Deep Learning Assignment 2

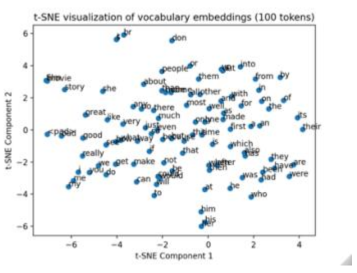
Kok Teng, Ng (1936360), nkokteng

MinJeong, Lee (1978925), minjlee

Question 4: Pretrained Embedding & Visualization

a. The file ***"data/word-embedding.txt"*** contains pre-trained word embeddings as known as GloVe, which are essentially numerical representations of words in a high-dimensional space. These embeddings are generated for a vocabulary set of 29,841 words, each ***represented*** as a ***dense***, ***100-dimensional vector***. These ***word embeddings*** are the ***product of*** a specific type of neural network as ***embedding neural network***. The primary function of this network is to transform sparse, high-dimensional word vector (***usually one-hot encoding vectors***) into a lower-dimensional, ***dense format***. This transformation is achieved through the learning process (such as ***Skip-Gram model or Continuous Bag-of-Words (CBOW)***), where the network adjusts the word vectors based on the contextual usage of words within a large corpus of text. The primary advantage of these dense vectors is ***their capability to capture semantic relationships between words***. This is achieved using ***cosine similarity***, a metric that ***measure the cosine of the angle between two vectors***. In the context of word embeddings, a ***high cosine similarity score*** between two-word vectors ***indicates*** that the words are used in ***similar contexts and having same direction and close to each other within the embedding (latent representation) space***, suggesting a close semantic relationship. Conversely, dissimilar words have vectors that diverge, resulting in a negative similarity score.
One of the most fascinating aspects of these word embedding is their ability to ***capture complex semantic relationship*** through ***simple vector arithmetic***. For instance, these embedding's vector arithmetic operation: ***"king" – "man" + "woman" = "queen"***. This operation demonstrates that the relationships between words, such as ***gender***, can be represented and manipulated within the embedding space.

b. Based on the bottom illustration, the t-Distribution Stochastic Neighbor Embedding (t-SNE) visualization is a technique used to visualize high-dimensional data (in our case, the high-dimensional data is regarding to the word representation vector) in a two- or three-dimensional map. This method allows us to inspect the structure of high-dimensional data more intuitively.



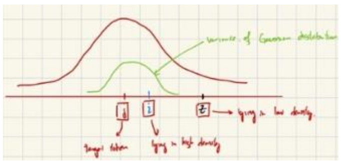
In the context of word representation vectors, tokens with similar semantics in the high-dimensional space, the goal of t-SNE will ensure and aim to map them closer together

Universität Mannheim, ng.kok.teng@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

Universität Mannheim, minjlee@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

in the lower-dimensional visualization. Investigate from the illustration, we could view that the pair of ***"film" and "movie"***, ***"i", "me", and "my"***, ***"a" and "an"***, and others are having the same semantic, and they were also being mapped close to each other in the lower-dimensional by using the method of t-SNE to plot them in the 2-Dimensional visualization. Additionally, the tokens of ***"they" and "i"*** are having different semantic since the semantics of ***"they"*** is regarding to many people and ***"i"*** is regarding to one people; therefore, both tokens are having different direction to each other as getting a large distance.

The measure of similarity between tokens is determined using an isotropic Gaussian distribution, which essentially investigates the similarity of tokens in the high-dimensional (embedding) space. The isotropic Gaussian $P(z_j \mid z_i) = \mathcal{N}(z_j \mid z_i, \sigma^2 \boldsymbol{I})$. Based on this Gaussian formula, we could plot the distribution of a given token ***"j"*** and accesses the density of other tokens within this distribution. If another token ***"i"*** lies within a high-density region of the distribution of token ***"j"***, it is close to ***"j"***.



During the computation of similarity, the variance of the Gaussian distribution plays a crucial role. It controls the spread of the distribution, thereby determining which tokens are considered neighbors or close to the target token ***"j"***. A small variance of Gaussian distribution results in a small group of data points being labeled as having similar embedding values or being close to each other. Conversely, a large variance might include a larger group of data points as having similar embedding values, but there might also include noise into the data. To determine the variance, needed to compute the Shannon Entropy H_i of the neighbor distribution $P(z_j \mid z_i)$. Intuitively, this represents the number of effective neighbors of data point ***"i"*** in bits. Compute the perplexity of the distribution from the Shannon Entropy as 2^{H_i}. Perplexity could be thought of as a measure of how well a probability distribution predicts a sample. Use the target perplexity to determine the variance of the ***t-SNE's isotropic Gaussian***.

Additionally, we are utilizing the ***Uniform Manifold Approximation and Projection (UMAP)*** to reduce the dimensionality of word embeddings for visualization purposes. UMAP is favored for its consistency across different runs, unlike t-SNE which can produce different visualizations with the same data due to its gradient descent procedure that can sometimes get stuck in local minima.

