Lab-Assignment#2

Text & Web Mining (H6751)

Instructor: WKW SCI Jin Cheon Na, NTU

Huang Fan - September 20, 2020

Classifying Surnames with a Multilayer Perceptron (week6)

Requirement

1. Build the best model (based on test loss and test accuracy) by exploring following options:

1) learning_rate

2) batch_size

3) dropout (use only if it helps)

4) batch norm (use only if it helps)

5) weight_decay (L2 regularisation) (use only if it helps)

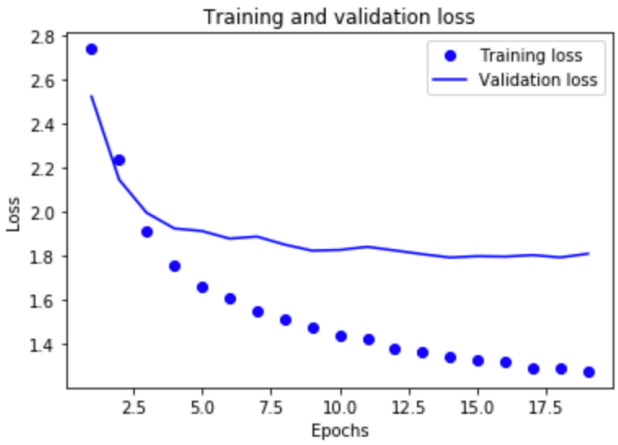
6) hidden_dim

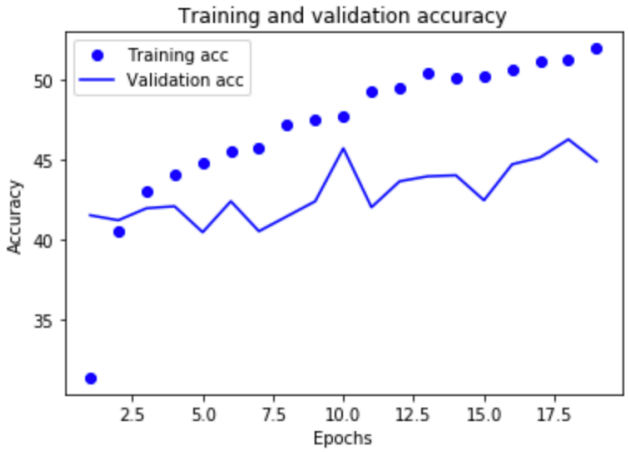
Note that it is not necessary to adjust other parameter values even though you are allowed to do so.

2. Submit one zip file, named lab-no2-yourname.zip, that contains your one Jupyter Notebook file and data files (e.g., input data and model files) through Turnitin on the class website.

3. The Jupyter notebook file must show all output results of your solution code. So please make sure that you run all the cells in the notebook file before your submission. Also note that Turnitin does not allow you to resubmit your lab assignment file.

Baseline





Curabitur vulputate viverra pede

Test loss: 1.7909419298171998;

Test Accuracy: 45.18749999999999

=====

'Irish'

==========

drewer -> English (p=0.47)

drewer -> German (p=0.25)

drewer -> Dutch (p=0.12)

drewer -> Polish (p=0.07)

drewer -> Czech (p=0.04)

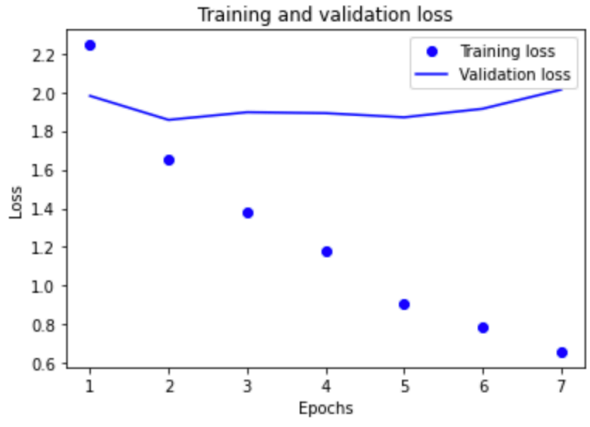
learning_rate = 0.001

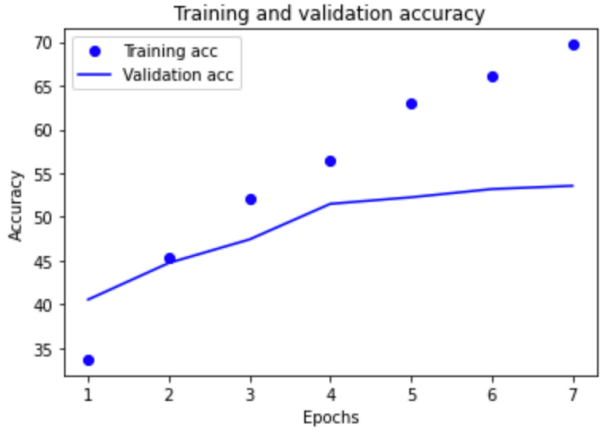
batch_size = 64

hidden_dim = 300

Tests

1. Change the value of learning_rate (0.001) to 0.01 or 0.0001



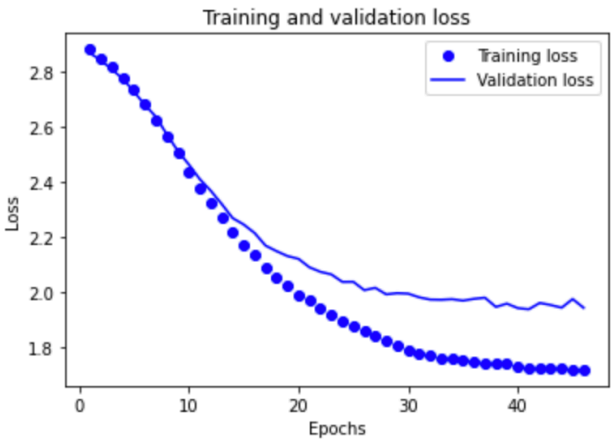


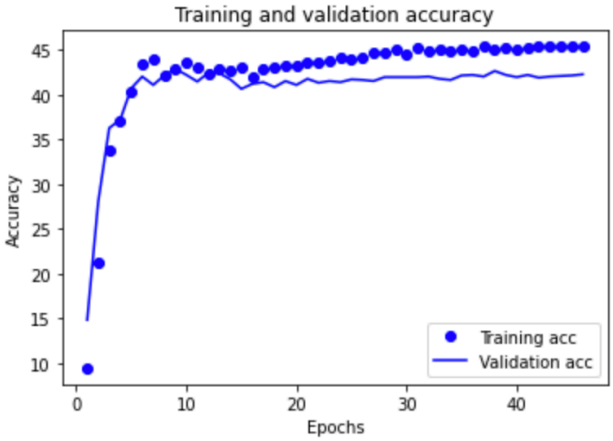
Test loss: 1.8986661529541016;
Test Accuracy: 45.12499999999999
==========
drewer -> German (p=0.48)
drewer -> Dutch (p=0.24)
drewer -> English (p=0.19)
drewer -> Polish (p=0.03)
drewer -> Scottish (p=0.03)

