RNN Model using Keras

Why RNN over CNN?

I chose to use a recurrent neural network (RNN) over a convolution neural network (CNN) because I understand they're more suited to handle temporal data. Specifically, identifying and modelling trends over time. Since the dataset I'm working with entails climate data dating back to 1960, I assumed an RNN would be a good choice. As you will see below, I used both LSTMs and GRUs during my different iterations in an attempt to improve accuracy and loss convergence.

Starting Hyperparameters, Performance and Confusion Matrix

Epochs: 50

Batch size: 16

# of hidden layers: 32

Dropout: 0.5

Accuracy and Loss: Accuracy starts low, around 0.08, and over the epochs it decreases to around 0.05; while the loss does the opposite- starting around 10 and increasing to 33 by the end. This is the opposite of what we want to see. In other words, the model is not converging.

```
180/180 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step
Pred        BASEL  BELGRADE  BUDAPEST  DUSSELDORF  LJUBLJANA  MADRID  \
True
BASEL           0        17         4           0          0    1448
BELGRADE        0        14         4           2          0     991
BUDAPEST        0         4         0           0          1     208
DEBILT          0         0         0           0          0      82
DUSSELDORF      0         1         0           0          0      28
HEATHROW        0         2         1           0          0      75
KASSEL          0         0         0           0          0      11
LJUBLJANA       0         1         1           0          0      59
MAASTRICHT      0         0         0           1          0       5
MADRID          1        12         1           0          0     251
MUNCHENB        0         0         1           0          0       5
OSLO            0         0         0           0          0       5
STOCKHOLM       0         0         0           0          0       4
VALENTIA        0         1         0           0          0       0

Pred        MUNCHENB  OSLO
True
BASEL              6  2207
BELGRADE           2    79
BUDAPEST           1     0
DEBILT             0     0
DUSSELDORF         0     0
HEATHROW           0     4
KASSEL             0     0
LJUBLJANA          0     0
MAASTRICHT         0     3
MADRID             5   188
MUNCHENB           1     1
OSLO               0     0
STOCKHOLM          0     0
VALENTIA           0     0
```

Observation: This model had a strong bias toward classifying samples into certain stations, such as Oslo, Basel, Madrid and Belgrade; this came at the cost of barely identifying any of the other stations correctly, perhaps due to insufficient learning or class imbalances.

Ending Hyperparameters, Performance and Confusion Matrix

Epochs: 50

Batch size: 32

# of hidden layers: 128

Dropout: 0.4

Accuracy and Loss: Accuracy starts higher than the first iteration, around 0.14, however by the end it drops to 0.09; while not as stark a change as the first iteration, the loss still increases over time, starting at 10 and ending at just over 11.

180/180 ━━━━━━━━━━━━━━━━━━━━ **3s** 9ms/step

Pred True	BELGRADE	BUDAPEST	DEBILT	KASSEL	MADRID	SONNBLICK	VALENTIA
BASEL	33	0	1	0	1447	1856	345
BELGRADE	79	0	0	3	962	12	36
BUDAPEST	11	0	0	0	199	0	4
DEBILT	0	0	0	0	82	0	0
DUSSELDORF	0	0	1	0	28	0	0
HEATHROW	1	0	1	0	75	0	5
KASSEL	0	0	0	0	11	0	0
LJUBLJANA	1	1	0	0	58	0	1
MAASTRICHT	0	0	0	0	5	0	4
MADRID	3	0	0	0	335	9	111
MUNCHENB	0	0	0	0	6	0	2
OSLO	0	0	0	0	5	0	0
STOCKHOLM	0	0	0	0	4	0	0
VALENTIA	0	0	0	0	0	0	1

Observation: There is still a strong bias toward certain stations, in this case Madrid, Sonnblick and Basel. Besides that, the model is slightly better than the previous iteration, however, still there are many stations that are barely classified, and not all 15 are even recognized. Once again, class imbalance is playing a role.

Final Keras layout:

Layer (type)	Output Shape	Param #
conv1d_18 (Conv1D)	(None, 14, 128)	2,432
max_pooling1d_18 (MaxPooling1D)	(None, 7, 128)	0
bidirectional_6 (Bidirectional)	(None, 7, 256)	198,144
batch_normalization_13 (BatchNormalization)	(None, 7, 256)	1,024
bidirectional_7 (Bidirectional)	(None, 256)	296,448
dropout_16 (Dropout)	(None, 256)	0
dense_16 (Dense)	(None, 15)	3,855

Total params: 1,003,296 (3.83 MB)

Trainable params: 501,391 (1.91 MB)

Non-trainable params: 512 (2.00 KB)

Optimizer params: 501,393 (1.91 MB)