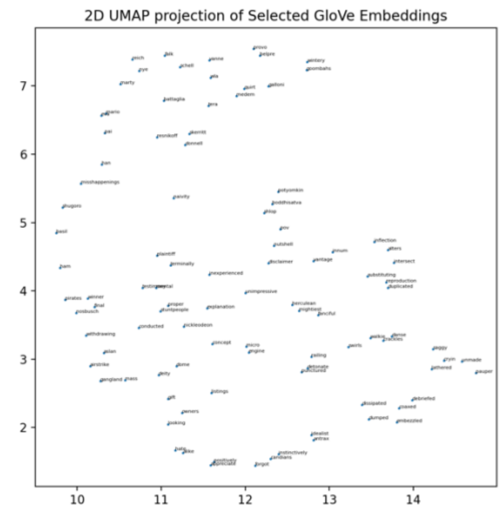
Due to ***UMAP*** uses spectral embedding to compute the probability of neighboring data points, making it more stable and less likely to be trapped in local minima. This stability in crucial ensuring the reliability of our visualizations.

In our exploration, we used ***UMAP*** to project word embeddings into 2-Dimensional space. We observed that semantically similar tokens, such as ***"positively" and "appreciate"***, ***"reproduction" and "duplicated",*** and others were mapped close to each other in the ***UMAP*** visualization. This demonstrates ***UMAP***'s effectiveness in

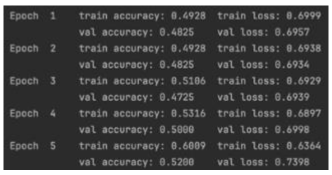
Universität Mannheim, ng.kok.teng@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

Universität Mannheim, minjlee@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

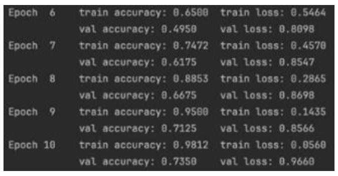
preserving semantic relationships when reducing the dimensionality of word embeddings.



c. During the training with the specified hyperparameters on the ***SimpleRNN*** model for 10 epochs, we observed the potential ***overfitting*** issue. This was indicated from the low loss on the training set and the comparatively high loss on the validation set. Therefore, we can infer that the model's performance on the validation set might be poor. This could also reflect in the t-SNE visualization by using the thought vector of each review from validation set, which appear more dispersed for the validation set compared to the training set according to the label of the review. Additionally, we could also investigate from the result as the model was starting to overfitting after the epoch 3 as the loss from the validation set was starting to increase but the loss from training set was still decreasing.



```
Epoch  1    train accuracy: 0.4928  train loss: 0.6999
            val accuracy: 0.4825    val loss: 0.6957
Epoch  2    train accuracy: 0.4928  train loss: 0.6938
            val accuracy: 0.4825    val loss: 0.6934
Epoch  3    train accuracy: 0.5106  train loss: 0.6929
            val accuracy: 0.4725    val loss: 0.6939
Epoch  4    train accuracy: 0.5316  train loss: 0.6897
            val accuracy: 0.5000    val loss: 0.6998
Epoch  5    train accuracy: 0.6009  train loss: 0.6364
            val accuracy: 0.5200    val loss: 0.7398
```

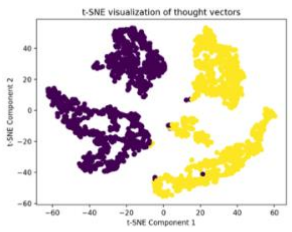


```
Epoch  6    train accuracy: 0.6500  train loss: 0.5464
            val accuracy: 0.4950    val loss: 0.8098
Epoch  7    train accuracy: 0.7472  train loss: 0.4570
            val accuracy: 0.6175    val loss: 0.8547
Epoch  8    train accuracy: 0.8853  train loss: 0.2865
            val accuracy: 0.6675    val loss: 0.8698
Epoch  9    train accuracy: 0.9500  train loss: 0.1435
            val accuracy: 0.7125    val loss: 0.8566
Epoch 10    train accuracy: 0.9812  train loss: 0.0560
            val accuracy: 0.7350    val loss: 0.9660
```

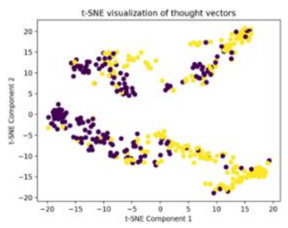
In the context of ***t-SNE*** visualization (by using ***tsne_thought()*** function), we will only use the last hidden states (also known as ***"thought vectors"***, which is the vector of prediction with respect to the reviews before fitting into the logit function) of the trained model, and project into a lower-dimensional space as 2-dimensional.
During the feedforward process, reviews sharing the same label are expected to yield similar or close thought vectors. This is because these vectors, which represent the final hidden state, compress the information from all previous time steps in the sequence.
When these thought vectors are passed through the logit function, they are likely to produce the same label. Therefore, in the lower-dimensional space created by ***t-SNE***, reviews with similar content (will also having the similar thought vector) are expected to be mapped closer together in the lower-dimensional. This visualization could provide insight how the model is internally processing and categorizing the reviews.

Universität Mannheim, ng.kok.teng@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

Universität Mannheim, minjlee@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)



(a) training set



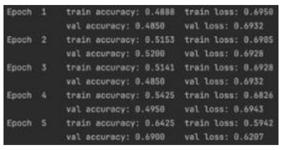
(b) validation set

Based on the above t-SNE visualization, it appears that the trained model is performing as expected. Reviews with the same label tend to have similar "thought vectors" produced by the feedforward RNN sequence model. When these vectors are projected into a lower-dimensional and visualized using t-SNE, reviews with the same labels are mapped closer to each other in the lower-dimensional in the training set. This suggests that the model is effectively compressing and summarizing the information from all previous time steps in the sequence.