learning_rate = 0.01

batch_size = 64

hidden_dim = 300





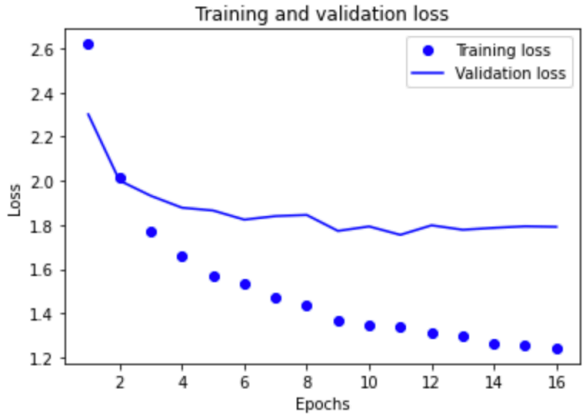
Test loss: 1.9229955101013183;
Test Accuracy: 42.56249999999999
==========
drewer -> English (p=0.27)
drewer -> German (p=0.22)
drewer -> Dutch (p=0.18)
drewer -> Polish (p=0.08)
drewer -> Czech (p=0.04)

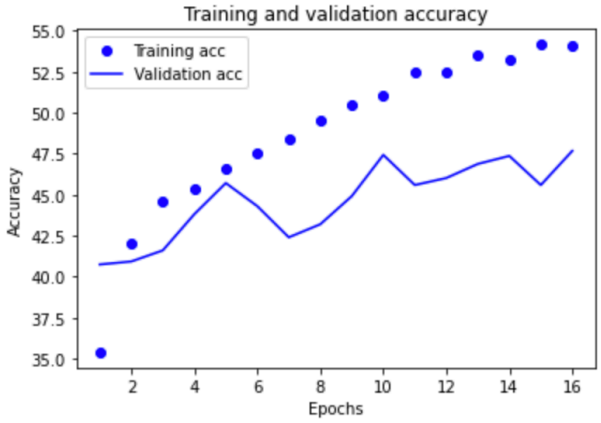
learning_rate = 0.0001

batch_size = 64

hidden_dim = 300

2. Change batch_size, such as 32 or 128



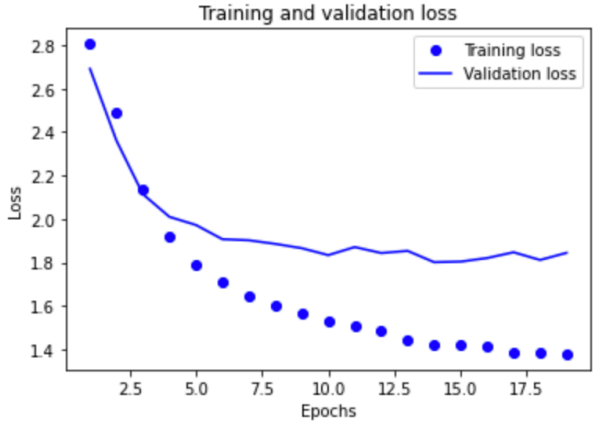


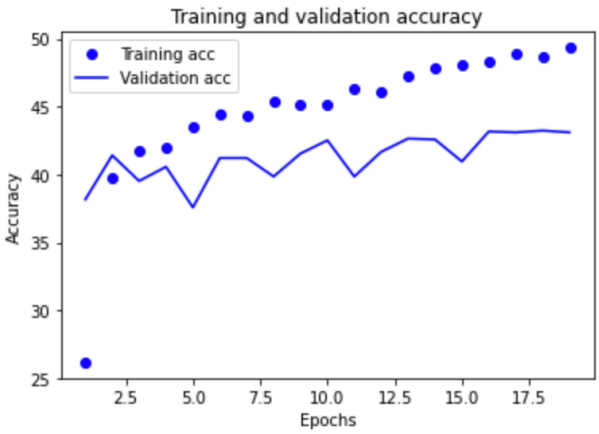
Test loss: 1.7946544348024855;
Test Accuracy: 45.52696078431372
==========
drewer -> English (p=0.49)
drewer -> German (p=0.23)
drewer -> Dutch (p=0.13)
drewer -> Polish (p=0.06)
drewer -> Czech (p=0.05)

learning_rate = 0.001

batch_size = 32

hidden_dim = 300



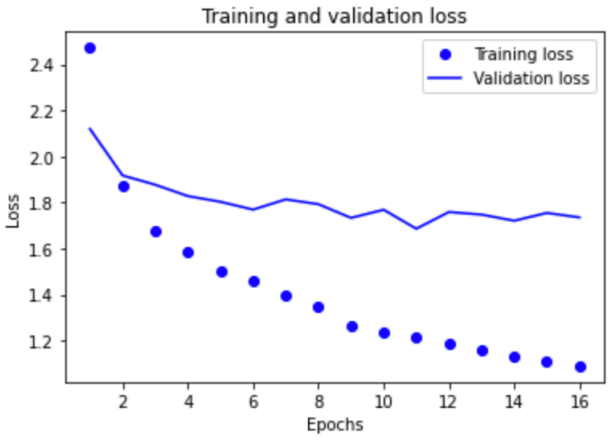


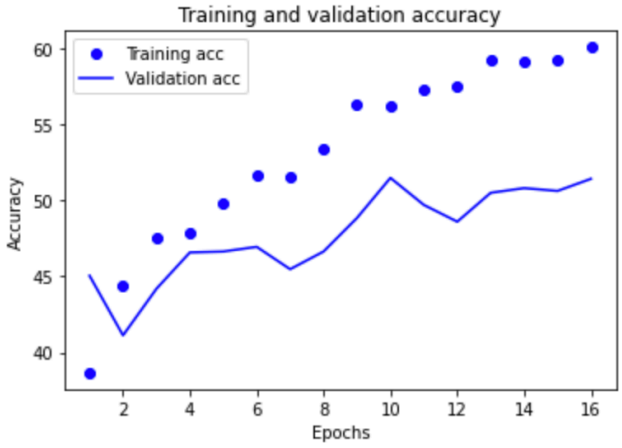
Test loss: 1.829984794060389;
Test Accuracy: 43.359375
==========
drewer -> English (p=0.45)
drewer -> German (p=0.25)
drewer -> Dutch (p=0.13)
drewer -> Polish (p=0.08)
drewer -> Czech (p=0.04)