Bonus: CNN Model Using Keras

Starting Hyperparameters, Performance and Confusion Matrix

Epochs: 50

Batch size: 32

# of hidden layers: 64

Dropout: 0.5

Learning rate: 0.0001

Accuracy and Loss: Accuracy remains low throughout, hovering around 0.10-0.12, while the loss shows an exponential increase, starting at 15 and ending at 936 by the 11th epoch, where the model decided to end it. With the model diverging instead of converging, this could be due to a learning rate that's too high, or the model complexity is too high.

```
180/180 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
Pred        BASEL  BELGRADE  BUDAPEST  DUSSELDORF  LJUBLJANA  MADRID  \
True
BASEL           7       159         8           2         15    3489
BELGRADE        4       191         2           3          2     889
BUDAPEST        0        26         0           1          3     183
DEBILT          0         6         1           0          0      74
DUSSELDORF      0         4         0           1          0      24
HEATHROW        1         7         1           0          0      73
KASSEL          0         4         0           0          0       7
LJUBLJANA       0         3         0           0          0      57
MAASTRICHT      0         0         0           0          0       8
MADRID          0         4         0           0          1     453
MUNCHENB        0         0         0           0          0       8
OSLO            0         0         2           0          0       3
STOCKHOLM       0         0         0           0          0       4
VALENTIA        0         0         0           0          0       1

Pred        STOCKHOLM
True
BASEL               2
BELGRADE            1
BUDAPEST            1
DEBILT              1
DUSSELDORF          0
HEATHROW            0
KASSEL              0
LJUBLJANA           1
MAASTRICHT          1
MADRID              0
MUNCHENB            0
OSLO                0
STOCKHOLM           0
VALENTIA            0
```

Ending Hyperparameters, Performance and Confusion Matrix

Epochs: 50

Batch size: 32

# of filters: 32

Dropout: 0.5

Learning rate: 0.00001

Accuracy and Loss: Once again, accuracy and validation accuracy remain low throughout, starting at 0.10 and 0.16 respectively, then decreasing to 0.09 and 0.08. This indicates that this architecture may not be ideal for this data. The loss and validation loss gradually increase over the epochs, from 8 to 24 and 8 to 21 respectively, although this is an improvement from the first iteration of CNN only.

180/180 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step

Pred True	BASEL	BELGRADE	BUDAPEST	DEBILT	DUSSELDORF	HEATHROW	KASSEL	\
BASEL	611	1	43	2	11	235	90	
BELGRADE	272	3	0	0	4	161	12	
BUDAPEST	39	0	0	0	0	25	1	
DEBILT	18	0	0	0	0	14	0	
DUSSELDORF	3	0	0	0	0	4	0	
HEATHROW	12	0	0	0	0	5	0	
KASSEL	3	0	0	0	0	1	0	
LJUBLJANA	14	0	0	0	0	2	0	
MAASTRICHT	0	0	0	0	0	0	1	
MADRID	26	2	0	0	2	9	4	
MUNCHENB	1	0	0	0	0	0	0	
OSLO	0	0	0	0	0	0	0	
STOCKHOLM	0	0	0	0	0	0	0	
VALENTIA	0	0	0	0	0	0	0	

Pred True	LJUBLJANA	MAASTRICHT	MADRID	MUNCHENB	OSLO	SONNBLICK	\
BASEL	662	27	1597	55	276	50	
BELGRADE	25	1	591	4	16	1	
BUDAPEST	0	0	142	1	5	0	
DEBILT	0	0	49	1	0	0	
DUSSELDORF	0	0	22	0	0	0	
HEATHROW	0	0	62	0	1	0	
KASSEL	0	0	7	0	0	0	
LJUBLJANA	0	0	45	0	0	0	
MAASTRICHT	1	0	6	0	1	0	
MADRID	13	0	366	1	25	1	
MUNCHENB	0	0	6	1	0	0	
OSLO	0	0	4	0	0	0	
STOCKHOLM	0	0	4	0	0	0	
VALENTIA	0	0	1	0	0	0	

Pred True	STOCKHOLM	VALENTIA
BASEL	8	14
BELGRADE	2	0
BUDAPEST	1	0
DEBILT	0	0
DUSSELDORF	0	0
HEATHROW	1	1
KASSEL	0	0
LJUBLJANA	0	0
MAASTRICHT	0	0
MADRID	9	0
MUNCHENB	0	0
OSLO	1	0
STOCKHOLM	0	0
VALENTIA	0	0

Confusion matrix observation: This is the first time the model has recognized and included all 15 stations in the matrix. However, as you can see, there are still significant class imbalances, and many stations are vastly underrecognized when attempting to classify.