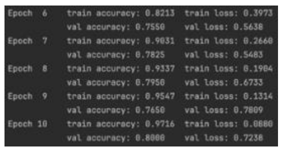
However, we could also investigate the model's performance is having the signal of ***overfitting***, as the model doesn't perform as well on the validation set. This could be due to the model doesn't effectively capturing and compressing the information from all previous time steps in the sequence to generate the thought vectors. Therefore, reviews with the same label do not always generate similar or close vector representation (also called ***"thought vector"***) and predict it wrongly. This results in a sparse distribution of data points in the ***t-SNE*** visualization, indicating that the model struggles to map reviews with similar labels closer to each other within the lower-dimensional due to there are having dissimilar thought vector with the same label of review. Therefore, the model can't promise that it could generalize well on those unseen data.

d. In this question, the pretrained ***GloVe embeddings model*** is incorporated as the ***embedding layer*** within the custom ***SimpleLSTM sequence model***. This integration enables the model to transform categorical input data, represented as ***One-Hot Encoding***, into ***meaningful word embeddings***.

   Upon loading the pretrained embeddings, the model is subsequently trained on a specific task – in this case, sentiment analysis. During this training phase, the weights of ***the embedding layer are fine-tuned***. This fine-tuning process is accomplished by computing the ***gradient of the embedding layer's weights with respect to the loss function.*** These gradients are then used to ***update and adjust the weights***, enabling them to better align with the specific task at hand. This fine-tuning process allows the model to leverage the general language understanding capabilities of the pretrained embeddings while also adapting to the nuances of the specific task. It is often observed that fine-tuning the embedding layer leads to ***better performance compared to*** the upcoming case where the embedding layer is ***not fine-tuned***.



```
Epoch  1   train accuracy: 0.4888  train loss: 0.6950
           val accuracy: 0.4850    val loss: 0.6932
Epoch  2   train accuracy: 0.5153  train loss: 0.6985
           val accuracy: 0.5200    val loss: 0.6928
Epoch  3   train accuracy: 0.5141  train loss: 0.6928
           val accuracy: 0.4850    val loss: 0.6932
Epoch  4   train accuracy: 0.5425  train loss: 0.6826
           val accuracy: 0.4950    val loss: 0.6943
Epoch  5   train accuracy: 0.6425  train loss: 0.5942
           val accuracy: 0.6900    val loss: 0.6207
```



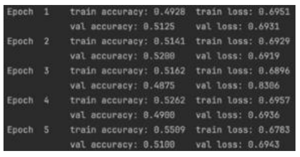
```
Epoch  6   train accuracy: 0.8213  train loss: 0.3973
           val accuracy: 0.7550    val loss: 0.5638
Epoch  7   train accuracy: 0.9031  train loss: 0.2660
           val accuracy: 0.7825    val loss: 0.5483
Epoch  8   train accuracy: 0.9337  train loss: 0.1984
           val accuracy: 0.7950    val loss: 0.6733
Epoch  9   train accuracy: 0.9547  train loss: 0.1314
           val accuracy: 0.7650    val loss: 0.7809
Epoch 10   train accuracy: 0.9716  train loss: 0.0880
           val accuracy: 0.8000    val loss: 0.7238
```

Universität Mannheim, ng.kok.teng@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

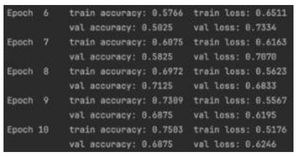
Universität Mannheim, minjlee@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

The results presented indicate that the model is trained effectively on the training set, as evidenced by the low loss and high accuracy. However, it appears that the model begins to ***overfit*** after the fourth epoch. This suggested by the increasing validation loss, despite the model's continued high performance on the training set. This indicates that the model is becoming excessively specialized to the training data, thereby losing its capability to generalize effectively on unseen data.

e. In this question, the ***pretrained GloVe embeddings model*** is once again employed as ***embedding layer*** within the ***custom SimpleLSTM sequence model***. However, in contrast to the previous case, the parameters of this embedding layer are ***not subject to fine-tuning***. This implies that the model continues to ***leverage the general language understanding*** derived from the pretrained embeddings to ***perform the current task***.



```
Epoch  1    train accuracy: 0.4928  train loss: 0.6951
            val accuracy: 0.5125    val loss: 0.6931
Epoch  2    train accuracy: 0.5141  train loss: 0.6929
            val accuracy: 0.5200    val loss: 0.6919
Epoch  3    train accuracy: 0.5162  train loss: 0.6896
            val accuracy: 0.4875    val loss: 0.8306
Epoch  4    train accuracy: 0.5262  train loss: 0.6957
            val accuracy: 0.4900    val loss: 0.6936
Epoch  5    train accuracy: 0.5509  train loss: 0.6783
            val accuracy: 0.5100    val loss: 0.6943
```

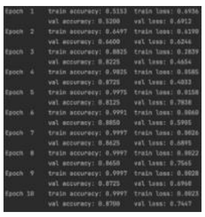


```
Epoch  6    train accuracy: 0.5766  train loss: 0.6511
            val accuracy: 0.5025    val loss: 0.7334
Epoch  7    train accuracy: 0.6075  train loss: 0.6163
            val accuracy: 0.5825    val loss: 0.7070
Epoch  8    train accuracy: 0.6972  train loss: 0.5623
            val accuracy: 0.7125    val loss: 0.6833
Epoch  9    train accuracy: 0.7309  train loss: 0.5567
            val accuracy: 0.6875    val loss: 0.6195
Epoch 10    train accuracy: 0.7503  train loss: 0.5176
            val accuracy: 0.6875    val loss: 0.6246
```

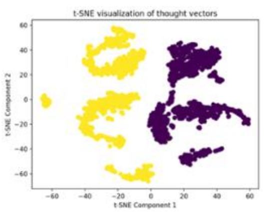
Based on the results presented, it can be inferred that the model may be ***underfitting***. This is because the parameters of the embedding layer are not adjusted during the training process to better suit the specific task. Therefore, the model relies solely on the general language understanding to execute the task. During the backward propagation phase, the model computes the gradient of the embedding layer's parameter with respect to the loss function and updates these parameters to minimize the loss generated by the model during forward pass. However, without fine-tuning these parameters, the model may struggle to generalize effectively to unseen data and training data, leading to potential underfitting problem.