learning_rate = 0.001

batch_size = 128

hidden_dim = 300

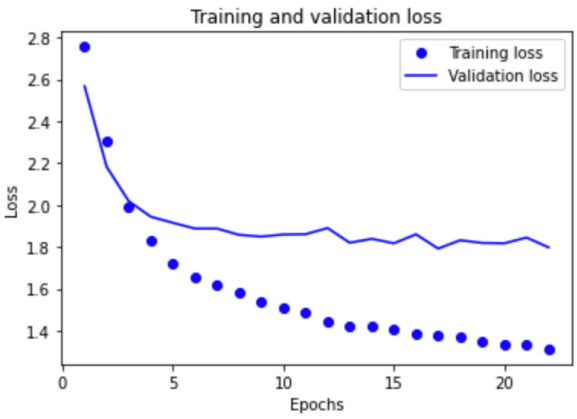


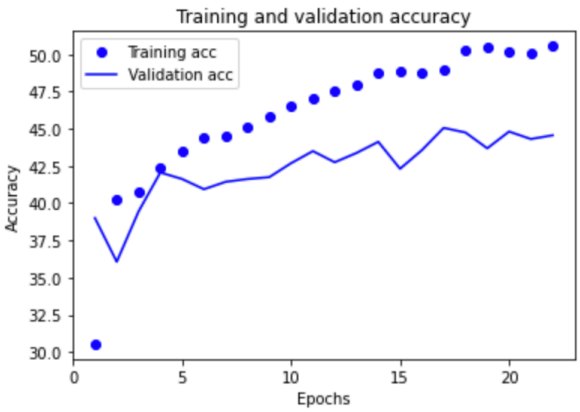


Test loss: 1.7529428664920401;
Test Accuracy: 50.30339805825243
Top 5 predictions:
==========
drewer -> English (p=0.47)
drewer -> German (p=0.25)
drewer -> Dutch (p=0.15)
drewer -> Czech (p=0.06)
drewer -> Polish (p=0.04)

learning_rate = 0.001
batch_size = 16
hidden_dim = 300

3. Try out dropout with p=0.2



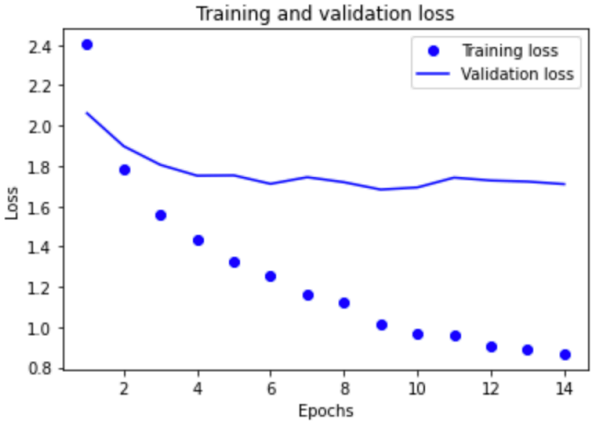


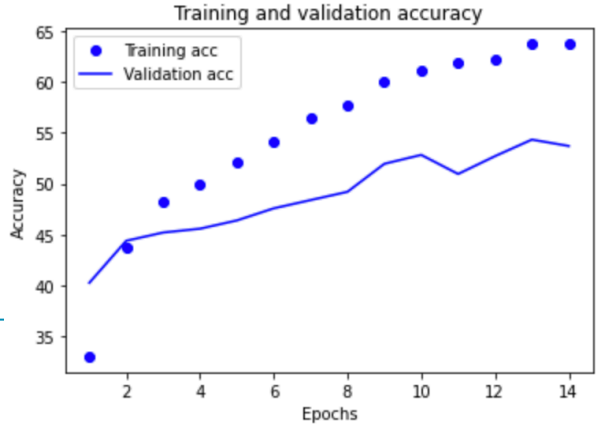
Test

loss: 1.8367971992492675;
Test Accuracy: 46.0
==========
drewer -> English (p=0.43)
drewer -> German (p=0.26)
drewer -> Dutch (p=0.17)
drewer -> Czech (p=0.05)
drewer -> Polish (p=0.04)

learning_rate = 0.001
batch_size = 64
p = 0.2
hidden_dim = 300

4. Try out batch norm



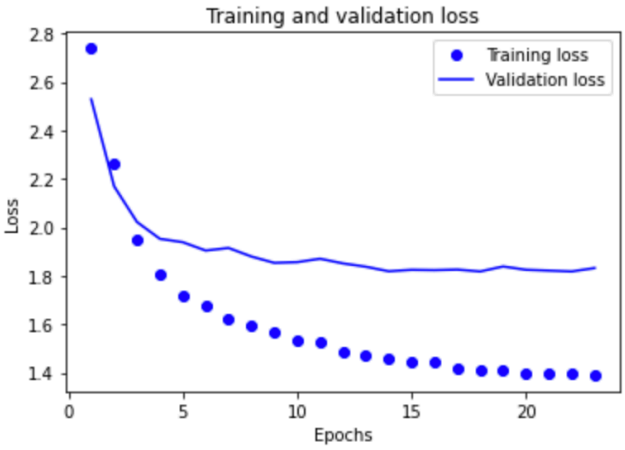


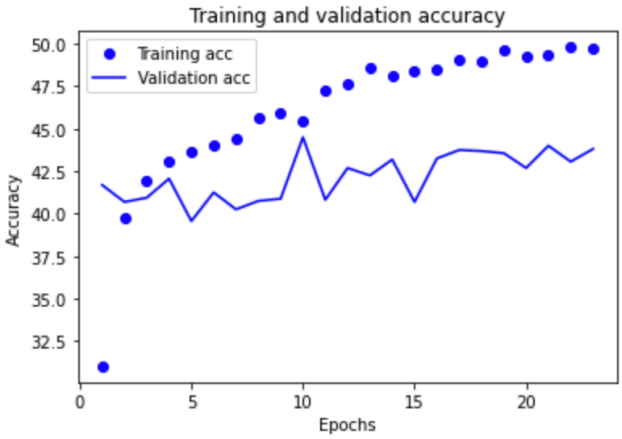
Test loss: 1.7861554527282715;
Test Accuracy: 53.37500000000001
==========
drewer -> English (p=0.56)
drewer -> German (p=0.23)
drewer -> Dutch (p=0.11)
drewer -> Polish (p=0.04)
drewer -> Czech (p=0.03)

learning_rate = 0.001
batch_size = 64
batch norm
hidden_dim = 300

Conclusion: Using batch norm would help to reduce loss and improve accuracy.

