**Final Project Report**

Xueting Li, Meng Xu, Poonam Ahuja

**Develop your deep learning model in Tensorflow.**

**You need to submit two files:**

* **Submit a coding ipynb file. Make sure that the coding notebook has been fully executed from beginning to end, and all of the cell outputs are visible.**
* **Submit a report pdf. Use this word document and write your answers below each question and insert your combination tables.**

**In your coding pdf you should:**

1. **Describe and preprocess your data. Split data to a fixed training, validation and test data.**
2. **Construct, compile and fit a Sequential model to your training and validation data. If you use image data, construct a convolutional neural network model.**
3. **Monitor overfitting on validation data by (use accuracy or loss plots).**
4. **Evaluate the model on test data.**
5. **Use callbacks to save the weights of your final (best) Tensorflow model.**

**In your report pdf you should:**

1. **Fit training set well:** 
   1. **What metric(s) are you using to evaluate your training performance (training accuracy, recall, precision, false positives, etc.)? Explain why these metrics are useful in your data context.**

In the project, we used training accuracy to evaluate the performance of models.

The training accuracy is the accuracy of a model on examples it was constructed on. We can compare the training and validation accuracies to identify overfitting and underfitting issue. A training accuracy that is subjectively far higher than validation accuracy indicates over-fitting, while the opposite is underfitting.

* 1. **Comparing to human level performance (or any other benchmark in your data context), how well is your model fitting the training data?**

Here we have considered the benchmark as our training model

Our initial model is a CNN model with one conventional layer, one max pooling layer.

Before we trained our model, we got the test accuracy which is 0.034.

**图形用户界面, 文本, 应用程序

描述已自动生成**

After training our model with 15 epochs and 64 batch size, the test accuracy became 0.965.

图形用户界面, 应用程序

描述已自动生成

* 1. **Can you improve your training performance using a bigger neural network? Report a table in which you show the training performance for a total of ten different models (with different number of hidden layers and/or different number of hidden units).**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Model** | **accuracy** | **fit issue** |
| 1 | add 1 dense layer with 32 units | 0.967 | slight overfitting |
| 2 | add 1 dense layer with 128 units | 0.976 | slight overfitting |
| 3 | add 1 conventional layer with 16 filters | 0.976 | overfitting |
| 4 | add 1 conventional layer with 20 filters | 0.975 | slight overfitting |
| 5 | add 1 conventional layer with 16 filters and 1 max pooling layer | 0.967 | slight overfitting |
| 6 | add 1 conventional layer with 16 filters, 1 max pooling layer and 1 dense layer with 32 units | 0.973 | None(Good fit) |
| 7 | add 1 dense layer with 32 units and another dense layer with 64 unit | 0.979 | None(Good fit) |
| 8 | add 1 dense layer with 32 units and another dense layer with 16 unit | 0.961 | None(Good fit) |
| 9 | add 1 conventional layer with 16 filters, 1 max pooling layer, 1 dense layer with 32 units and another dense layer with 16 unit | 0.976 | None(Good fit) |
| 10 | add 1 conventional layer with 16 filters, 1 max pooling layer, 1 conventional layer with 6 filters, 1 max pooling layer and 1 dense layer with 16 units | 0.968 | None(Good fit) |

* 1. **Can you improve your training performance using a better** **optimization algorithm? Report a table in which you show the training performance for at least three different optimization algorithms.**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **optimization algorithm** | **accuracy** | **fit issue** |
| 1 | SGD, with learning rate of 0.001 | 0.969 | None(Good fit) |
| 2 | Nadam with learning rate of 0.001 | 0.981 | None(Good fit) |
| 3 | Adagrad, with learning rate of 0.001 | 0.956 | None(Good fit) |

* 1. **Can you improve your training performance using a higher number of iterations (****epochs)? Report a table in which you show the training performance for at least three different number of epochs.**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **number of epochs** | **accuracy** | **fit issue** |
| 1 | 20 | 0.982 | None(Good fit) |
| 2 | 25 | 0.980 | overfitting |
| 3 | 30 | 0.984 | overfitting |

* 1. **Can you improve your training performance using a better weight initialization? Report a table in which you show the training performance for at least three different weight initializations.**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **weight initialization** | **accuracy** | **fit issue** |
| 1 | HeUniform | 0.979 | None(Good fit) |
| 2 | LecunUniform | 0.977 | slight overfitting |
| 3 | RandomUniform | 0.978 | overfitting |

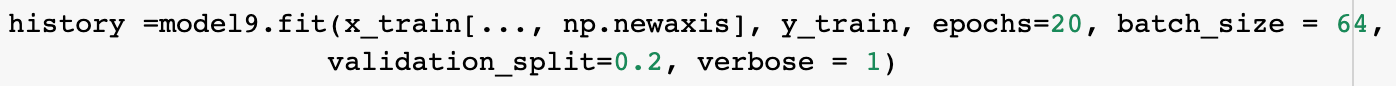
* 1. **Explain which model is the best model fitting your training set? Use this model for the next part.**

From the results above, we select the following as our best model. Considering the training and validation accuracy, Model 9’s test accuracy is top ranked and without overfitting and underfitting. Moreover, when the optimizer is Nadam and the number of epochs is 20, the model works better. Therefore, we built the model below. 图片包含 文本