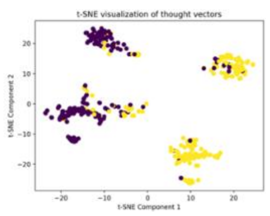
Question 5: Exploration

a. There are having seven different setups based on the ***SimpleLSTM model*** as blueprint.



```
Epoch  1    train accuracy: 0.5153  train loss: 0.6936
            val accuracy: 0.5200    val loss: 0.6912
Epoch  2    train accuracy: 0.6497  train loss: 0.6190
            val accuracy: 0.6600    val loss: 0.6246
Epoch  3    train accuracy: 0.8825  train loss: 0.2839
            val accuracy: 0.8225    val loss: 0.4654
Epoch  4    train accuracy: 0.9825  train loss: 0.0585
            val accuracy: 0.8725    val loss: 0.4033
Epoch  5    train accuracy: 0.9975  train loss: 0.0158
            val accuracy: 0.8125    val loss: 0.7838
Epoch  6    train accuracy: 0.9991  train loss: 0.0060
            val accuracy: 0.8850    val loss: 0.5905
Epoch  7    train accuracy: 0.9997  train loss: 0.0026
            val accuracy: 0.8625    val loss: 0.6895
Epoch  8    train accuracy: 0.9997  train loss: 0.0022
            val accuracy: 0.8650    val loss: 0.7565
Epoch  9    train accuracy: 0.9997  train loss: 0.0020
            val accuracy: 0.8725    val loss: 0.6960
Epoch 10    train accuracy: 0.9997  train loss: 0.0023
            val accuracy: 0.8700    val loss: 0.7447
```





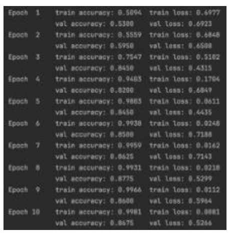
Performance ***t-SNE training set*** ***t-SNE validation set***

Bidirectional LSTM with 0.0 dropout

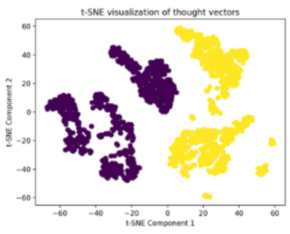
The model, which is configured with a ***Bidirectional LSTM cell*** and a ***0,0-dropout rate*** begins to exhibit signs of ***overfitting after the fourth epoch.*** This is evidenced by the model's superior performance on the training set, contrasted with its inability to generalize effectively on the validation set.

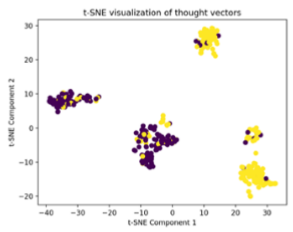
The t-SNE visualization of the model's thought vector further substantiates this observation. In the training set, the model successfully generates closely mapped representations for reviews with the same label in the lower-dimensional space. However, it struggles to consistently cluster reviews with the same label in proximity,

indicating a lack of generalization. This issue becomes more pronounced in the validation set, where the model fails to map similar reviews closely together.
This discrepancy between the training and validation sets is clear indication of the model's tendency to overfit. In other words, while the model is highly adept at learning the specific characteristic of the training data, it struggles to apply this knowledge effectively to unseen data, thereby compromising its predictive accuracy.



```
Epoch  1    train accuracy: 0.5094  train loss: 0.6977
            val accuracy: 0.5300    val loss: 0.6923
Epoch  2    train accuracy: 0.5559  train loss: 0.6848
            val accuracy: 0.5950    val loss: 0.6508
Epoch  3    train accuracy: 0.7547  train loss: 0.5102
            val accuracy: 0.8450    val loss: 0.4315
Epoch  4    train accuracy: 0.9483  train loss: 0.1704
            val accuracy: 0.8200    val loss: 0.6049
Epoch  5    train accuracy: 0.9883  train loss: 0.0611
            val accuracy: 0.8450    val loss: 0.4435
Epoch  6    train accuracy: 0.9938  train loss: 0.0248
            val accuracy: 0.8500    val loss: 0.7188
Epoch  7    train accuracy: 0.9959  train loss: 0.0162
            val accuracy: 0.8625    val loss: 0.7143
Epoch  8    train accuracy: 0.9931  train loss: 0.0218
            val accuracy: 0.8775    val loss: 0.5299
Epoch  9    train accuracy: 0.9966  train loss: 0.0112
            val accuracy: 0.8600    val loss: 0.5964
Epoch 10    train accuracy: 0.9981  train loss: 0.0081
            val accuracy: 0.8675    val loss: 0.5266
```





