The Approach

* I employed a Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN)
* The motivation was to circumvent the Bayesian model's limitations, notably its reliance on manually extracted features such as word stemming and sentiment analysis.
* I envisage that a deep learning model, capable of autonomously extracting features from raw text, would be a more robust model
* LSTMs are capable of selectively remembering and forgetting information through their architecture, which includes memory cells and gates (input, forget, and output gates).
* These gates regulate the flow of information. Such an arrangement allows LSTMs not only to maintain crucial contextual information across long pasages of text but also to dynamically adapt to new inputs, effectively capturing the intricacies and nuances of natural language.
* The network effectively learns which information might be needed later on in a sequence and when that information is no longer needed. For instance, in the context of natural language processing, the network can learn grammatical dependencies.
* This ability allows them to preserve essential contextual data over long text sequences and adapt dynamically to new information, thereby by theory, able to capture the subtleties and complexities of natural language.
* Configuring the LSTM RNN model involved making thoughtful decisions regarding its architecture, such as selecting the appropriate number of layers, setting dropout rates, and choosing the type of word embeddings.
* I predict to gain a deeper comprehension of textual context and sentiment, achieving a level of feature learning sophistication unattainable with the Bayesian methodology.
* Bi-directional LSTM model captures patterns in past and future contexts, at the cost of adding comlexity to double the parameters.
* Implemented a deep learning model with an input layer, embedding layer, bi-directional LSTM layer and output layer.
* To help reduce overfitting, I implenmented dropout to the LSTM layer and regularsation to the output layer.

The architecture of a deep learning model for NLP sentiment analysis can indeed seem complex relative to the dataset size. However, the complexity of the model should be appropriate for the task's requirements and the data's complexity, not just the number of entries.

For a dataset with 1,200 entries, here are a few points to consider:

1. \*\*Overfitting\*\*: A model with a large number of parameters is more prone to overfitting, especially when the dataset is small. Regularization techniques like dropout are essential to prevent this.

2. \*\*Embedding Layer\*\*: The embedding layer's size (1,280,000 parameters) indicates a large vocabulary. For smaller datasets, you might not need such a large vocabulary. It could be more efficient to use pre-trained embeddings like Word2Vec or GloVe or reduce the `input\_dim` if you have many rare words that do not contribute much to the sentiment analysis.

3. \*\*Bidirectional LSTM\*\*: While a Bidirectional LSTM captures patterns from both past and future data context, it also doubles the number of parameters of a regular LSTM. Consider whether this complexity is warranted by the predictive power it provides.

4. \*\*Model Simplification\*\*: For smaller datasets, a simpler model may be sufficient. You could experiment with reducing the embedding dimension, using a regular (unidirectional) LSTM, or even trying simpler models like Convolutional Neural Networks (CNNs) for text, which can be quite effective with fewer parameters.

5. \*\*Transfer Learning\*\*: Leveraging transfer learning by using a pre-trained model as a starting point can allow you to benefit from knowledge gained from larger datasets while fine-tuning on your specific small dataset.

6. \*\*Data Augmentation\*\*: Techniques to artificially expand the dataset can be beneficial, especially if the model is complex. For text, this might involve synonym replacement, back-translation, or other NLP-specific augmentations.

7. \*\*Parameter Tuning\*\*: Given the relatively small dataset size, extensive hyperparameter tuning (e.g., grid search, random search, Bayesian optimization) is important. It's often more valuable than increasing model complexity.

In summary, it might be worthwhile to simplify the model or apply other strategies to fit the small dataset better while still capturing the nuances necessary for accurate sentiment analysis. You can also monitor validation performance to decide if the model's complexity is justified. If the simpler models perform similarly or better, they are likely more appropriate for your dataset.

References

*https://en.wikipedia.org/wiki/Long\_short-term\_memory*

*Russel . Norvig - Artifical Intelligence – A Modern Approach – 4th edition – Chapter 22.6.2*

*Russel . Norvig - Artifical Intelligence – A Modern Approach – 4th edition – Chapter 25.2.3*

[*https://towardsdatascience.com/multi-class-text-classification-with-lstm-1590bee1bd17*](https://towardsdatascience.com/multi-class-text-classification-with-lstm-1590bee1bd17)

[*https://www.embedded-robotics.com/sentiment-analysis-using-lstm/*](https://www.embedded-robotics.com/sentiment-analysis-using-lstm/)

[*https://www.analyticsvidhya.com/blog/2021/06/natural-language-processing-sentiment-analysis-using-lstm/*](https://www.analyticsvidhya.com/blog/2021/06/natural-language-processing-sentiment-analysis-using-lstm/)