EMOTION CLASSIFICATION

MSC Artificial Intelligence For Media
2024-2025

Alexander Moed

0.1 Introduction

The purpose of this assignment is to train a model that classifies the emotion of a sentence. The categories are anger, fear, joy, sadness, and surprise. The dataset consists of 150,000 lines.

0.2 Research

While researching emotion classification techniques, I discovered a relevant paper comparing LSTM and GRU neural network architectures for analysing hotel reviews. This research, 'Sentiment analysis of hotel comments based on LSTM and GRU' (Xu (2024)), would translate nicely to the assignment.

0.2.1 RNN

RNN, Recurrent Neural Network, is a neural network that is de- signed to keep sequential information and to have a memory that allows it to draw on past details that it has learned and has made connections. However, the main issues are exploding gradients, where the gradients become too large. During backwards propagation, this causes unstable training. Also, vanishing gradients occur, where the gradients become too small. This makes it difficult for the network to retain information from many steps back in the sequence.(Xu (2024))

0.2.2 LSTM

Long Short-Term Memory (LSTM) is a type of RNN network. As mentioned above, RNN had an exploding or diminishing gradient problem. LSTM was designed to deal with that problem. It works by using gates, which act as filters to keep track of new information and determine what to forget because it has been deemed irrelevant. It also has a gate to determine what the output should be. Managing these gates helps prevent issues with gradients. The problem with this method is that it has many parameters and a long training time, which is not ideal. (Xu (2024))

0.2.3 GRU

A Gated Recurrent Unit (GRU) was designed to address the inefficiencies of LSTM, including, as mentioned above, having an excess of parameters and longer training time.(Xu (2024)) GRU also works with gates to control data. It uses just two gates: the update gate, which controls the impact of previously learned information in the hidden layer, and the reset gate, which determines how much past information to forget when computing the new content. The reset gate helps the model decide

which parts of the previous hidden state are relevant to the current state. According to (Xu (2024)), this is more efficient and has not negatively impacted the quality of the model.

0.3 Model choice

From the research above, GRU represents the best of both worlds: it incorporates the improvements of LSTM with the implementation of gates to control data flow. It is simpler in terms of the number of parameters and much faster at training (Xu (2024)). There might be other considerations for specific applications, but based on the literature review, GRU seems like the correct starting place for our emotion classification analysis task.

0.4 Loss function and optimiser

I selected the Adam optimiser based on previous success in other projects and wanted to evaluate its performance in this context. The noted benefits of Adam include its efficiency and adaptability. It makes quick adjustments by maintaining running averages of both gradients (first moment) and their squared values (second moment) (Cerón Viveros (2021)). This second moment helps Adam understand volatility in a gradient. When gradients vary significantly, the optimiser reduces the learning rate to prevent overshooting the optimal value; when gradients are consistent, it increases the learning rate for faster convergence. (Cerón Viveros (2021)) Starting Adam at 0.001 typically works well in practice. It also includes bias correction terms that compensate for initialisation bias in the early stages of training, which explains the rapid improvements often observed in the first few epochs. Adam's downsides include, occasionally, less than optimal solutions than well-tuned Stochastic Gradient Descent (SGD). But I felt it was worth trying, considering my previous success with Adam(Cerón Viveros (2021)).

The model uses categorical cross-entropy for its loss function, adapted from binary cross-entropy for multiclass classification, comparing predicted results with actual outcomes while considering prediction confidence. It is important to penalise confident and wrong predictions because they are dangerous. Sparse Categorical Crossentropy (SCC) further adapts this approach, allowing classes to be represented as single numeric values rather than hot encoded vectors (Versloot (2019)). The loss function aims to accurately represent differences between predictions and outcomes, with confidence tracking providing insights into the model's decision-making process. It also works with multi-class classification. (Versloot (2019))

0.5 Preparing the data

I imported the CSV file and used Pandas because it's known to be efficient and designed for large data frames. We have 150,000 lines of text, a relatively large data frame. What I knew was that I needed to have a list of the options for possible emotions. I turned the emotion column into a set because sets do not allow duplicates. I then assigned a number to the emotion based on the position of the emotion in the set in a dictionary, putting the number as the key.

