KTH Royal Institute of Technology
DD2424 Deep Learning in Data Science

Assignment 3

Yage Hao (yage@kth.se)

24/05/2021

1 Introduction

In this assignment we trained a two-layer ConvNet to predict the language of a surname from its spelling.

2 Code Explanation

We followed the instruction and went through Assignment 3 (also some detailed instructions in Assignment 2) document to implemented the work. The code contains two main classes, i.e. "DataProcess" and "ConvNet".

In detail, "DataProcess" class is used to preparing data. It mainly includes the following two important functions:

- read_file: read in .txt data files.
- create_one_hot: construct one-hot encoding.

The class "ConvNet" includes more typical functional parts, such as:

- the ones used to set hyper-parameters and initialize the ConvNet's parameters: initialize.
- the one used to construct the convolution matrices: MFMatrix, MXMatrix.
- the ones used to apply mini-batch gradient descent with momentum: evaluateClassifier, computeCost, computeAccuracy, evaluation_and_compute_gradients.

3 Results

3.1 Gradient check

The check is done by computing relative error. We compared the analytically and numerically computed results of gradients with respective to weights, filter 1 and filter 2 obtained through function 'evaluation_and_compute_gradients' and 'computeGradientsNum' respectively. We assert that they provide the same values in level of 10^{-5}. Detailed results are provided in Table 1.

gradient	Filter1	Filter2	Weights
differences	7.8164e-07	4.4027e-03	8.2768e-07
relative error	2.2342e-09	2.0117e-05	1.9401e-09

Table 1: Gradient check.

3.2 Compensate for the imbalanced dataset

The original dataset contains 18 classes in which the largest class has size 9296 and the smallest class has size 39. To avoid ConvNet only to learn to classify the dominating classes correctly, we compensate for the imbalanced dataset by sampling the same number of examples as the smallest class.

The following validation loss plots and final confusion matrices for cases with and without compensating are computed with hyper-parameter setting: $n_1 = 20, n_2 = 20, k_1 = 5, k_2 = 3, eta = 0.01, rho = 0.9, batch\_size = 100, n\_ite = 2000$.

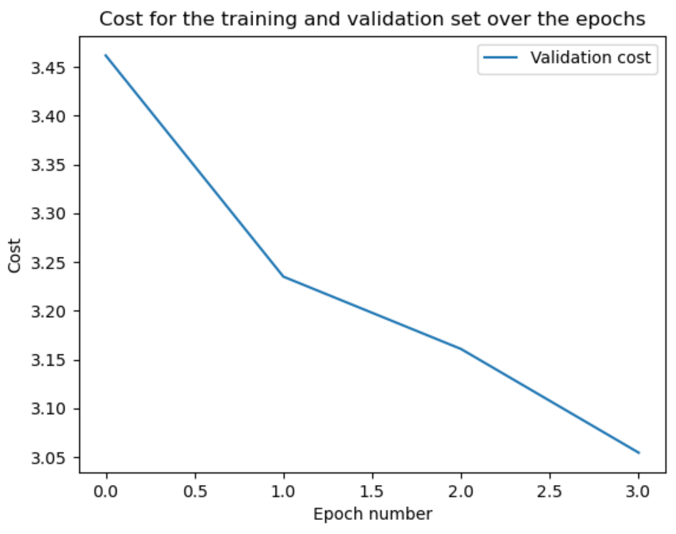
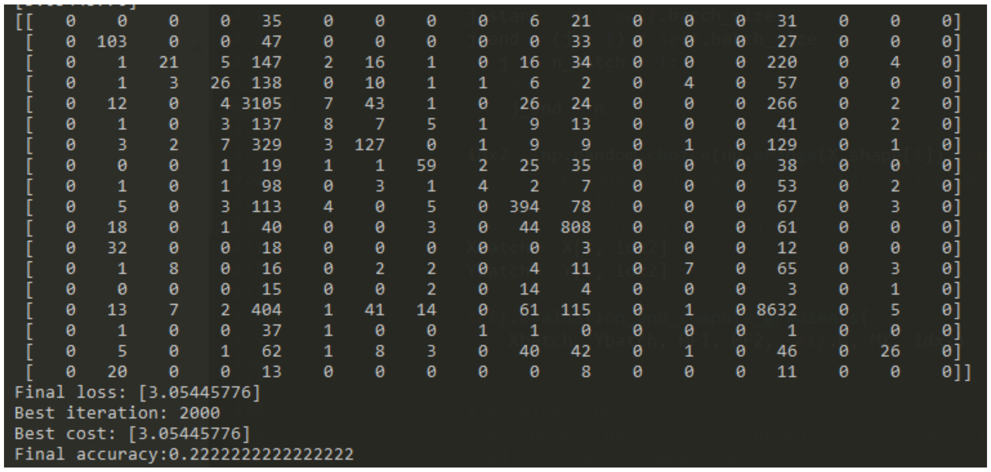