5. Try out weight_decay (L2 regularization)

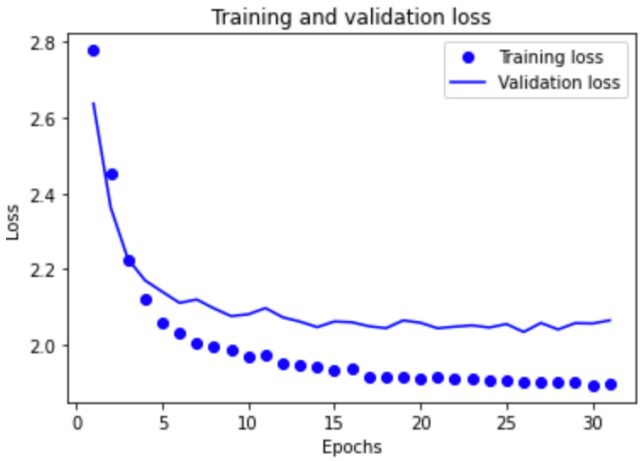


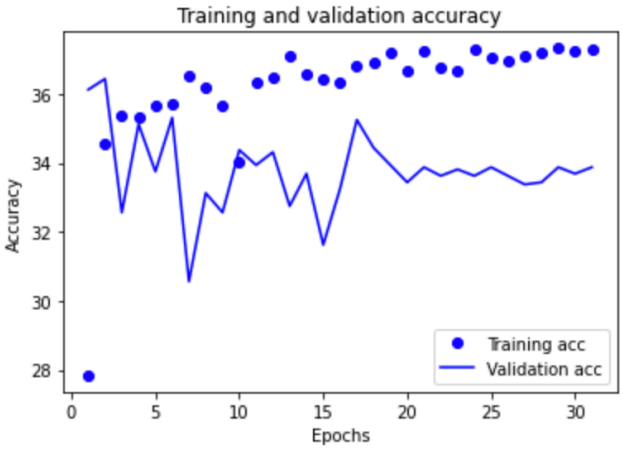


Test loss: 1.8105071878433225;
Test Accuracy: 46.0625
==========
drewer -> English (p=0.41)
drewer -> German (p=0.26)
drewer -> Dutch (p=0.16)
drewer -> Polish (p=0.08)
drewer -> Czech (p=0.04)

learning_rate = 0.001
batch_size = 64
weight_decay = 0.001

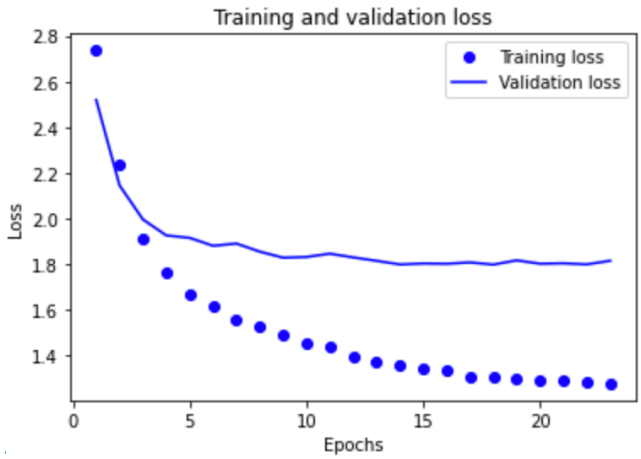
hidden_dim = 300

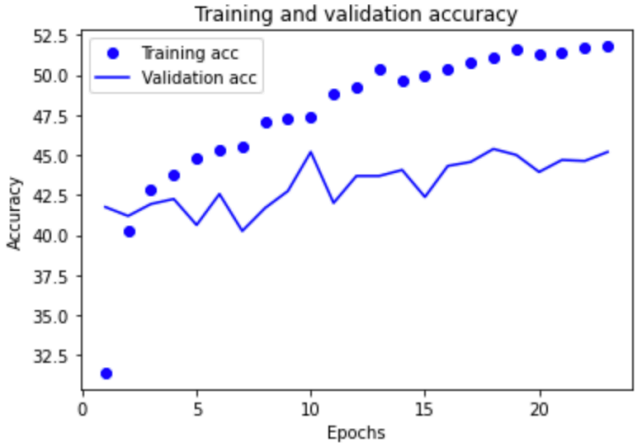




Test loss: 2.0312746429443362;
Test Accuracy: 33.125
==========
drewer -> German (p=0.27)
drewer -> Dutch (p=0.20)
drewer -> English (p=0.10)
drewer -> Polish (p=0.07)
drewer -> Czech (p=0.07)

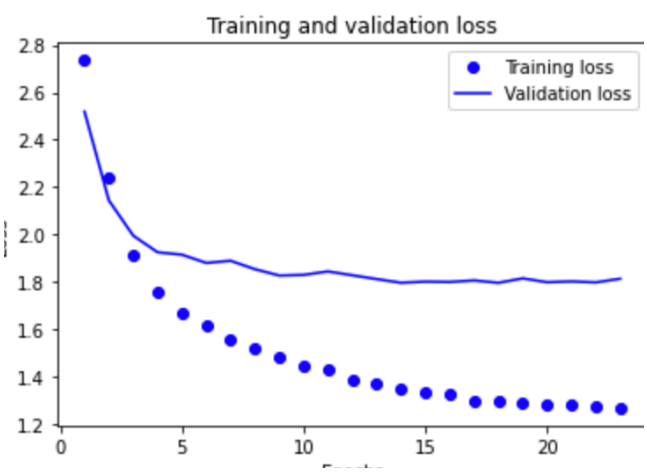
learning_rate = 0.001
batch_size = 64
weight_decay = 0.01
hidden_dim = 300

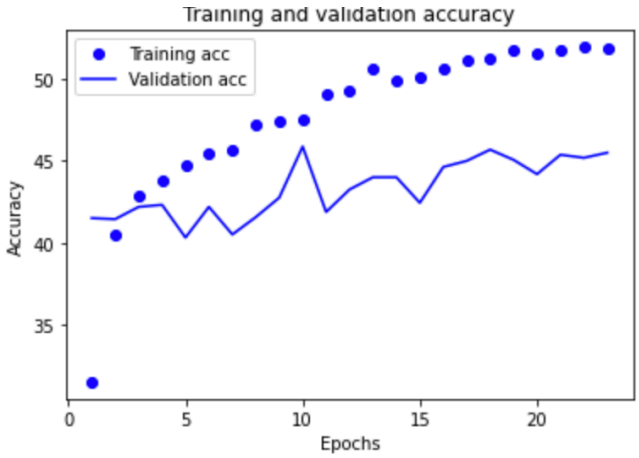




Test loss: 1.7989911818504334;
Test Accuracy: 46.93750000000001
==========
drewer -> English (p=0.49)
drewer -> German (p=0.22)
drewer -> Dutch (p=0.15)
drewer -> Polish (p=0.07)
drewer -> Czech (p=0.04)