Performance ***t-SNE training set*** ***t-SNE validation set***

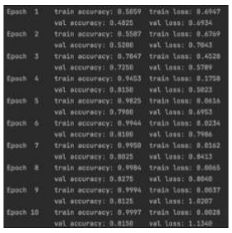
Bidirectional LSTM with 0.8 dropout

The model, configured with a ***Bidirectional LSTM cell*** and a ***0,8-dropout rate***. This dropout technique, which randomly deactivates a subset of neurons during each training epoch, effectively reduces the model's complexity by preventing certain neurons from updating their parameters. This strategy aids in mitigating overfitting.
However, the implementation of dropout, signs of overfitting are still observed in this model, but it has ***less overfitting compared to the previous case*** as the validation set's loss are lower than it. This is evident in the t-SNE visualization of the training set, where the model adeptly generates similar thought vectors for reviews with the same label, mapping them closely in the lower-dimensional space. This indicates that the model has learned the training set well. However, it struggles to consistently cluster reviews within the same label closely together. When we examine the t-SNE visualization of the validation set, it becomes clear that the model struggles slightly to generalize its learning to unseen data. Although the model's performance on the validation set has ***improved compared to the model without dropout***, there are still instances where data points with different labels are mapped closely together in the lower-dimensional space. This suggests that while the ***dropout technique has enhanced*** the model's capability to ***generalize***.
We also compare the performance of model configured with ***Unidirectional and Bidirectional LSTM cell***, each either with ***0,0- or 0,8-dropout rate***. The results indicate that the ***Bidirectional LSTM models outperform their Unidirectional counterparts***, irrespective of the dropout rate. This ***performance difference*** can be attributed to the inherent ***characteristics of LSTM cells***.
Unidirectional LSTM models process information in a ***single direction*** along the input sequence data. However, in the context of our task, which involves processing review texts, ***both preceding and succeeding tokens can influence the semantics of*** **a sentence**. Unidirectional LSTM model is ***unable to capture*** this bidirectional context, which may ***limit their performance***.
In contrast, ***Bidirectional LSTM models*** process the sentence in ***both directions***, thereby capturing a ***more comprehensive understanding*** of the sequence. The

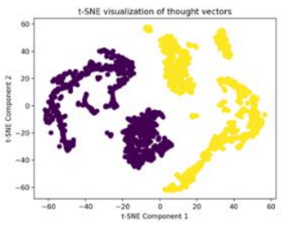
Universität Mannheim, ng.kok.teng@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

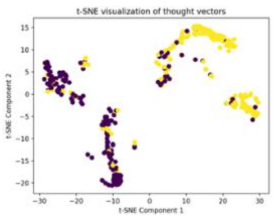
Universität Mannheim, minjlee@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

bidirectional processing capability is likely ***why Bidirectional LSTM models outperform Unidirectional LSTM model***.



```
Epoch  1   train accuracy: 0.5059  train loss: 0.6947
           val accuracy: 0.4825    val loss: 0.6934
Epoch  2   train accuracy: 0.5587  train loss: 0.6769
           val accuracy: 0.5200    val loss: 0.7043
Epoch  3   train accuracy: 0.7047  train loss: 0.4528
           val accuracy: 0.7250    val loss: 0.5789
Epoch  4   train accuracy: 0.9453  train loss: 0.1758
           val accuracy: 0.8150    val loss: 0.5023
Epoch  5   train accuracy: 0.9825  train loss: 0.0616
           val accuracy: 0.7900    val loss: 0.6953
Epoch  6   train accuracy: 0.9944  train loss: 0.0234
           val accuracy: 0.8100    val loss: 0.7986
Epoch  7   train accuracy: 0.9950  train loss: 0.0162
           val accuracy: 0.8025    val loss: 0.8413
Epoch  8   train accuracy: 0.9984  train loss: 0.0065
           val accuracy: 0.8275    val loss: 0.8040
Epoch  9   train accuracy: 0.9994  train loss: 0.0037
           val accuracy: 0.8125    val loss: 1.0207
Epoch 10   train accuracy: 0.9997  train loss: 0.0028
           val accuracy: 0.8150    val loss: 1.1340
```

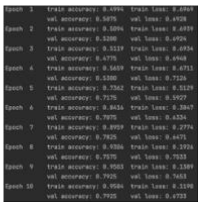




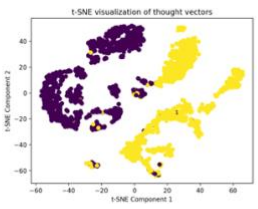
Performance ***t-SNE training set*** ***t-SNE validation set***