Figure 1: Validation loss plot without compensating training.



```
[[   0    0    0    0   35    0    0    0    0    6   21    0    0    0   31    0    0    0]
 [   0  103    0    0   47    0    0    0    0    0   33    0    0    0   27    0    0    0]
 [   0    1   21    5  147    2   16    1    0   16   34    0    0    0  220    0    4    0]
 [   0    1    3   26  138    0   10    1    1    6    2    0    4    0   57    0    0    0]
 [   0   12    0    4 3105    7   43    1    0   26   24    0    0    0  266    0    2    0]
 [   0    1    0    3  137    8    7    5    1    9   13    0    0    0   41    0    2    0]
 [   0    3    2    7  329    3  127    0    1    9    9    0    1    0  129    0    1    0]
 [   0    0    0    1   19    1    1   59    2   25   35    0    0    0   38    0    0    0]
 [   0    1    0    1   98    0    3    1    4    2    7    0    0    0   53    0    2    0]
 [   0    5    0    3  113    4    0    5    0  394   78    0    0    0   67    0    3    0]
 [   0   18    0    1   40    0    0    3    0   44  808    0    0    0   61    0    0    0]
 [   0   32    0    0   18    0    0    0    0    0    3    0    0    0   12    0    0    0]
 [   0    1    8    0   16    0    2    2    0    4   11    0    7    0   65    0    3    0]
 [   0    0    0    0   15    0    0    2    0   14    4    0    0    0    3    0    1    0]
 [   0   13    7    2  404    1   41   14    0   61  115    0    1    0 8632    0    5    0]
 [   0    1    0    0   37    1    0    0    1    1    0    0    0    0    1    0    0    0]
 [   0    5    0    1   62    1    8    3    0   40   42    0    1    0   46    0   26    0]
 [   0   20    0    0   13    0    0    0    0    0    8    0    0    0   11    0    0    0]]
Final loss: [3.05445776]
Best iteration: 2000
Best cost: [3.05445776]
Final accuracy:0.2222222222222222
```

Figure 2: Confusion matrix without compensating training.

Accuracy for non-compensated dataset is about 0.22. By observing the confusion matrix, we find that there are many zero entries indicating that the non-dominating classes are poorly predicted.