Final reflections: While this has been useful practice to understand the differences between CNNs and RNNs, as well as manipulating the hyperparameters and other parameters to change the outcome, we still are not at a place where the model is of any use. However, as the assignment indicates just to attempt to improve it, the model doesn't necessarily have to be up to task quite yet. Hopefully with later assignments where we optimize parameters we will arrive at a place where it can be deployed and garner useful and actionable insights.

Post-Feedback Revisions: CNN Model

Iteration #9: Reduced learning rate to 0.0005; applying class weights and random oversampling to address class imbalances

Epochs: 50

Batch size: 32

# of filters: 64->128->256

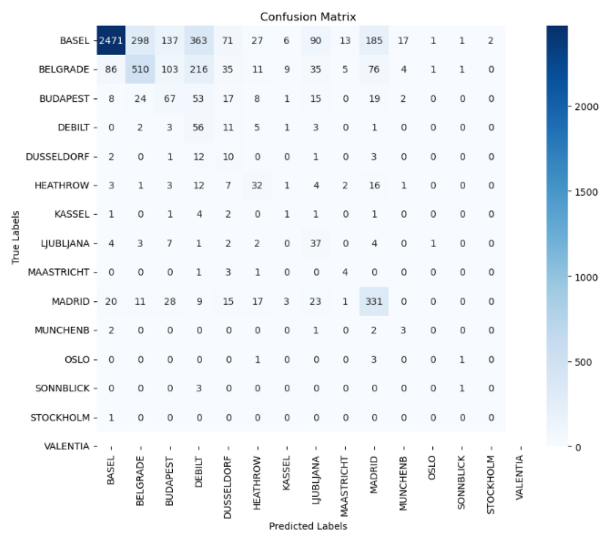
Learning rate: 0.0005

Accuracy and Loss: This showed the best results of any CNN or RNN model thus far: accuracy and validation accuracy were high (95% and 72% respectively); and loss and validation loss were low and stayed low (0.15 and 1.5 respectively). This tells us that the model performs much better on unseen data and has vastly reduced error rates, improving performance on all fronts.

```
Unique classes in y_true: [ 0  1  2  3  4  5  6  7  8  9 10 11 13 14]
Classification Report:
              precision    recall  f1-score   support

       BASEL       0.95      0.67      0.79      3682
    BELGRADE       0.60      0.47      0.53      1092
    BUDAPEST       0.19      0.31      0.24       214
      DEBILT       0.08      0.68      0.14        82
  DUSSELDORF       0.06      0.34      0.10        29
    HEATHROW       0.31      0.39      0.34        82
      KASSEL       0.05      0.09      0.06        11
   LJUBLJANA       0.18      0.61      0.27        61
  MAASTRICHT       0.16      0.44      0.24         9
      MADRID       0.52      0.72      0.60       458
    MUNCHENB       0.11      0.38      0.17         8
        OSLO       0.00      0.00      0.00         5
   STOCKHOLM       0.25      0.25      0.25         4
    VALENTIA       0.00      0.00      0.00         1

    accuracy                           0.61      5738
   macro avg       0.25      0.38      0.27      5738
weighted avg       0.78      0.61      0.67      5738
```



Classification report and confusion matrix: There still remains class imbalances, although this version is vastly improved compared to previous iterations. Stations with more representation (such as Basel and Belgrade) had good performance, while smaller class stations (like Dusseldorf and Valentia) didn't do as well. Precision and recall for the larger classes were better- above 0.6 for a number of stations, and yet those same metrics were close to or at zero for the worst-performing stations. There's still some work to for the final product but this model is trending in the right direction.

Post-Feedback Revisions: RNN Model

Iteration #10: reduced learning rate, added another LSTM layer and used random oversampling

Epochs: 50

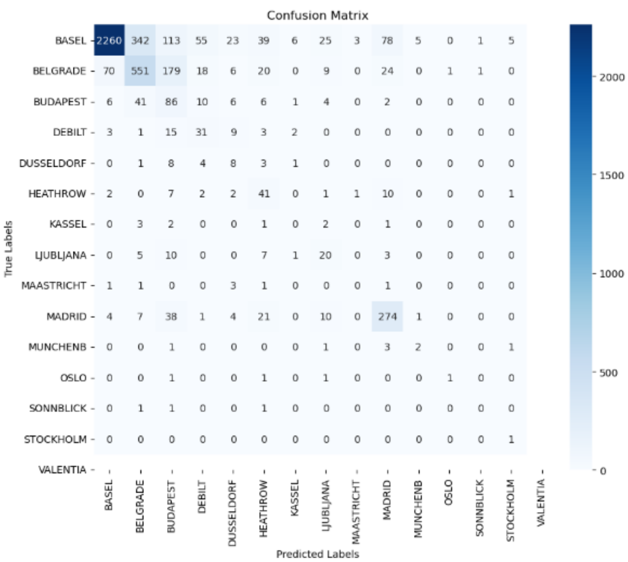
Batch size: 32

Learning rate: 0.0001

Accuracy and Loss: This is the best-performing model yet, with a final validation accuracy of 72.6% and a low validation loss of 0.93. This tells us the model can perform reasonably well on unseen data while minimizing error.

