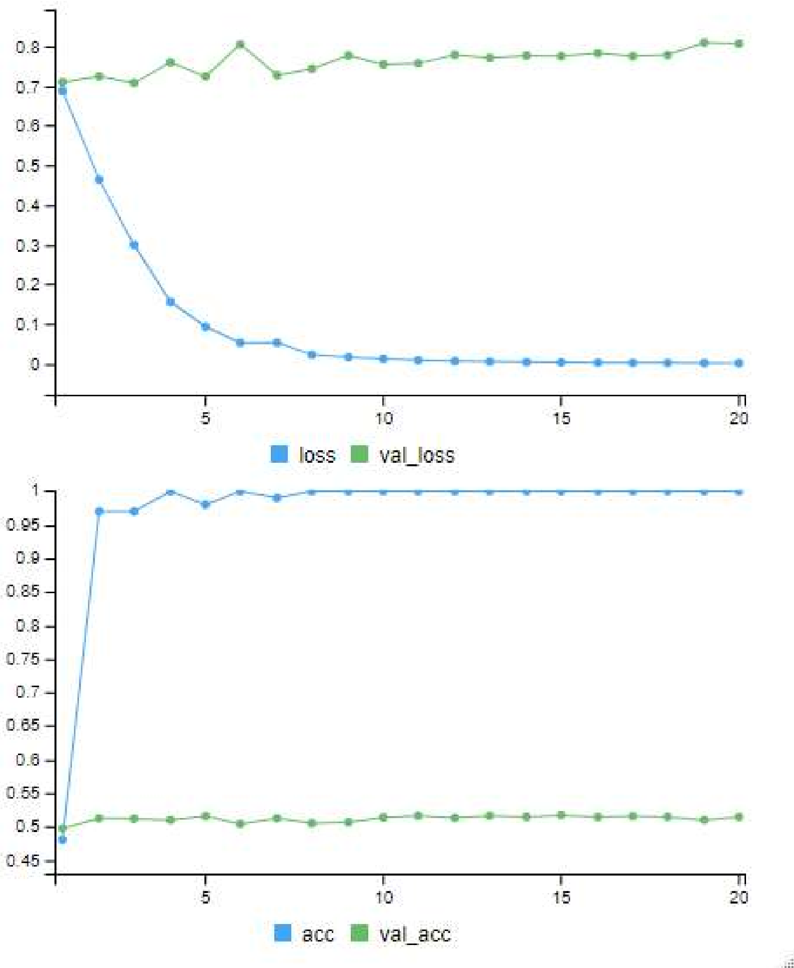
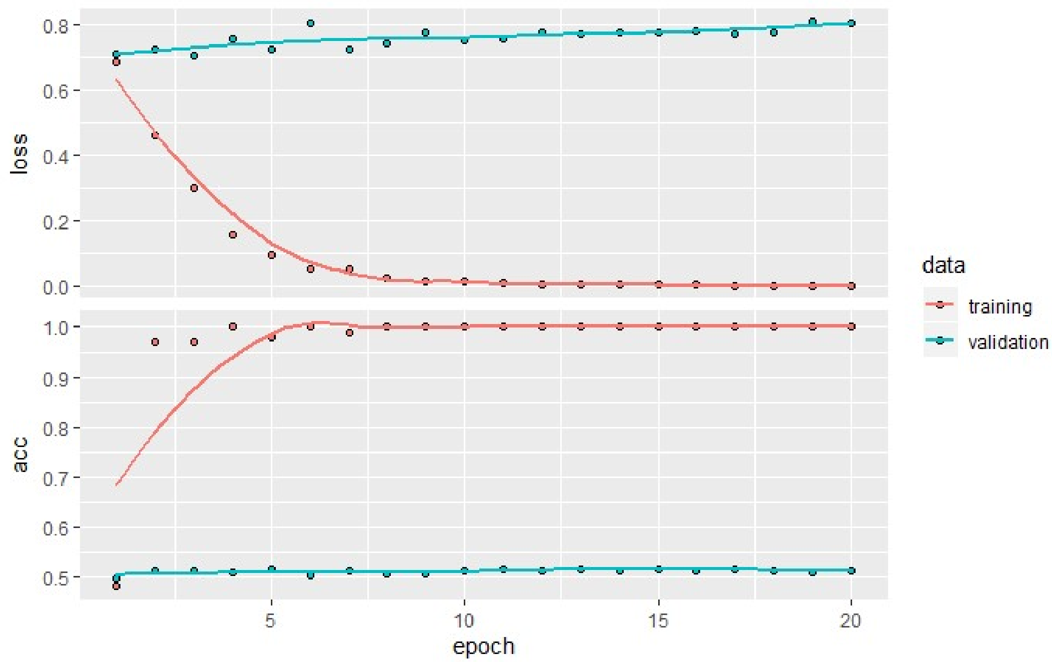
Run 1 = dict {
‘word limit’: **150**,
‘train samples’: **100**,
‘validation samples’: 10000,
‘top words’: 10000 }