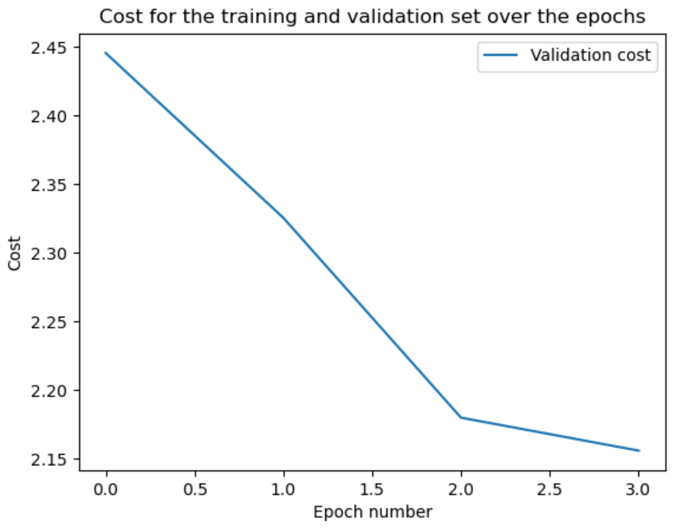
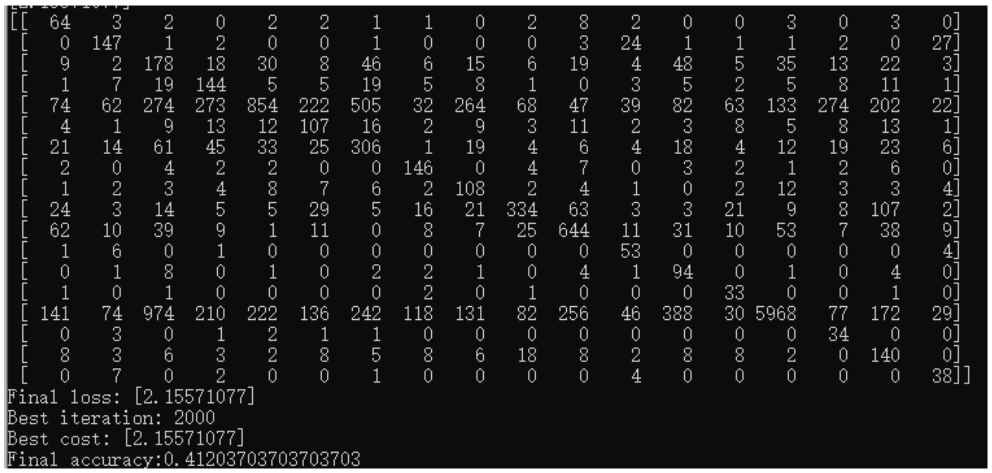


Figure 3: Validation loss plot with compensating training.



```
[[  64    3    2    0    2    2    1    1    0    2    8    2    0    0    3    0    3    0]
 [   0  147    1    2    0    0    1    0    0    0    3   24    1    1    1    2    0   27]
 [   9    2  178   18   30    8   46    6   15    6   19    4   48    5   35   13   22    3]
 [   1    7   19  144    5    5   19    5    8    1    0    3    5    2    5    8   11    1]
 [  74   62  274  273  854  222  505   32  264   68   47   39   82   63  133  274  202   22]
 [   4    1    9   13   12  107   16    2    9    3   11    2    3    8    5    8   13    1]
 [  21   14   61   45   33   25  306    1   19    4    6    4   18    4   12   19   23    6]
 [   2    0    4    2    2    0    0  146    0    4    7    0    3    2    1    2    6    0]
 [   1    2    3    4    8    7    6    2  108    2    4    1    0    2   12    3    3    4]
 [  24    3   14    5    5   29    5   16   21  334   63    3    3   21    9    8  107    2]
 [  62   10   39    9    1   11    0    8    7   25  644   11   31   10   53    7   38    9]
 [   1    6    0    1    0    0    0    0    0    0    0   53    0    0    0    0    0    4]
 [   0    1    8    0    1    0    2    2    1    0    4    1   94    0    1    0    4    0]
 [   1    0    1    0    0    0    0    2    0    1    0    0    0   33    0    0    1    0]
 [ 141   74  974  210  222  136  242  118  131   82  256   46  388   30 5968   77  172   29]
 [   0    3    0    1    2    1    1    0    0    0    0    0    0    0    0   34    0    0]
 [   8    3    6    3    2    8    5    8    6   18    8    2    8    8    2    0  140    0]
 [   0    7    0    2    0    0    1    0    0    0    0    4    0    0    0    0    0   38]]
Final loss: [2.15571077]
Best iteration: 2000
Best cost: [2.15571077]
Final accuracy:0.41203703703703703
```

Figure 4: Confusion matrix with compensating training.