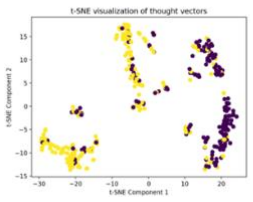
Unidirectional LSTM with 0.0 dropout

The ***Unidirectional LSTM model with a 0,0-dropout rate*** demonstrates good generalization on the training set, as evidenced by the t-SNE visualization. Reviews with the same label are mapped closely in the lower-dimensional space, indicating that the model has learned to generate similar thought vectors for these reviews. However, on the validation set, different labels are mapped closely together, suggestion that the model struggles to generalize well to unseen data.



```
Epoch  1   train accuracy: 0.4994  train loss: 0.6969
           val accuracy: 0.5075    val loss: 0.6928
Epoch  2   train accuracy: 0.5094  train loss: 0.6939
           val accuracy: 0.5200    val loss: 0.6924
Epoch  3   train accuracy: 0.5119  train loss: 0.6934
           val accuracy: 0.4775    val loss: 0.6948
Epoch  4   train accuracy: 0.5659  train loss: 0.6711
           val accuracy: 0.5300    val loss: 0.7126
Epoch  5   train accuracy: 0.7362  train loss: 0.5129
           val accuracy: 0.7175    val loss: 0.5927
Epoch  6   train accuracy: 0.8416  train loss: 0.3847
           val accuracy: 0.7075    val loss: 0.6334
Epoch  7   train accuracy: 0.8959  train loss: 0.2774
           val accuracy: 0.7825    val loss: 0.6471
Epoch  8   train accuracy: 0.9306  train loss: 0.1926
           val accuracy: 0.7575    val loss: 0.7533
Epoch  9   train accuracy: 0.9503  train loss: 0.1389
           val accuracy: 0.7925    val loss: 0.7653
Epoch 10   train accuracy: 0.9584  train loss: 0.1198
           val accuracy: 0.7925    val loss: 0.6733
```

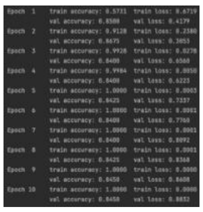




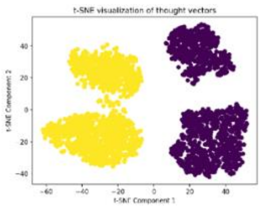
Performance ***t-SNE training set*** ***t-SNE validation set***

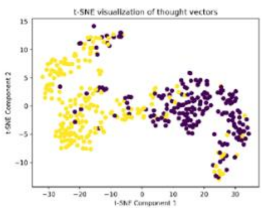
Unidirectional LSTM with 0.8 dropout

The ***Unidirectional LSTM model with a 0,8-dropout rate*** appears to ***slightly underfitting***, as indicated by t-SNE visualization of the training set. The model struggles to generate similar thought vectors for reviews with the same label, resulting in mixed classes within the same cluster in the lower-dimensional space. This issue could be due to high dropout rate, which deactivates some neurons during training, potentially preventing the model from learning necessary patterns from the data. This underfitting is further confirmed by the model's performance on the validation set, where it fails to accurately map reviews with the same labels.



```
Epoch  1   train accuracy: 0.5731  train loss: 0.6719
           val accuracy: 0.8500    val loss: 0.4179
Epoch  2   train accuracy: 0.9128  train loss: 0.2380
           val accuracy: 0.8675    val loss: 0.3055
Epoch  3   train accuracy: 0.9928  train loss: 0.0278
           val accuracy: 0.8400    val loss: 0.6560
Epoch  4   train accuracy: 0.9984  train loss: 0.0050
           val accuracy: 0.8400    val loss: 0.6223
Epoch  5   train accuracy: 1.0000  train loss: 0.0003
           val accuracy: 0.8425    val loss: 0.7337
Epoch  6   train accuracy: 1.0000  train loss: 0.0001
           val accuracy: 0.8400    val loss: 0.7760
Epoch  7   train accuracy: 1.0000  train loss: 0.0001
           val accuracy: 0.8400    val loss: 0.8092
Epoch  8   train accuracy: 1.0000  train loss: 0.0001
           val accuracy: 0.8425    val loss: 0.8368
Epoch  9   train accuracy: 1.0000  train loss: 0.0000
           val accuracy: 0.8450    val loss: 0.8608
Epoch 10   train accuracy: 1.0000  train loss: 0.0000
           val accuracy: 0.8450    val loss: 0.8832
```





Performance ***t-SNE training set*** ***t-SNE validation set***

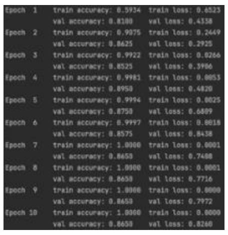
Unidirectional GRU with 0.0 dropout

The model, configured with a ***Unidirectional GRU cell*** and a ***0,0-dropout rate***, demonstrates ***superior performance on the training set compared to models utilizing either Bidirectional or Unidirectional LSTM cells***. This is substantiated by the t-SNE visualization, where the Unidirectional GRU model generates ***thought vectors for***

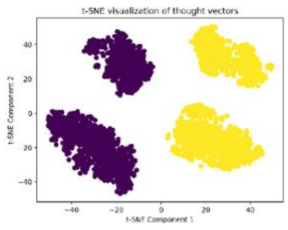
Universität Mannheim, ng.kok.teng@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

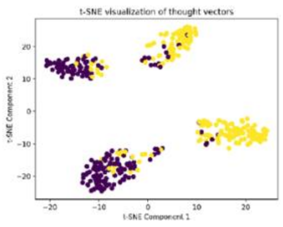
Universität Mannheim, minjlee@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