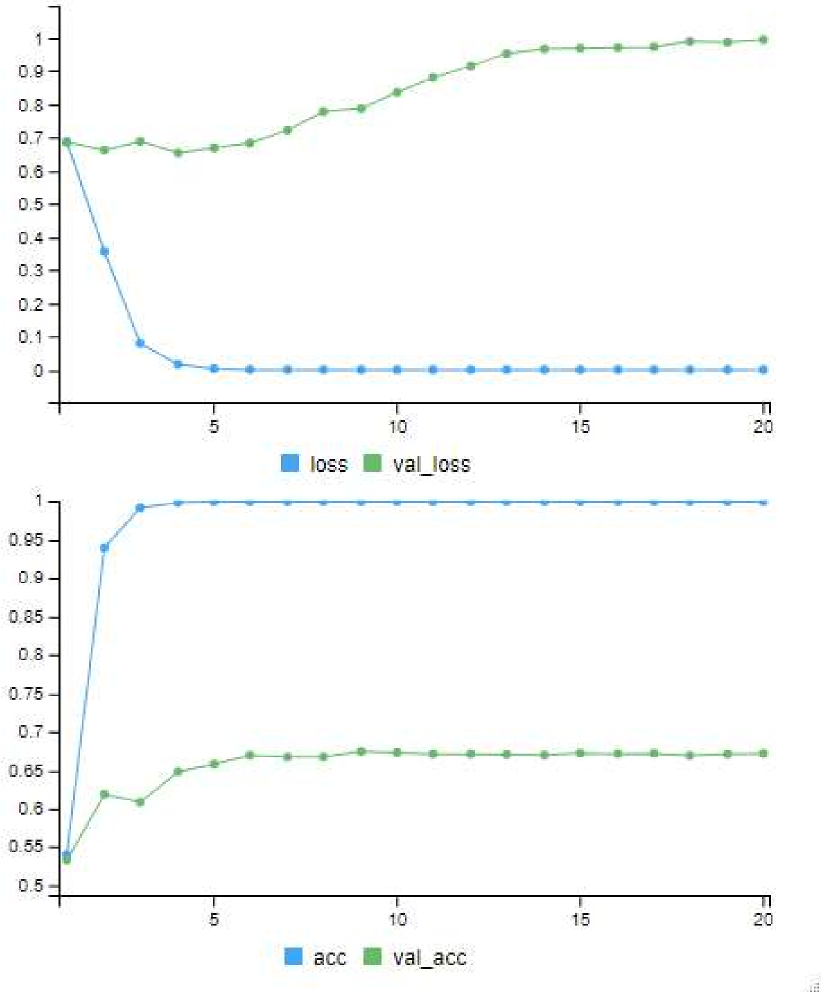
150 per review limit allows for more sentiment detail to be captured but **limiting** training data to just **100 samples** proves the **demise** of this model.



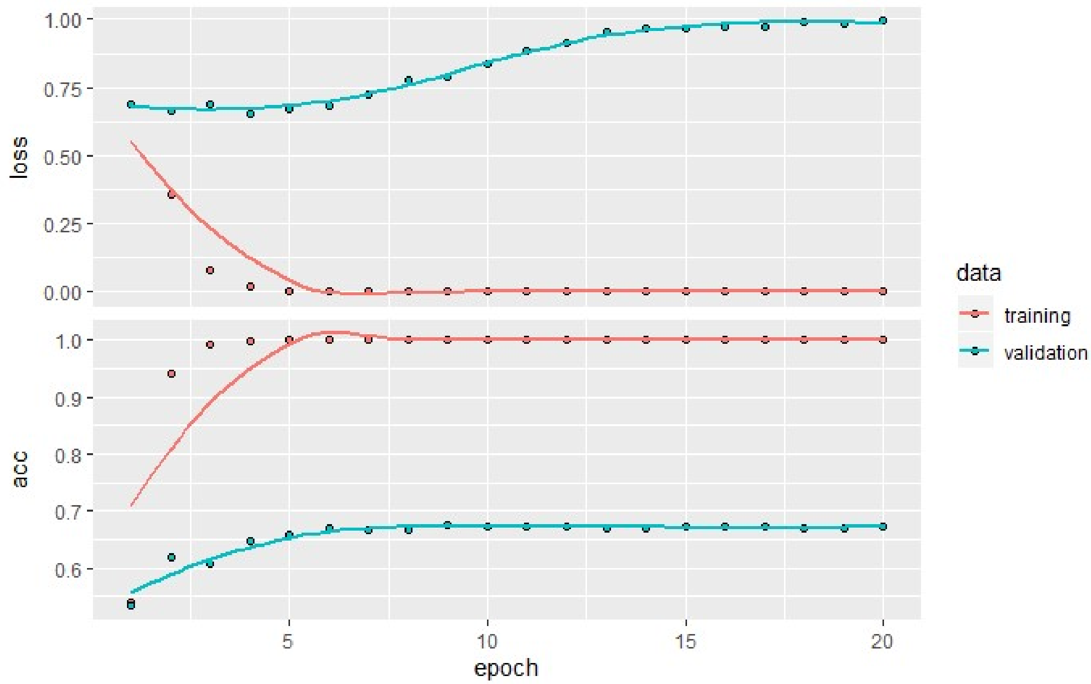
Task-specific embedding layer **does not** outperform pre-trained embedding layer at 100 samples, sadly producing a **53% accuracy** on the final test data.

Run 2 = dict {

‘word limit’: 150,
‘train samples’: **1000**,
‘validation samples’: 10000,
‘top words’: 10000 }



While the edited model still overfit relatively quickly, the **validation accuracy** was able to flatline around **0.65** versus the first run at 0.5.



At **1000 samples** (10% of the validation samples), the task-specific “learn-as-you-go” embedding model **exceeds** the accuracy of the pre-trained model. The graph illustrates the accuracy of the former.

Final Notes

While the custom-trained embedding layer improved over the pre-trained one, the improvement isn't much. Embedding that learns the specific model will always do better than one pre-trained on different data, but a **much larger training sample size** is needed to reap these benefits.