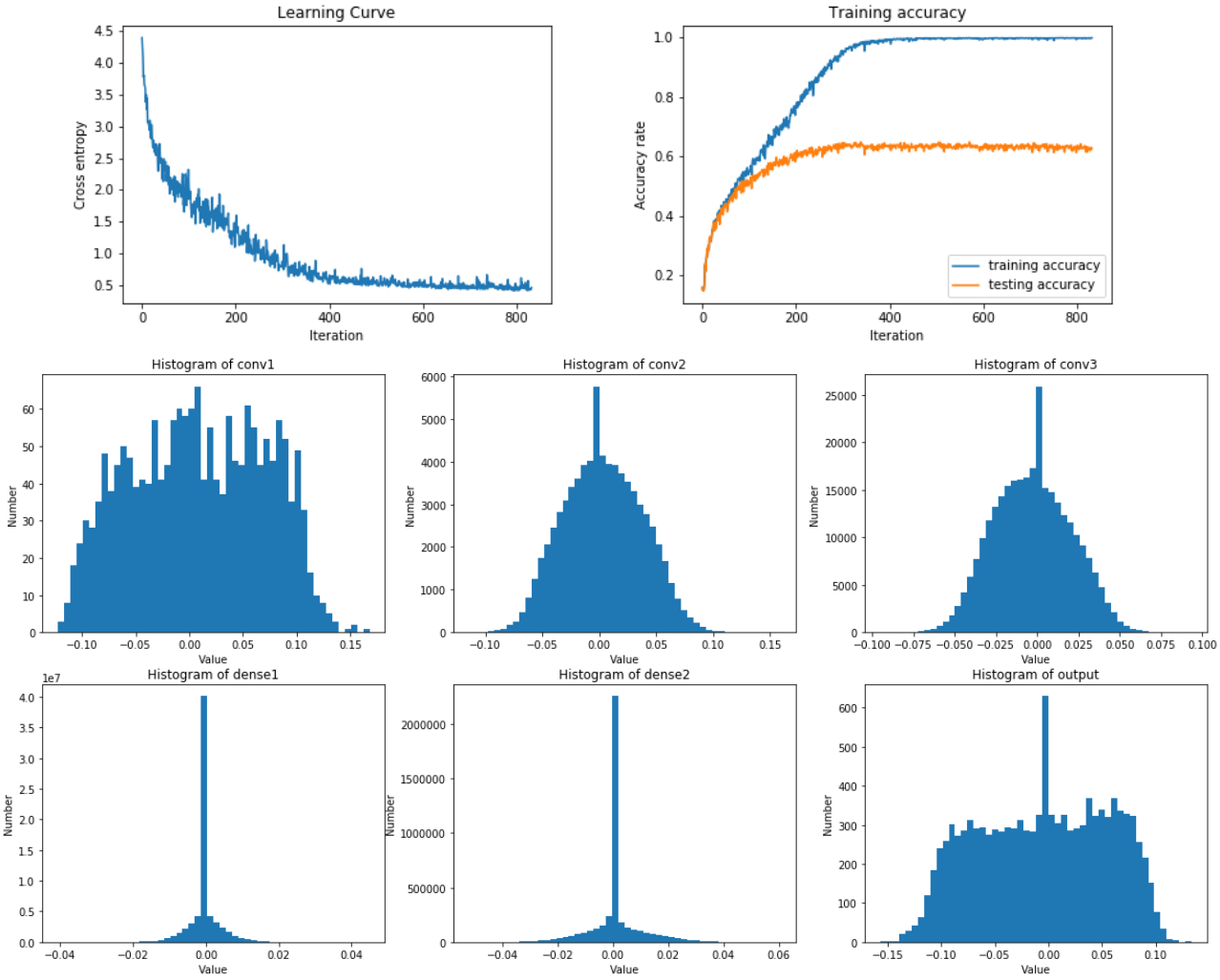
iv. Follow (iii). Show their feature maps in convolution layers and explain what you observe.



v. Follow (i). Train a CNN with L2 regularization:

$$E = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk} \ln t_{nk} + \alpha||\mathbf{w}||_2^2$$

Discuss about the difference between with and without regularization.