The next step was to assign a number. If one of the emotions matched the emotion from the set, it was transferred to a new column next to the emotion column in the data frame. It identified a match and then assigned the key for that match to a new column in the data frame. I also removed any duplicate sentences. This became apparent when I compared what was in my testing and training sets, and there were lines that appeared in both.

I then split both the testing and training. I realised we just wanted the numeric label, so I selected only the newly created column of the numeric value and the text itself and did a test and train split with a shuffle. See Figure 1 for the results.

	Text	Emotion	Emotion_Label
0	i could get depressed about feeling isolated b...	sadness	1
1	i am so thankful that though things are a bit ...	fear	0
2	i remember one day years ago when the kids wer...	anger	4
3	i feel so funny he have no topic to chat with ...	surprise	3
4	i didnt take it personally but i could feel so...	fear	0
5	i often feel rushed and comfort sitting on a l...	anger	4
6	i feel as though my mind is restless searching...	fear	0
7	i feel as if they rushed these out to the publ...	anger	4
8	i can pound out something i feel works really ...	joy	2
9	i find god s presence if i feel scared lonely ...	fear	0

Figure 1: Sample Dataframe

0.6 Model version 1

The model is a simple GRU consisting of two layers and is not pretrained. The first GRU layer has 128 units, with 30% dropout applied to reduce overfitting. The second GRU layer follows with 64 units and another 30% dropout. Each unit contains internal gates that control the flow of information, as described in Xu (2024). This model initially required over 15 epochs to exceed 70% accuracy.

In retrospect, the initial parameters were suboptimal. The batch size of 500 meant the model processed only 500 sentences at a time, and the maximum input sequence length was limited to just 15 tokens. These constraints significantly hampered the model's ability to comprehend longer text inputs and capture their emotional context.

0.7 Model version 2

In this model, after researching and evaluating resource utilisation, I decided to push my resources to the limit. I increased the maximum sequence length to 140 tokens, which might be excessive but ensures the model can process complete sentences. Another significant change was increasing the batch size to 8192, which nearly maximises the available memory on an L2 GPU. My rationale was that processing more sentences simultaneously would enhance and accelerate learning. The remaining model architecture stayed the same. These modifications proved effective, as demonstrated by the following results:
Epoch 10/30 - accuracy: 0.9552 - loss: 0.1468 - val_accuracy: 0.9357 - val_loss: 0.2068 by epoch 30 we achieved
Epoch 30/30 accuracy: 0.9912 - loss: 0.0275 - val_accuracy: 0.9518 - val_loss: 0.1861 This, to me, is an excellent result after minimal adjustments, but this raised all kinds of red flags. I believed it was too good to be true. One thing I came across was that some text samples appeared in both testing and training sets - 7,767 lines, but this was negligible considering the total dataset contained 150,000 lines. I also converted the predictions from numeric values to text labels and examined the testing and training results. The next step is to push the training performance even further. I want to get it as close to 100% accuracy as possible.

0.8 Model version 3

This version made only a slight improvement in overall accuracy but showed significant gains in efficiency. In the previous model, it took around 20 epochs just to reach 95% testing accuracy, and it plateaued from that point until epoch 30, while training accuracy continued rising from 98% to 99%. In contrast, this version achieved the following results by epoch 10: accuracy: 0.9817, loss: 0.0433, validation accuracy: 0.9716, validation loss: 0.0702. The earlier version could not reach this level of testing accuracy even after 30 epochs.

A few key changes led to these improvements. I increased the dropout rate from 0.3 to 0.5. I reasoned that the model was overfitting, and training accuracy kept rising while validation accuracy stagnated. Increasing dropout helped break out of that, though it did not directly improve speed. Next, I switched the optimiser from Adam to AdamW, which gave a slight boost in accuracy and helped me reach 97% test accuracy more reliably.

The biggest efficiency gain came from adjusting the learning rate. Initially, I used 0.001, which proved too low. After trial and error, I settled on a learning rate of 0.026. If any higher, the model would not train properly. With this combination of changes, I achieved nearly identical training accuracy and a 2% gain in testing accuracy while reducing training time by two-thirds. The training accuracy is

about 1% lower than before, but the improved generalisation and speed made that a worthwhile tradeoff.