learning_rate = 0.001
batch_size = 64
weight_decay = 0.0001
hidden_dim = 300



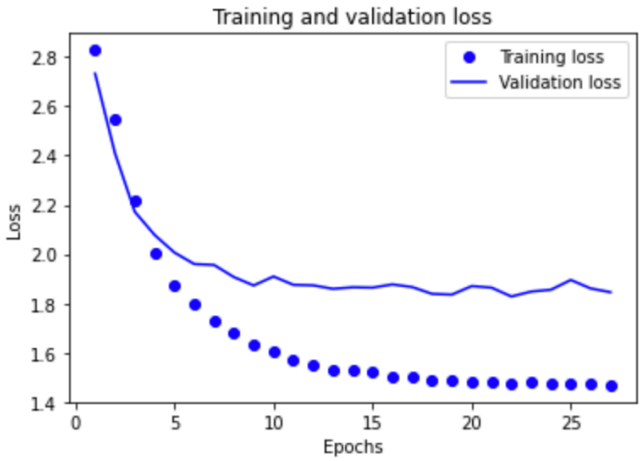


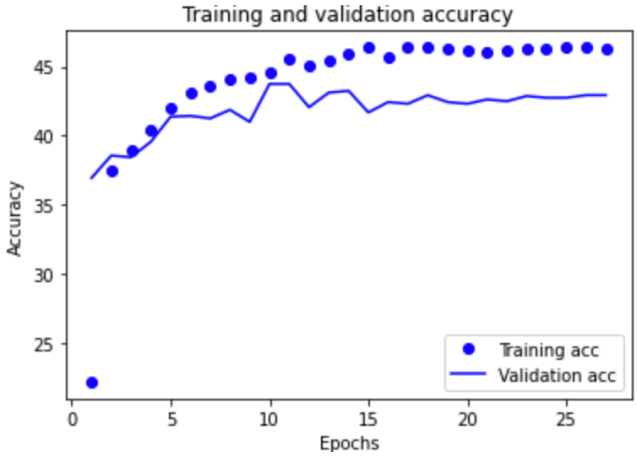
Test loss: 1.8016564869880678;
Test Accuracy: 46.37499999999999
==========
drewer -> English (p=0.50)
drewer -> German (p=0.21)
drewer -> Dutch (p=0.15)
drewer -> Polish (p=0.06)
drewer -> Czech (p=0.04)

learning_rate = 0.001
batch_size = 64
weight_decay = 0.00001
hidden_dim = 300

Conclusion: Using weight decay would increase both loss and accuracy (when equals 0.001). And when weight decay equals 0.0001, the loss increases a little and accuracy increases relatively more.

6. hidden_dim

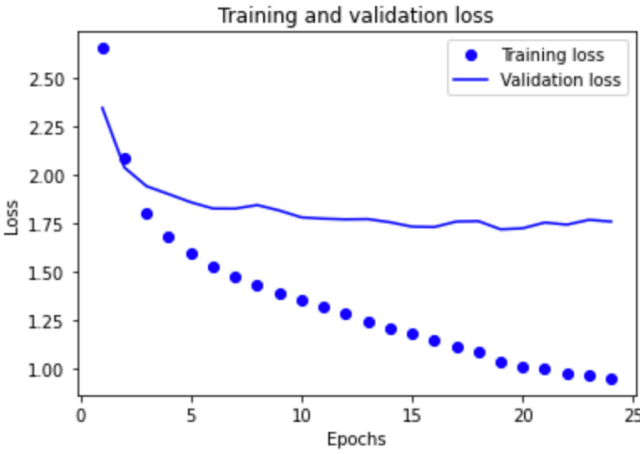


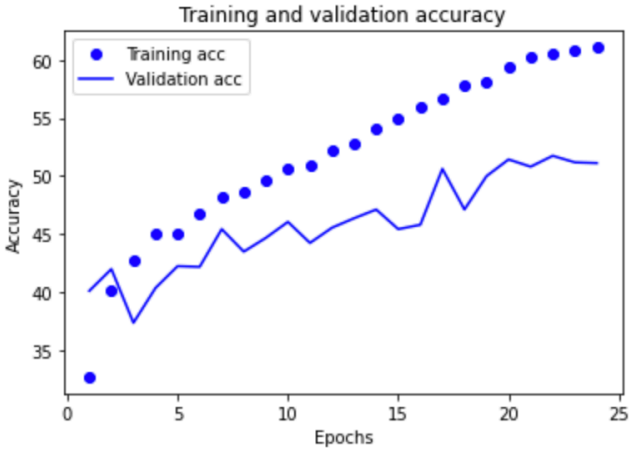


Test loss: 1.8291779136657715;
Test Accuracy: 42.0
==========
drewer -> English (p=0.37)
drewer -> German (p=0.29)
drewer -> Dutch (p=0.15)
drewer -> Polish (p=0.08)
drewer -> Czech (p=0.04)

learning_rate = 0.001
batch_size = 64
hidden_dim = 100

Test loss: 1.7758802032470704;





Test Accuracy: 51.12500000000001

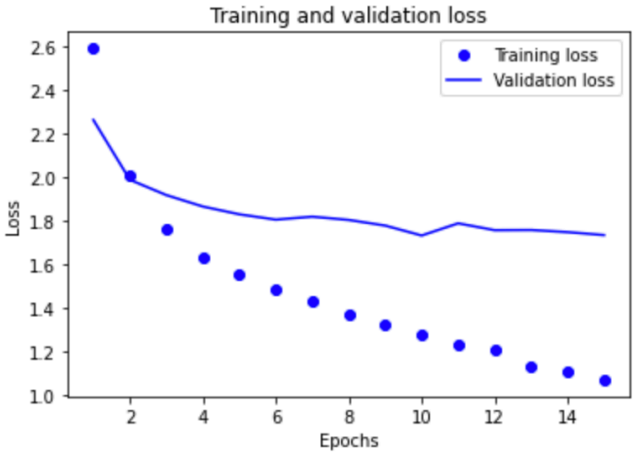
==========

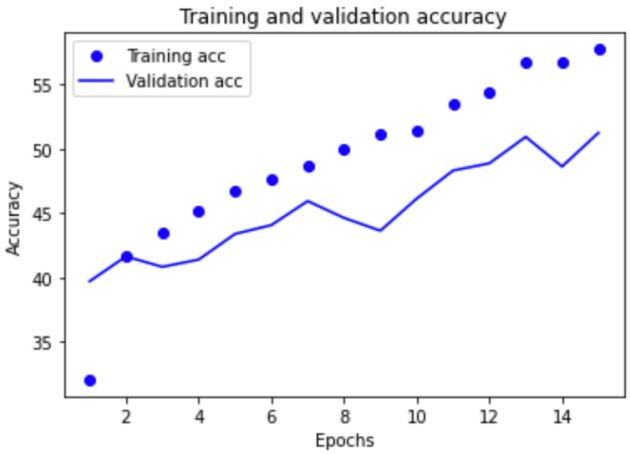
drewer -> English (p=0.39)
drewer -> German (p=0.36)
drewer -> Dutch (p=0.14)
drewer -> Polish (p=0.05)
drewer -> Czech (p=0.03)