Unique classes in y_true: [0 1 2 3 4 5 6 7 8 9 10 11 13 14]
Classification Report:

	precision	recall	f1-score	support
BASEL	0.96	0.76	0.85	2955
BELGRADE	0.58	0.63	0.60	879
BUDAPEST	0.19	0.53	0.28	162
DEBILT	0.26	0.48	0.34	64
DUSSELDORF	0.13	0.32	0.19	25
HEATHROW	0.28	0.61	0.39	67
KASSEL	0.00	0.00	0.00	9
LJUBLJANA	0.27	0.43	0.34	46
MAASTRICHT	0.00	0.00	0.00	7
MADRID	0.69	0.76	0.72	360
MUNCHENB	0.25	0.25	0.25	8
OSLO	0.50	0.25	0.33	4
STOCKHOLM	0.00	0.00	0.00	3
VALENTIA	0.12	1.00	0.22	1
accuracy			0.71	4590
macro avg	0.30	0.43	0.32	4590
weighted avg	0.80	0.71	0.75	4590



Classification report and confusion matrix: As has been the case throughout this project, class imbalance still plays a major role in why this model doesn't perform even better We can see that well-represented stations (Basel, Belgrade, etc) continue to perform well and are classified correctly, while those that are under-represented (Dusseldorf, Kassel, etc) do quite poorly at times, with a lot of them being misclassified into the dominant classes. This is reflected in the classification report, with moderately-high recall and precision scores of 0.6 and above for those larger classes, and then very low to zero for those that are in the minority.

Iteration #12: random oversampling with increased model complexity through a third LSTM layer

Epochs: 50

Batch size: 32

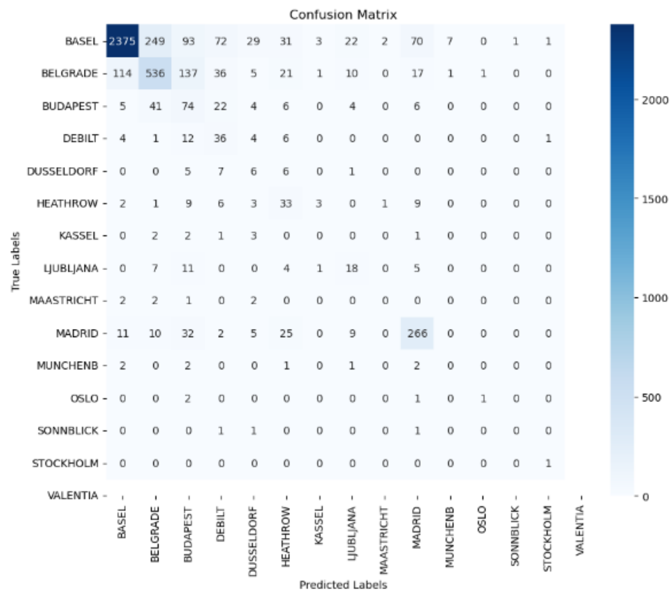
Learning rate: 0.0001

Accuracy and Loss: This model improves upon the 10th iteration, showing a final validation accuracy of 72.9% while keeping the loss low as well. It appears that the addition of a third LSTM has marginally improved overall performance.

```
Unique classes in y_true: [ 0  1  2  3  4  5  6  7  8  9 10 11 13 14]
Classification Report:
              precision    recall  f1-score   support

       BASEL       0.94      0.80      0.87      2955
    BELGRADE       0.63      0.61      0.62       879
    BUDAPEST       0.19      0.46      0.27       162
      DEBILT       0.20      0.56      0.29        64
  DUSSELDORF       0.10      0.24      0.14        25
    HEATHROW       0.25      0.49      0.33        67
      KASSEL       0.00      0.00      0.00         9
   LJUBLJANA       0.28      0.39      0.32        46
  MAASTRICHT       0.00      0.00      0.00         7
      MADRID       0.70      0.74      0.72       360
    MUNCHENB       0.00      0.00      0.00         8
        OSLO       0.50      0.25      0.33         4
   STOCKHOLM       0.00      0.00      0.00         3
    VALENTIA       0.33      1.00      0.50         1

    accuracy                           0.73      4590
   macro avg       0.29      0.40      0.31      4590
weighted avg       0.80      0.73      0.76      4590
```



Classification report and confusion matrix: This shows similar results to the iteration shown above, with high recall and precision for those classes that are well-represented, and much lower scores for those classes that are not. This is reflected in the confusion matrix as well, albeit with slightly fewer misclassifications.