0.9 Additional testing

In the paper about sentiment analysis of hotel reviews, the authors documented using Nadam as their optimiser (Xu (2024)). Having not encountered this optimiser before, I decided to test it against AdamW in our model. Running 10 epochs with the same learning rate of 0.026, Nadam produced slightly inferior results: Epoch 10/10 accuracy: 0.9853 loss: 0.0390 val_accuracy: 0.9700 - val_loss: 0.0869 with a test accuracy of 0.9691. These results did not surpass the performance achieved with AdamW at a learning rate of 0.001 and 20 epochs as in version 2. To thoroughly evaluate Nadam, I conducted additional experiments with different learning rates. Tests with learning rates of 0.0001 and 0.05 also failed to outperform AdamW in accuracy and training efficiency. Based on these results, Nadam did not demonstrate improvements over AdamW for the sentiment analysis task. Results:

```
 Epoch 10/10 test acc: 0.9817 loss: 0.0433

- val_acc 0.9716 - val_loss: 0.0702
```

0.9.1 Example text processed:

1. ”i feel sure that s is right in that the despondency follows the suppression of anger” → Predicted: joy — Actual: joy

2. ”i least understand the one that with just a look can make me feel despised respected loved and inconsequential” → Predicted: anger — Actual: anger

3. ”i feel its a bit hostile given the fact that many of my peers are having babies and getting married” → Predicted: anger — Actual: anger

4. ”i could truly feel anymore sitting petrified on that ice cream parlor bench as my mother sold the last remaining bits of her crumpled humanity for a reserved seat in hell” → Predicted: fear — Actual: fear

5. ”i was asked to join my friend in a boat at mangochi lake malawi and when making a turn” → Predicted: fear — Actual: fear

6. ”i kneel down to stop the fifteenth public temper tantrum of the day and i can feel uptight suburban mothers wonder what kind of unfit pregnant teenager i must have been and how ive grown into a complete failure of a mom” → Predicted: fear — Actual: fear

7. ”im not blessed to feel fabulous during pregnancy however i at least am blessed to be able to get pregnant much too easy and to have healthy pregnancies” → Predicted: joy — Actual: joy

8. ”i did what i needed to do which was to feel miserable without a time limit” → Predicted: sadness — Actual: sadness

9. ”i feel kindof sad over it but im trying to make what i have work instead of feeling like omg i need a basic black shoe because i dont” → Predicted: sadness — Actual: sadness

10. ”i would have thought that being where i am is a promising sign for a creator writer however i feel more apprehensive than confident as my schedule will soon get hectic beginning in january” → Predicted: fear — Actual: fear

0.10 Conclusion

I implemented a GRU-based sentiment classification model that reached 97.16% validation accuracy. Key improvements came from increasing sequence length from 15 to 140 tokens, enlarging batch size to 8192, and optimising the learning rate to 0.026. Comparative testing showed AdamW outperformed both Adam and Nadam optimisers for this specific task. As this project progressed, I was able to get higher results in fewer epochs. After getting these results, I decided to revisit and investigate what was happening with the remaining errors why couldn't the model learn the final 2-3%? I focused on examining high confidence threshold errors and discovered they weren't completely off-base. These misclassifications were usually examples that I could see going both ways, such as texts that could reasonably be classified as either fear or anger. I believe unless we start using a pretrained model, we might have hit a ceiling at 97%, perhaps gaining at most a percentage point or two with further optimization. Some sentences simply contain too much nuance for the current model architecture to fully understand.

Here's some examples:

Table 1: High Confidence Errors in Emotion Classification Examples

Text	**True Label**	**Predicted**	**Conf.**
“i no doubt will also be angry at myself for feeling so helpless...”	sadness	fear	0.943
“i know for positive booklet when shes faced with a challenging selection...”	sadness	fear	0.911
“i have been hearing rumours that you have not been allowing people...”	anger	sadness	0.928
“i be on my term which seem pretty run of the mill but after...”	anger	sadness	0.952
“i blog because i feel much less inhibited in expressing myself...”	fear	sadness	0.960

Bibliography

Cerón Viveros, G., 2021. *Study and Analysis of Training Strategies to Improve the Reliability of Artificial Neural Networks*. Master's thesis, Politecnico di Torino.

Versloot, C., 2019. About loss and loss functions. URL `https://machinecurve.com/index.php/2019/10/04/about-loss-and-loss-functions`, accessed: 2025-04-22.

Xu, Z., 2024. Sentiment analysis of hotel comments based on lstm and gru. *Applied and Computational Engineering*, 38, 7–15.