learning_rate = 0.001

batch_size = 64

hidden_dim = 500





Test loss: 1.7670141363143919;
Test Accuracy: 47.3125

==========

drewer -> English (p=0.46)
drewer -> German (p=0.26)

drewer -> Dutch (p=0.16)
drewer -> Polish (p=0.05)
drewer -> Czech (p=0.04)

learning_rate = 0.001
batch_size = 64
hidden_dim = 700

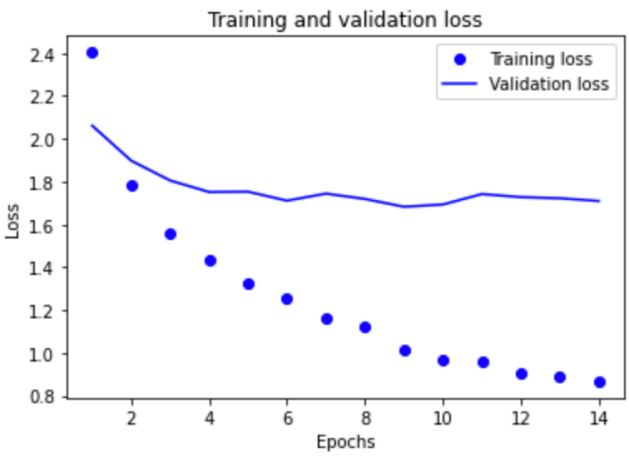
Conclusion: bigger hidden dim would help to decrease loss and increase accuracy. But if hidden dim is too big (like 700), loss would still even smaller, but the increase degree would become lower. It would be nice to choose hidden dim between 500 to 700.

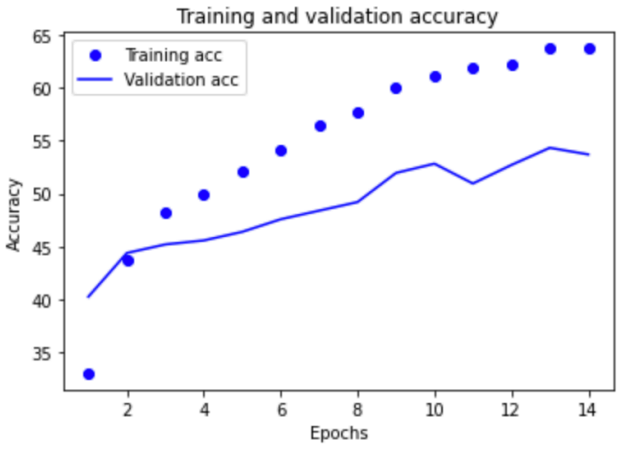
Optimisation

Plan A: relative low loss combined with relative high accuracy
Plan B: relative low loss and not very low accuracy
Plan C: change batch size to get the best final result

Plan A.1:

learning_rate = 0.001
batch_size = 64
p NOT USED
batch_norm
weight_decay NOT USED
hidden_dim = 300



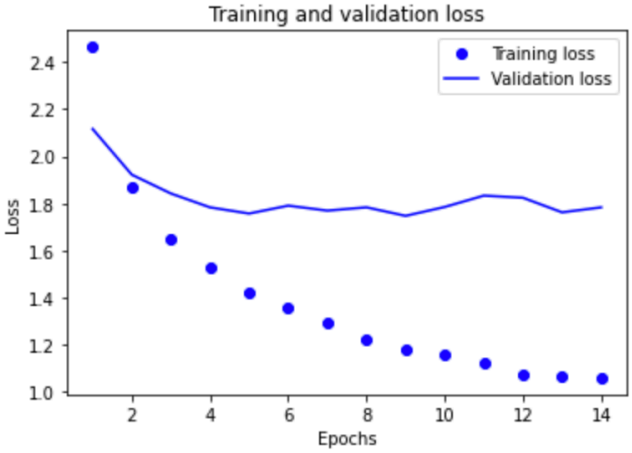


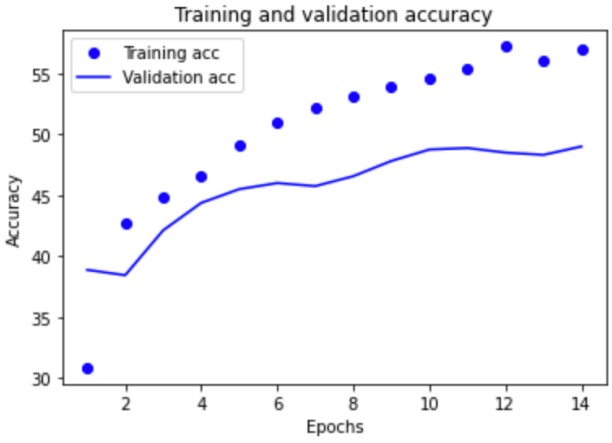
Test loss: 1.7861554527282715;
Test Accuracy: 53.37500000000001
==========
drewer -> English (p=0.56)
drewer -> German (p=0.23)
drewer -> Dutch (p=0.11)
drewer -> Polish (p=0.04)
drewer -> Czech (p=0.03)

It is good to use batch norm.

Plan A.2:

learning_rate = 0.001
batch_size = 64
p = 0.2
batch_norm
weight_decay NOT USED
hidden_dim = 300





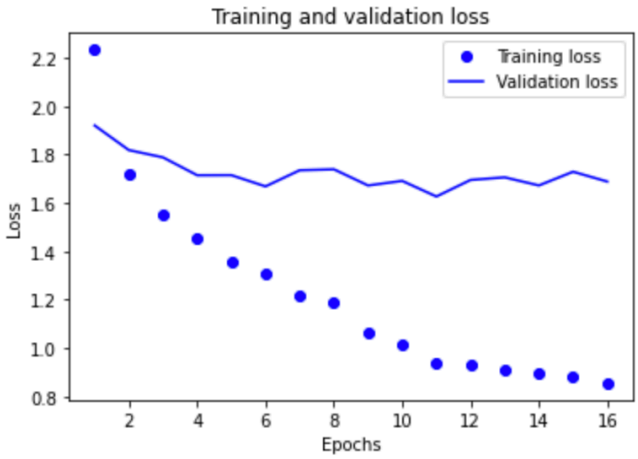
Test loss: 1.7962462091445925;
Test Accuracy: 49.5625
==========
drewer -> English (p=0.42)
drewer -> German (p=0.39)