reviews with the same label that are more closely clustered in the lower-dimensional space compared to those generated by the LSTM cell models. This suggests that the GRU model is more effective in generalizing the thought vectors for reviews with the same label within the context of the current task.
However, signs of overfitting begin to emerge after the second epoch too. This is indicated by a decreasing loss for training set, contrasted with an increasing loss for the validation set. The t-SNE visualization of the validation set further confirms this, as data points with the same label are slightly more dispersed.



```
Epoch  1    train accuracy: 0.5934  train loss: 0.6523
            val accuracy: 0.8100    val loss: 0.4338
Epoch  2    train accuracy: 0.9075  train loss: 0.2449
            val accuracy: 0.8625    val loss: 0.2925
Epoch  3    train accuracy: 0.9922  train loss: 0.0266
            val accuracy: 0.8525    val loss: 0.3986
Epoch  4    train accuracy: 0.9981  train loss: 0.0053
            val accuracy: 0.8950    val loss: 0.4820
Epoch  5    train accuracy: 0.9994  train loss: 0.0025
            val accuracy: 0.8750    val loss: 0.6809
Epoch  6    train accuracy: 0.9997  train loss: 0.0018
            val accuracy: 0.8575    val loss: 0.8438
Epoch  7    train accuracy: 1.0000  train loss: 0.0001
            val accuracy: 0.8650    val loss: 0.7408
Epoch  8    train accuracy: 1.0000  train loss: 0.0001
            val accuracy: 0.8650    val loss: 0.7716
Epoch  9    train accuracy: 1.0000  train loss: 0.0000
            val accuracy: 0.8650    val loss: 0.7972
Epoch 10    train accuracy: 1.0000  train loss: 0.0000
            val accuracy: 0.8650    val loss: 0.8260
```

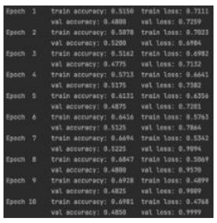




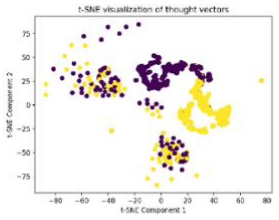
Performance ***t-SNE training set*** ***t-SNE validation set***

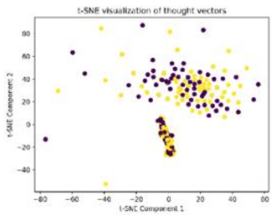
Bidirectional GRU with 0.8 dropout

The model, configured with ***a Bidirectional GRU cell*** and a ***0,8-dropout rate***, demonstrates impressive performance on the training set. This is substantiated by the t-SNE visualization, where the model generates thought vectors for reviews with the same label that are closely clustered in the lower-dimensional space. This means that the model is effectively learning to generate similar thought vectors for reviews with the same label, thereby forming distinct clusters. However, signs of ***overfitting*** begin to emerge after the second epoch.



```
Epoch  1    train accuracy: 0.5150  train loss: 0.7111
            val accuracy: 0.4800    val loss: 0.7259
Epoch  2    train accuracy: 0.5078  train loss: 0.7023
            val accuracy: 0.5200    val loss: 0.6984
Epoch  3    train accuracy: 0.5162  train loss: 0.6982
            val accuracy: 0.4775    val loss: 0.7132
Epoch  4    train accuracy: 0.5713  train loss: 0.6641
            val accuracy: 0.5175    val loss: 0.7382
Epoch  5    train accuracy: 0.6131  train loss: 0.6356
            val accuracy: 0.4875    val loss: 0.7201
Epoch  6    train accuracy: 0.6416  train loss: 0.5763
            val accuracy: 0.5125    val loss: 0.7864
Epoch  7    train accuracy: 0.6694  train loss: 0.5342
            val accuracy: 0.5225    val loss: 0.9094
Epoch  8    train accuracy: 0.6847  train loss: 0.5069
            val accuracy: 0.4800    val loss: 0.9570
Epoch  9    train accuracy: 0.6928  train loss: 0.4899
            val accuracy: 0.4825    val loss: 0.9909
Epoch 10    train accuracy: 0.6981  train loss: 0.4768
            val accuracy: 0.4850    val loss: 0.9999
```





Performance ***t-SNE training set*** ***t-SNE validation set***

Unidirectional Elman (RNN) with 0.0 dropout

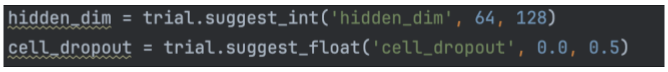
The model, configured with ***Unidirectional Elman* cell** and ***0,0-dropout rate***, having the least performance, and indicating a case of underfitting model. This suggests that the model is too simplistic to effectively handle the complexity of the current task. This is evident from the accuracy and loss metrics on both the validation and training sets, with the accuracy less than 0,7 and loss exceeding 0,45. The t-SNE visualization of the training and validation sets further confirms this. Reviews with different labels have similar thought vectors and are closely mapped in the lower-dimensional space. Conversely, reviews with the same label have dissimilar though vector and are mapped further apart.
In conclusion, our comprehensive analysis indicates that the model configured with ***a Bidirectional GRU cell,*** and ***a 0,8-dropout rate*** emerges as the ***most optimal solution*** for the task at hand, ***outperforming*** the ***other six model configurations***. This conclusion is primarily drawn from its superior performance during the training and

Universität Mannheim, ng.kok.teng@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

Universität Mannheim, minjlee@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

validation phase and the t-SNE visualization. The t-SNE visualization of the thought vector for this mode, for both the training and validation sets, provides further evidence of its effectiveness. The visualization reveals that reviews sharing the same label are closely mapped to each other in the lower-dimensional space. This clustering pattern is a clear indication of the model's capability to effectively generalize and distinguish between different classes. In contrast, other models with ***different hyperparameters are having slightly bad performance***, further underscoring the effectiveness of the chosen hyperparameter setup for the Bidirectional GRU model. Therefore, we recommend the use of a ***Bidirectional GRU cell with 0,8-dropout rate*** for this current task.