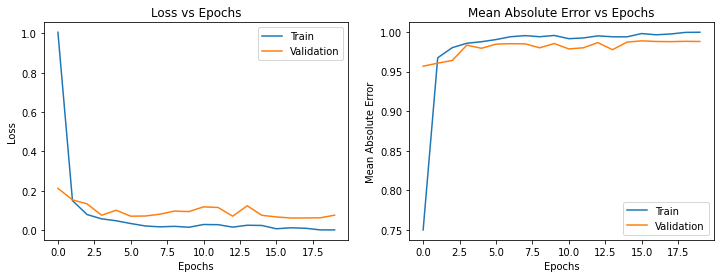
描述已自动生成

**图形用户界面, 文本

描述已自动生成**

****

1. **Fit validation set well:**
2. **Compared to training performance, how well is your model fitting the validation data? Explain if you have an overfitting problem.**



From the model for this step, the overfitting problem is not a big concern. The training accuracy is just a little higher than that of validation set.

1. **Can you improve your validation performance using L2 regularization? Report a table in which you show the validation performance for five different penalty () rates.**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **penalty (λ) rates** | **accuracy** | **fit issue** |
| 1 | 0.0001 | 0.979 | None(Good fit) |
| 2 | 0.001 | 0.980 | None(Good fit) |
| 3 | 0.01 | 0.982 | slight overfitting |
| 4 | 0.1 | 0.980 | None(Good fit) |
| 5 | 0.2 | 0.979 | None(Good fit) |

1. **Can you improve your validation performance using dropout regularization? Report a table in which you show the validation performance for five different dropout rates.**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **dropout rates** | **accuracy** | **fit issue** |
| 1 | 0.1 | 0.979 | None(Good fit) |
| 2 | 0.2 | 0.979 | None(Good fit) |
| 3 | 0.3 | 0.979 | underfitting |
| 4 | 0.4 | 0.684 | underfitting |
| 5 | 0.5 | 0.100 | underfitting |

1. **Can you improve your validation performance using a mixture of dropout regularization and L2 regularization? Report a table in which you show the validation performance for five different combinations.**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **mixture of dropout regularization and L2 regularization** | **accuracy** | **fit issue** |
| 1 | Dropout = 0.1, penalty rate = 0.01 | 0.984 | None(Good fit) |
| 2 | Dropout = 0.2, penalty rate = 0.01 | 0.984 | None(Good fit) |
| 3 | Dropout = 0.1, penalty rate = 0.001 | 0.979 | None(Good fit) |
| 4 | Dropout = 0.2, penalty rate = 0.001 | 0.983 | slight underfitting |
| 5 | Dropout = 0.2, penalty rate = 0.1 | 0.979 | slight underfitting |

1. **Can you improve your validation performance using batch-normalization? Report a table in which you show the validation performance for five different batch-normalizations (one model with default TF batch-normalization parameters and four models with customized parameters).**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **batch-normalization** | **accuracy** | **fit issue** |
| 1 | default | 0.979 | overfitting |
| 2 | epsilon=0.01 | 0.983 | overfitting |
| 3 | momentum=0.5 | 0.985 | overfitting |
| 4 | center=False | 0.984 | slight overfitting |
| 5 | scale=False | 0.980 | overfitting |

1. **Can you improve your validation performance using a mixture of batch-normalization and dropout regularization? Report a table in which you show the validation performance for five different combinations.**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **combinations of batch-normalization and dropout regularization** | **accuracy** | **fit issue** |
| 1 | dropout=0.1, default batch-normalization | 0.983 | None(Good fit) |
| 2 | dropout=0.2, default batch-normalization | 0.987 | slight underfitting |
| 3 | dropout=0.2, batch-normalization epsilon=0.01 | 0.984 | slight underfitting |
| 4 | dropout=0.2, batch-normalization momentum=0.5 | 0.984 | None(Good fit) |
| 5 | dropout=0.3, default batch-normalization | 0.983 | None(Good fit) |

1. **Can you improve your validation performance using a mixture of batch-normalization and dropout regularization and L2 regularization? Report a table in which you show the validation performance for five different combinations.**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **combinations of batch-normalization and dropout regularization and L2 regularization** | **accuracy** | **fit issue** |
| 1 | dropout=0.1, default batch-normalization, penalty rate = 0.001 | 0.983 | None(Good fit) |
| 2 | dropout=0.2, default batch-normalization, penalty rate = 0.001 | 0.979 | None(Good fit) |
| 3 | dropout=0.1, default batch-normalization, penalty rate=0.01 | 0.983 | slight underfitting |
| 4 | dropout=0.2, batch-normalization momentum=0.5, penalty rate=0.001 | 0.988 | None(Good fit) |
| 5 | dropout=0.2, batch-normalization center=False, penalty rate=0.001 | 0.977 | slight underfitting |

1. **Explain which model is the best model fitting your validations set? Use this model for the next part.**

Combining the training and validation accuracy, when we included the mixture of 0.3 Dropout rate and default Batch Normalization, the models performed better. Therefore, we used it in our final model.

1. **Best model:**

**Explain which model would be your final (best) model and why. Compare the training performance, validation performance and test performance.**

We combined all the features above to build our final model. In this model, we are aimed to reach a relatively high test accuracy and reduce the overfitting and underfitting problems. The model below is our final one:

图形用户界面, 文本, 应用程序

描述已自动生成

图形用户界面, 图表, 折线图

描述已自动生成