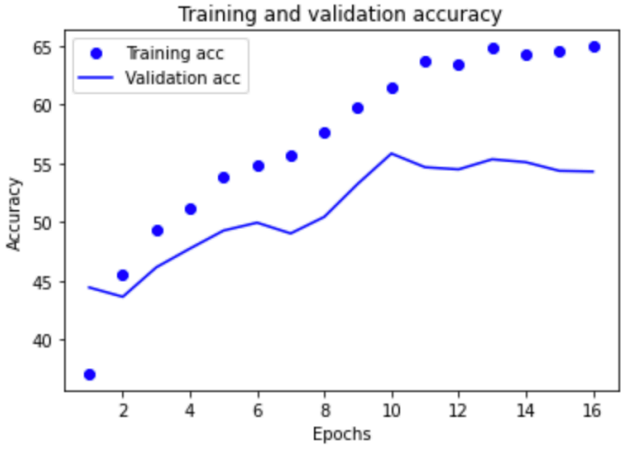
drewer -> Polish (p=0.06)
drewer -> Dutch (p=0.05)
drewer -> Czech (p=0.03)

It is not very good to use dropout because it make result from Plan A.1 become worse.

Plan A.3:

learning_rate = 0.001
batch_size = 16
p NOT USED
batch_norm
weight_decay NOT USED
hidden_dim = 300



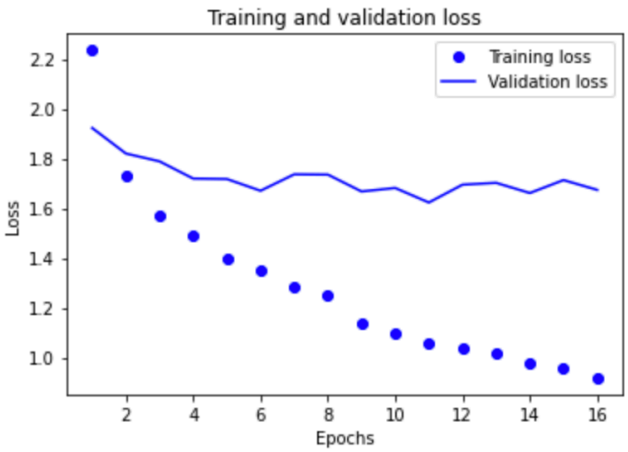


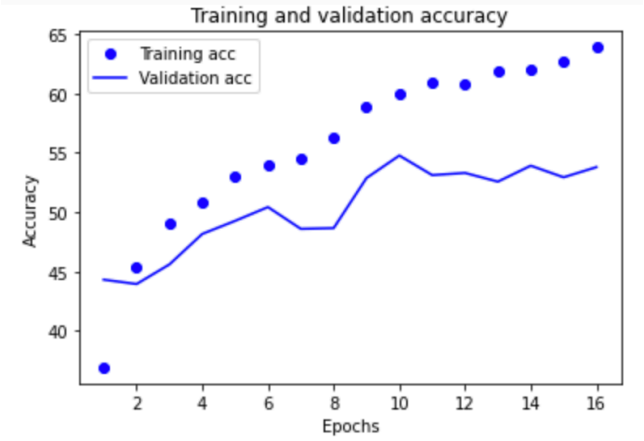
Test loss: 1.750648183730042;
Test Accuracy: 55.15776699029127
==========
drewer -> German (p=0.45)
drewer -> English (p=0.39)
drewer -> Czech (p=0.04)
drewer -> Scottish (p=0.04)
drewer -> Dutch (p=0.03)

It is good to use 16 as batch size to improve the outcome.

Plan A.4:

learning_rate = 0.001
batch_size = 16
p NOT USED
batch_norm
weight_decay = 0.001
hidden_dim = 300



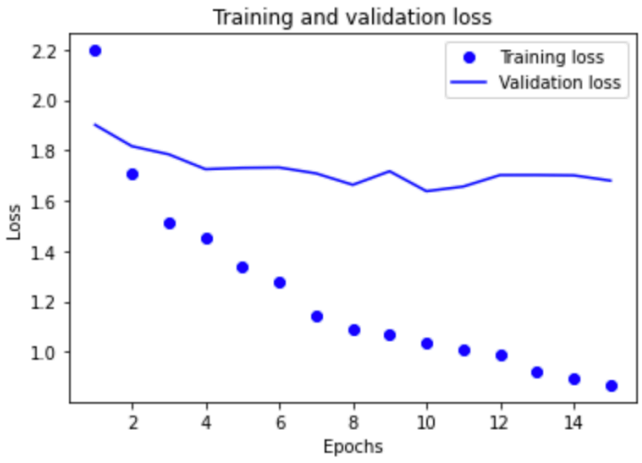


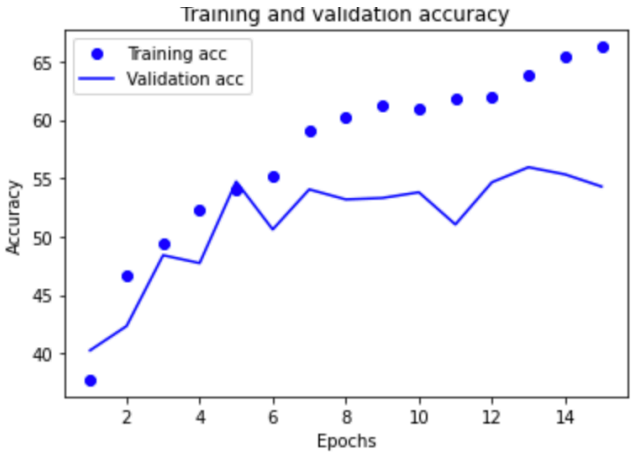
Test loss: 1.7241340024957381;
Test Accuracy: 54.30825242718448
==========
drewer -> German (p=0.43)
drewer -> English (p=0.42)
drewer -> Dutch (p=0.03)
drewer -> Czech (p=0.03)
drewer -> Polish (p=0.03)

It is good to use weight decay here to decrease the loss, while the accuracy also decrease a little.