Accuracy for compensated dataset is about 0.41. Compared with the non-compensated one, the compensated dataset shows a better performance not only on dominating classes but on the other smaller-sized classes as well.

However, the accuracy differences between the compensated and non-compensated datasets are not as large as we hypothesised. We analyzed that this may because that the dominating classes have too much more data than the others. Therefore, even the other small-sized classes got all predictions wrong, the general performance may still looks reasonable.

3.3 Best performing ConvNet

To get the best performing ConvNet, we set up parameters as the following: $n_1 = 50, n_2 = 50, k_1 = 5, k_2 = 3, eta = 0.01, rho = 0.9, batch\_size = 100, n\_ite = 2000$. These parameters are intuited by the previous two assignments. We applied these settings to the compensated balenced dataset and increasing number of iterates until we find the turning point on validation loss curve.

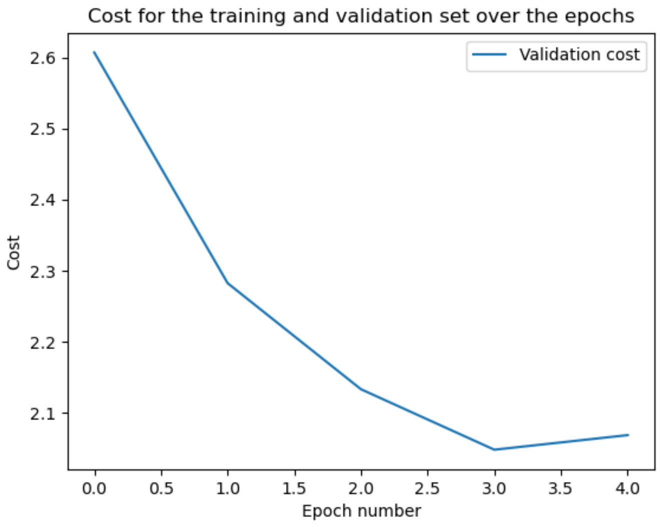
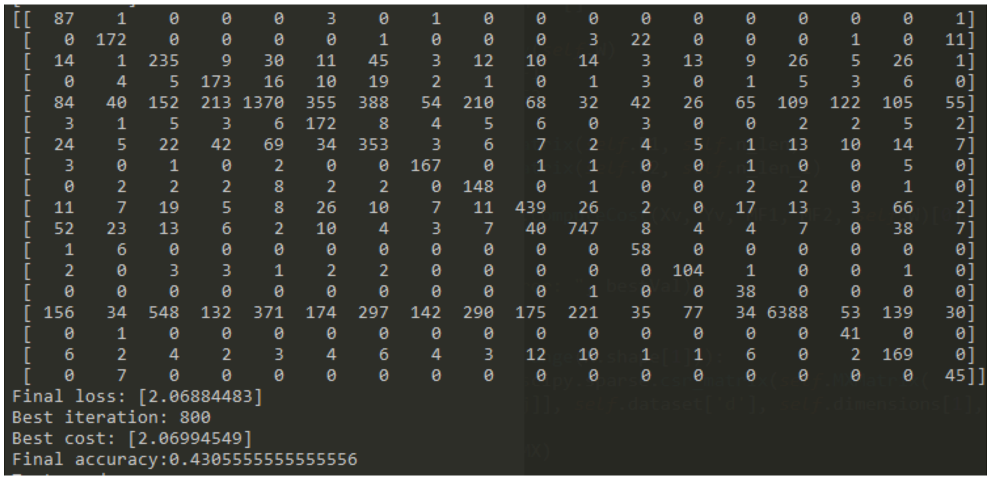


Figure 5: Validation loss plot of the best performing ConvNet.