2 Recurrent Neural Network for Language Model

In this exercise, you will implement a Recurrent Neural Network (RNN) model for character-level language model using the complete works of William Shakespeare.
This dataset contains the plays and poetry written by English poet and actor William Shakespeare (1564-1616). The plays are divided into the genres of tragedy, history, and comedy. For more details, please refer to the URL http://shakespeare.mit.edu/works.html.
Dataset file shakespeare_train.txt contains most part of the plays, the rest are in the shakespeare_valid.txt. You are provided with the code data_utils.py to perform data preprocessing.

i. Please explain how to separate the sequence data into mini-batches in details.
Given the sequence data **s** = [N,C,T,U, ,i,s, ,g,o,o,d], what is the first mini-batch for batch size=2 and sequence length=3? Show your result and explanation.

ii. Please construct a character-level language model with standard RNNs for Shakespeare's works. For character-level language model, we aim to minimize the bits-per-character (BPC), which is defined as

$$\text{BPC} = -\frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{K} t_{t,k}\log_2 y_{t,k}(\mathbf{x}_t, \mathbf{w})$$

With a batch of input data, objective function is

$$\begin{aligned} E(\mathbf{w}) &= -\frac{1}{N}\sum_{n=1}^{N}\text{BPC}(\{\mathbf{x}_t^n\}_{t=1}^{T}) \\ &= -\frac{1}{NT}\sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{k=1}^{K} t_{t,k}\log_2 y_{t,k}^n(\mathbf{x}_t^n, \mathbf{w}) \end{aligned}$$

Minimize the objective function $E(\mathrm{w})$ by running the error backpropagation algorithm using the stochastic gradient descent

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta\nabla E(\mathbf{w}^{(\tau)})$$

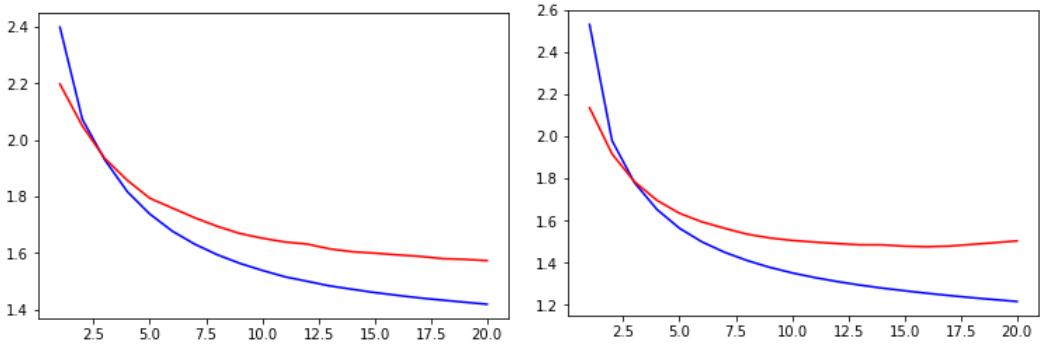
Design the network architecture by yourself, including number of hidden layers, number of hidden units, learning rate, number of iterations and mini-batch size. You have to show your (a) learning curve, (b) training error rate and (c) validation error rate in the report.

- In the first iteration for each epoch, initialize the hidden states to zero. Then use the final hidden states of the current minibatch as the initial hidden state of the subsequent minibatch (successive minibatches sequentially traverse the training set).
- Feel free to use any optimizer (SGD, Adadelta , ADAM and so on).
- You can implement the model by built-in functions, such as *tf.contrib.rnn*, *tf.nn.static_rnn*, *torch.nn.RNN* and so on.
- You will gain **BONUS point** if you implement the model by yourself

iii. Redo ii. with Long Short Term Memory networks (LSTMs), which is formulated by

$$
\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \\
\mathbf{g}_t &= \tanh(\mathbf{W}_g[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_g) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned}
$$

iv. Please discuss the difference between ii. and iii.



Learning curve of RNN (Left) and LSTM (Right) with 20 epoches. Red curve stands for Validation and blue curve stands for Training

v. Please generate some words by your model using either RNN or LSTM, you can prime the model with some starting text. This starts out the RNN with some hardcoded characters to warm it up with some context before it starts generating. For example,

PrimeText = [ROMEO]

ROMEO: What should I sand? what word is this anointed
To the place? I have spoke it in a straw.

PROTEUS:
A very villanous prince, a merceall hence.

LUCIANA:
He is outloved out of the wind of her.

LUCIANA:
What were thou call'd what I spake off me! Why, then, is
the discontent and mistress, and hither any things but
a story of true cruects, who hath a stoop, and
all my heads.

CAMILLO:
I did not send thee but a complexion in the watch;
the storm is not a thief-paties back to spoil of
the goose.