Plan A.5:

learning_rate = 0.001
batch_size = 16
p NOT USED
batch_norm
weight_decay = 0.001
hidden_dim = 500



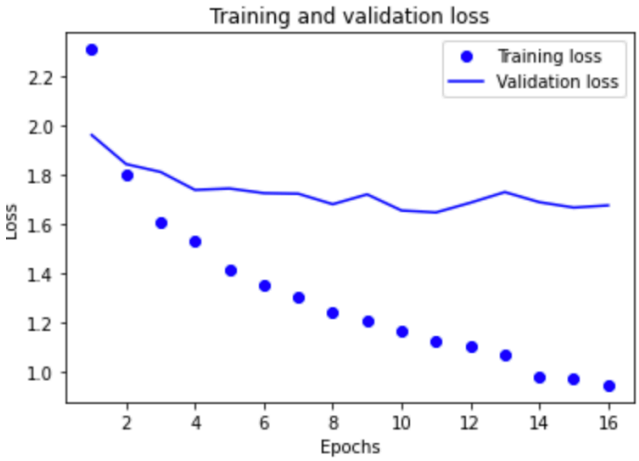


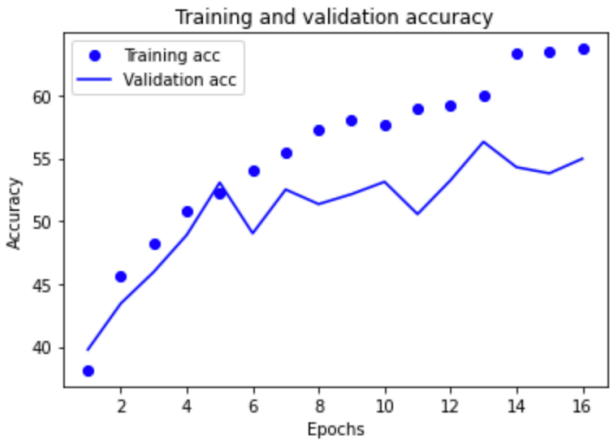
Test loss: 1.7346772644126296;
Test Accuracy: 55.33980582524273
==========
drewer -> German (p=0.43)
drewer -> English (p=0.45)
drewer -> German (p=0.35)
drewer -> Dutch (p=0.12)
drewer -> Scottish (p=0.04)
drewer -> Czech (p=0.02)

It is good to use change hidden dim compared with Plan A.3. But it depends whether to change hidden dim compared with Plan A.4, because A.5 increases both accuracy (by about 1%) and loss (by about 0.5%)

Plan B.1:

learning_rate = 0.0005
batch_size = 16
p NOT USED
batch_norm
weight_decay = 0.001
hidden_dim = 500



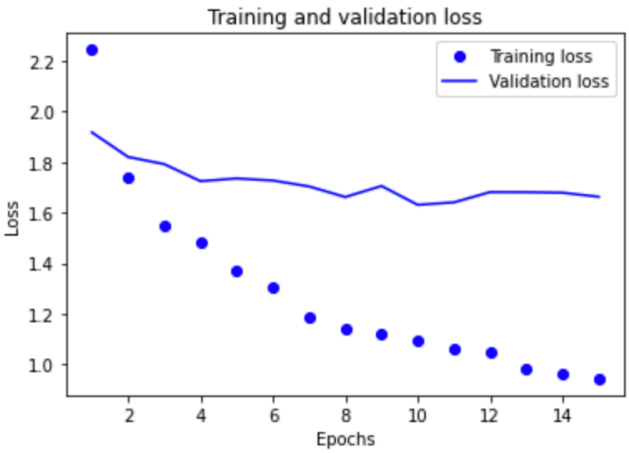


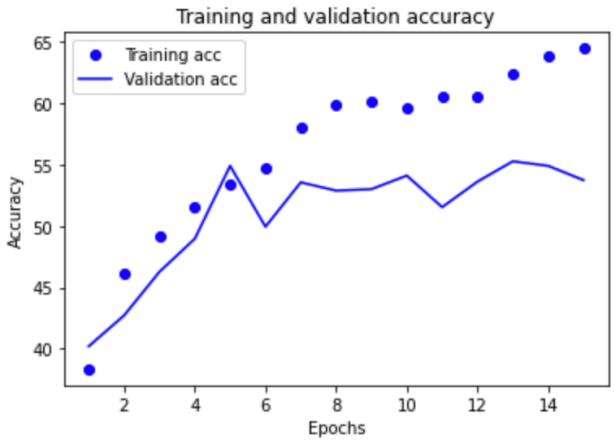
Test loss: 1.644436649327139;

Test Accuracy: 52.5485436893204
==========
drewer -> English (p=0.38)
drewer -> German (p=0.36)
drewer -> Scottish (p=0.09)
drewer -> Dutch (p=0.08)
drewer -> Czech (p=0.04)

Plan B.2:

learning_rate = 0.0007
batch_size = 16
p NOT USED
batch_norm
weight_decay = 0.001
hidden_dim = 500

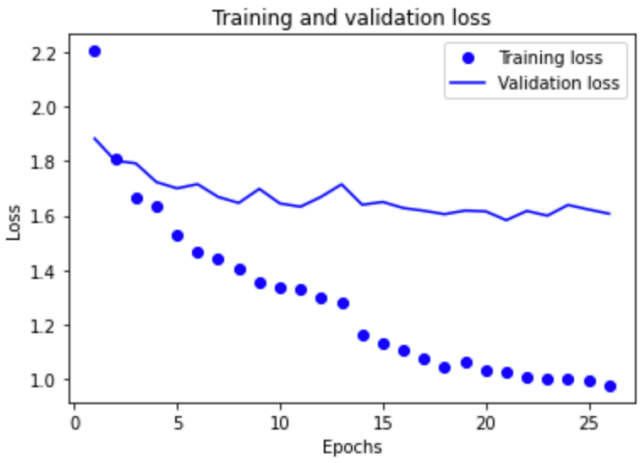


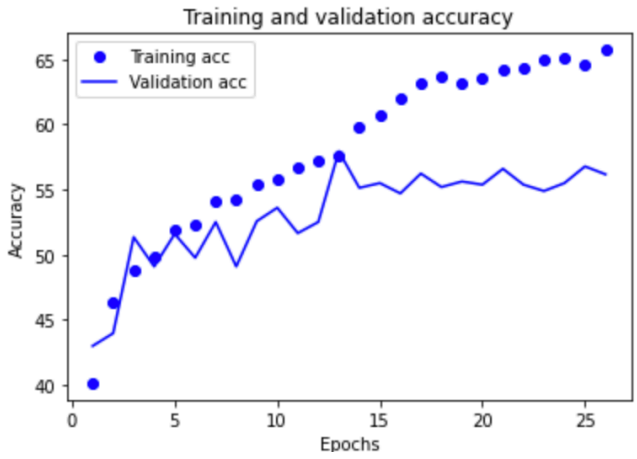


Test loss: 1.7186786092600779;
Test Accuracy: 54.672330097087375
==========
drewer -> German (p=0.39)
drewer -> English (p=0.39)
drewer -> Dutch (p=0.11)
drewer -> Scottish (p=0.05)
drewer -> Czech (p=0.02)

Plan C:

learning_rate = 0.0007
batch_size = 8
p NOT USED
batch_norm
weight_decay = 0.001
hidden_dim = 500





Test loss: 1.6162373506360586;
Test Accuracy: 57.789855072463766
==========
drewer -> German (p=0.46)

drewer -> English (p=0.35)
drewer -> Dutch (p=0.06)
drewer -> Scottish (p=0.04)
drewer -> Czech (p=0.04)

This is the best outcome so far.