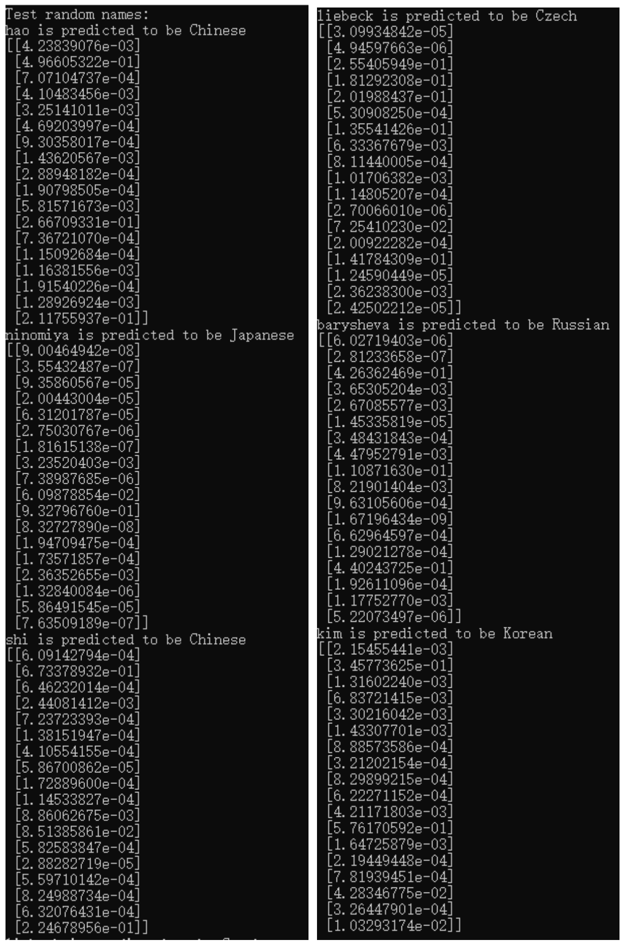


```
[[  87    1    0    0    0    3    0    1    0    0    0    0    0    0    0    0    0    1]
 [   0  172    0    0    0    0    1    0    0    0    3   22    0    0    0    1    0   11]
 [  14    1  235    9   30   11   45    3   12   10   14    3   13    9   26    5   26    1]
 [   0    4    5  173   16   10   19    2    1    0    1    3    0    1    5    3    6    0]
 [  84   40  152  213 1370  355  388   54  210   68   32   42   26   65  109  122  105   55]
 [   3    1    5    3    6  172    8    4    5    6    0    3    0    0    2    2    5    2]
 [  24    5   22   42   69   34  353    3    6    7    2    4    5    1   13   10   14    7]
 [   3    0    1    0    2    0    0  167    0    1    1    0    0    1    0    0    5    0]
 [   0    2    2    2    8    2    2    0  148    0    1    0    0    2    2    0    1    0]
 [  11    7   19    5    8   26   10    7   11  439   26    2    0   17   13    3   66    2]
 [  52   23   13    6    2   10    4    3    7   40  747    8    4    4    7    0   38    7]
 [   1    6    0    0    0    0    0    0    0    0    0   58    0    0    0    0    0    0]
 [   2    0    3    3    1    2    2    0    0    0    0    0  104    1    0    0    1    0]
 [   0    0    0    0    0    0    0    0    0    0    1    0    0   38    0    0    0    0]
 [ 156   34  548  132  371  174  297  142  290  175  221   35   77   34 6388   53  139   30]
 [   0    1    0    0    0    0    0    0    0    0    0    0    0    0    0   41    0    0]
 [   6    2    4    2    3    4    6    4    3   12   10    1    1    6    0    2  169    0]
 [   0    7    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0   45]]
Final loss: [2.06884483]
Best iteration: 800
Best cost: [2.06994549]
Final accuracy:0.4305555555555556
```

Figure 6: Confusion matrix of the best performing ConvNet.