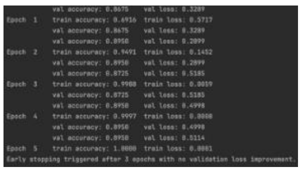
b. In our endeavor to optimize the performance of our ***Bidirectional GRU cell model***, we have employed the ***Optuna*** framework for hyperparameter tuning. This process involves conducting 20 trials to ***identify the optimal hyperparameters*** that ***yield the lowest validation loss***. The loss function used for this purpose is ***Binary Cross Entropy***.



```
hidden_dim = trial.suggest_int('hidden_dim', 64, 128)
cell_dropout = trial.suggest_float('cell_dropout', 0.0, 0.5)
```

To ***mitigate*** the risk of ***overfitting***, we have incorporated ***early stopping mechanism***. This is particularly crucial as our investigations revealed a tendency for models to overfit the training set after few epochs, we set the ***patient as 3*** if the validation loss ***doesn't decrease more than 3 times*** then we ***pause the training*** and ***take the model's stat (weights)*** as returned model to ***test on testing set***. The hyperparameters under consideration during this tuning process include the ***hidden dimension and dropout rate***. This comprehensive approach ensures effective optimization of our model's performance.

The ***optimal hyperparameters*** identified for our ***Bidirectional GRU cell sequence model*** is ***hidden layer dimension of 64 and 0,00421-dropout rate***. These settings yield the ***lowest validation loss***.



```
            val accuracy: 0.8675    val loss: 0.3289
Epoch  1    train accuracy: 0.6916  train loss: 0.5717
            val accuracy: 0.8675    val loss: 0.3289
            val accuracy: 0.8958    val loss: 0.2899
Epoch  2    train accuracy: 0.9491  train loss: 0.1452
            val accuracy: 0.8958    val loss: 0.2899
            val accuracy: 0.8725    val loss: 0.5185
Epoch  3    train accuracy: 0.9908  train loss: 0.0059
            val accuracy: 0.8725    val loss: 0.5185
            val accuracy: 0.8958    val loss: 0.4998
Epoch  4    train accuracy: 0.9997  train loss: 0.0008
            val accuracy: 0.8958    val loss: 0.4998
            val accuracy: 0.8958    val loss: 0.5114
Epoch  5    train accuracy: 1.0000  train loss: 0.0001
Early stopping triggered after 3 epochs with no validation loss improvement.
```

The model training process, as depicted above, demonstrates an interesting pattern. In the ***initial epoch***, the model achieves a ***validation loss of 0,328***, which further reduces to ***0,2899 in the second epoch***. However, ***post*** the second epoch, the model ***begins to overfit***, leading to ***an increase in validation loss***. This overfitting indicates that the model is not generalizing well to the validation set, resulting in a ***higher loss***.

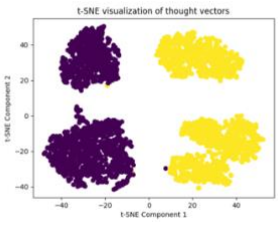
To mitigate this issue, we employ an ***early abort mechanism***. This mechanism monitors the ***validation loss*** and ***halts*** the training process if the ***validation loss doesn't improve*** over a certain number of epochs. In our setting, the training is stopped if there is no improvement in validation loss for ***three consecutive epochs***.

The ***parameters*** of the model at the point of ***lowest validation loss*** are saved in a ***PyTorch file***. This allows us to later load the model with the optimal parameters, ensuring that we utilize the most effective version of the model for our predictions. This approach helps in preventing utilize the overfitting model to generalize on unseen data

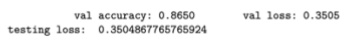
Universität Mannheim, ng.kok.teng@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

Universität Mannheim, minjlee@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

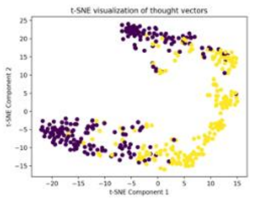
and achieving a balance between bias and variance, thereby improving the model's performance on unseen data.



The above t-SNE visualization of the thought vectors generated from ***the training set*** using these ***optimal hyperparameters and model's parameters with least validation loss within the training epoch*** provides further insights. It is evident that the model can generate similar thought vector for reviews that share the same label. Conversely, dissimilar thought vectors are generated for reviews with different labels. However, there are few instances where similar thought vectors are generated for reviews with different labels. This is an expected outcome considering our use of early stopping based on validation loss. The model used for this visualization is the one that achieved the lowest validation loss, but the training accuracy and loss haven't reached their optimal score. While it may not be perfectly accurate, it strikes a balance between avoiding overfitting and underfitting, thereby ensuring a robust model performance.



```
         val accuracy: 0.8650        val loss: 0.3505
testing loss:  0.3504867765765924
```



Testing set accuracy and validation and t-SNE visualization

When it comes to the testing set, the model achieves a ***testing accuracy score of 0,8650*** and ***testing loss of 0,3505***. These results are ***slightly lower than those achieved on the validation set***, which is expected as the model has not seen the testing data during training. The t-SNE visualization of the thought vectors generated from the testing set would likely show ***similar pattern to the validation set***, with the model generating similar thought vectors for reviews that share the same label and dissimilar thought vectors for reviews with different labels. However, given the slightly lower performance on the testing set, we could observe a slightly higher degree of overlap between thought vectors of different labels from the t-SNE visualization.

Universität Mannheim, ng.kok.teng@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)

Universität Mannheim, minjlee@students.uni-mannheim.de, Mannheim Master in Data Science (MMDS)