3.4 Test

To test whether the best performing ConvNet works, we create a test set using five of my friends' surnames and mine. They are hao, ninomiya, shi, liebeck, barysheva, kim and their corresponding nationalities are Chinese, Japanese, Chinese, English, Russian, Korean. The following figures are the output predictions of these six names.



```
Test random names:
hao is predicted to be Chinese
[[4.23839076e-03]
 [4.96605322e-01]
 [7.07104737e-04]
 [4.10483456e-03]
 [3.25141011e-03]
 [4.69203997e-04]
 [9.30358017e-04]
 [1.43620567e-03]
 [2.88948182e-04]
 [1.90798505e-04]
 [5.81571673e-03]
 [2.66709331e-01]
 [7.36721070e-04]
 [1.15092684e-04]
 [1.16381556e-03]
 [1.91540226e-04]
 [1.28926924e-03]
 [2.11755937e-01]]
ninomiya is predicted to be Japanese
[[9.00464942e-08]
 [3.55432487e-07]
 [9.35860567e-05]
 [2.00443004e-05]
 [6.31201787e-05]
 [2.75030767e-06]
 [1.81615138e-07]
 [3.23520403e-03]
 [7.38987685e-06]
 [6.09878854e-02]
 [9.32796760e-01]
 [8.32727890e-08]
 [1.94709475e-04]
 [1.73571857e-04]
 [2.36352655e-03]
 [1.32840084e-06]
 [5.86491545e-05]
 [7.63509189e-07]]
shi is predicted to be Chinese
[[6.09142794e-04]
 [6.73378932e-01]
 [6.46232014e-04]
 [2.44081412e-03]
 [7.23723393e-04]
 [1.38151947e-04]
 [4.10554155e-04]
 [5.86700862e-05]
 [1.72889600e-04]
 [1.14533827e-04]
 [8.86062675e-03]
 [8.51385861e-02]
 [5.82583847e-04]
 [2.88282719e-05]
 [5.59710142e-04]
 [8.24988734e-04]
 [6.32076431e-04]
 [2.24678956e-01]]
```

```
liebeck is predicted to be Czech
[[3.09934842e-05]
 [4.94597663e-06]
 [2.55405949e-01]
 [1.81292308e-01]
 [2.01988437e-01]
 [5.30908250e-04]
 [1.35541426e-01]
 [6.33367679e-03]
 [8.11440005e-04]
 [1.01706382e-03]
 [1.14805207e-04]
 [2.70066010e-06]
 [7.25410230e-02]
 [2.00922282e-04]
 [1.41784309e-01]
 [1.24590449e-05]
 [2.36238300e-03]
 [2.42502212e-05]]
barysheva is predicted to be Russian
[[6.02719403e-06]
 [2.81233658e-07]
 [4.26362469e-01]
 [3.65305204e-03]
 [2.67085577e-03]
 [1.45335819e-05]
 [3.48431843e-04]
 [4.47952791e-03]
 [1.10871630e-01]
 [8.21901404e-03]
 [9.63105606e-04]
 [1.67196434e-09]
 [6.62964597e-04]
 [1.29021278e-04]
 [4.40243725e-01]
 [1.92611096e-04]
 [1.17752770e-03]
 [5.22073497e-06]]
kim is predicted to be Korean
[[2.15455441e-03]
 [3.45773625e-01]
 [1.31602240e-03]
 [6.83721415e-03]
 [3.30216042e-03]
 [1.43307701e-03]
 [8.88573586e-04]
 [3.21202154e-04]
 [8.29899215e-04]
 [6.22271152e-04]
 [4.21171803e-03]
 [5.76170592e-01]
 [1.64725879e-03]
 [2.19449448e-04]
 [7.81939451e-04]
 [4.28346775e-02]
 [3.26447901e-04]
 [1.03293174e-02]]
```

Figure 7: Prediction results.

We find that our best performing ConvNet predicts quite satisfying on our own test set, that only surname 'liebeck' is falsely predicted. It is reasonable since surnames of Chinese, Japanese, Korean and Russian are quite different intuitively. I also tried to input surname which does not included in any of the 18 classes, for example, 'hector' which is actually Swedish is categorized as Germany, indicating for those languages with the same origin, surnames in those countries may be quite similar in some extent.