CS779 Competition: Sentiment Analysis

Dasari Charithambika
210302
cdasari21@iitk.ac.in
Indian Institute of Technology Kanpur (IIT Kanpur)

Abstract

Write a brief abstract about your models, competition, your rank and scores for various evaluation metrics, etc. Abstract should not be more than 100 words long.

1 Competition Result

Codalab Username: C_210302
Final leaderboard rank on the test set: 7
F1 Score wrt to the best rank: 0.66

2 Problem Description

Given a sentence, automatically predict the sentiment expressed in the sentence, i.e., assign one of the three labels (positive, neutral, and negative) to the sentence.

3 Data Analysis

1. Describe the train dataset that has been provided to you.

   Solution:

   - Train dataset dimension is (92228, 3) which contains columns named as *test_id, sentence* and *gold_label*
   - gold_label contains sentiment label as (-1,0,1) where -1 is negative sentiment, 0 is neutral sentiment and 1 is positive sentiment

2. Analysis of the data, e.g., corpus statistics, noise in the corpus, etc.

   Solution:

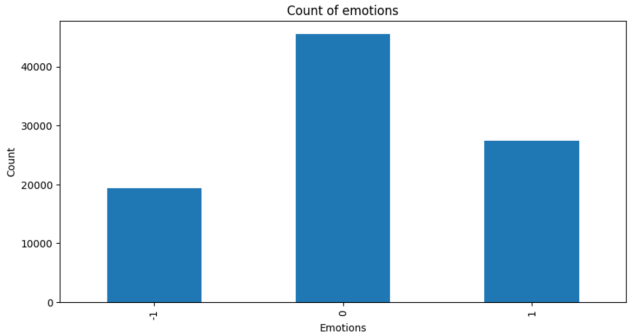

Figure 1: Label Distribution

- Mean of gold_label is 0.086384 and standard deviation of gold_label is 0.706626 which gives us the information that neutral sentiments count are more than both positive & negative

3. Test data will also be provided to you, so you can do analysis of that as well, e.g., how much does it differ from train data?

   Solution:

   - Test data dimension is (5110, 2) which have *text_id* and *sentence*
   - Test data differs from train data in one column which is *gold_label*
   - The model uses train data to train and test data to predict the gold_label(sentiment)

4. Some interesting insights about the data? You can use visualizations if you like. Be creative!

   Solution:

   - The top 10 common words in three gold_label are {the, I, to, and, a, was, of, in, for, is}

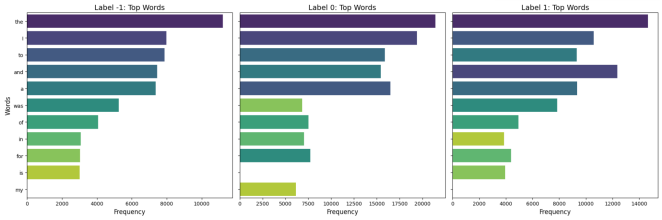

Figure 2: Top 10 words counts from different gold_label

4 Model Description

1. Model evolution: Describe the models in detail that you experimented with in each of the phases, what were the key learnings in each phase that lead to making changes to model architecture or switching to a new model. You can have figures for model architectures.

   Solution:

   - Phase 1: Baseline embedding-based model with a simple LSTM architecture
     - pretrained GloVe embeddings were used
     - Key Insight: Model struggled with capturing long-term dependencies due to limited representational power
   - Phase 2: Upgraded to BiLSTM
     - Added bidirectional LSTM to capture the context in both directions
     - Regularization through dropout
     - Key Learning: Bidirectional architecture significantly improved the accuracy on the validation set

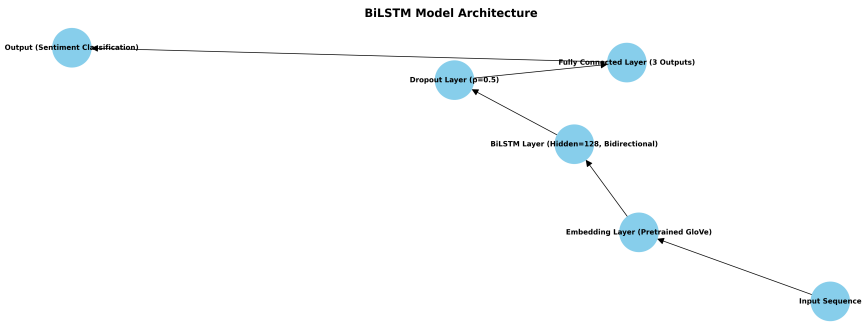

Figure 3: BiLSTM Architecture

2. If you took inspiration from an existing model, please cite the paper(s).

   Solution: The BiLSTM approach likely takes inspiration from "Long Short-Term Memory" by Hochreiter and Schmiddhuber(1997)

3. Detailed description of the final model that worked best on the test set. You can have figures for model architectures.

   Solution:

Layer	Details
Embedding Layer	Pretrained GloVe embeddings (size: 300), Weights frozen
BiLSTM Layer	Hidden size: 128, Bidirectional=True
Dropout Layer	Probability: 0.5
Fully Connected Layer	Outputs class logits for 3-class sentiment classification

Table 1: BiLSTM Model Structure

4. Model objective (loss) functions.

   Solution: Cross Entropy Loss(optimized for multi-class classification)

5. Other relevant details

   Solution:

   - Optimizer: Adam, with a learning rate of 0.001
   - Training Details: Epochs: 10, Batch size: 64
   - Dataset: Train and test datasets loaded as PyTorch datasets, GloVe embeddings mapped tokens to indices

5 Experiments

1. Data Pre-processing you did both for source and target language. What was the reason for doing this kind of pre-processing.

   Solution:

   - Pre-processing steps for Source Language:
     - Tokenization ensures structured input for the embedding layer
     - Lowercasing reduces vocabulary size and model complexity
     - Stopword Removal focuses the model on semantically significant tokens
     - Padding/Truncation handles variable-length sequences and optimizes batch processing
   - Pre-processing step for Target Language:
     - Mapping/Normalization ensures target labels align with the loss function requirements

2. Training procedure: Optimizer, learning rates, epochs, training time, etc for different models you tried

   Solution: For BiLSTM model,

Aspect	Details
Optimizer	Adam optimizer, chosen for its adaptive learning rate capabilities and effective convergence
Learning Rate	Initial learning rate: 0.001; dynamically adjusted during training
Batch size	64 for efficient use of memory per epoch
Epochs	Trained for 10 epochs
Training time	Approximately total time 46 minutes for all 10 epochs on GPU
Loss Function	Cross Entropy loss

   For LSTM model,

Aspect	Details
Optimizer	Adam optimizer, chosen for its adaptive learning rate capabilities and effective convergence
Learning Rate	Initial learning rate: 0.001; dynamically adjusted during training
Batch size	64 for efficient use of memory per epoch
Epochs	Trained for 10 epochs
Training time	Approximately total time 25 minutes for all 10 epochs on GPU
Loss Function	Cross Entropy loss

3. Details about different hyper-parameters for different models. You can use tabular format if you like. How did you arrive at these hyper-parameters?
   Solution:

Hyper-parameter	Value(s) Tried	Final Value	Reason for Selection
Embedding Dimension	100, 200, 300	300	Matches GloVe embedding size, improving semantic representation.
LSTM Hidden Size	64, 128, 256	128	Balanced representation capacity and computational efficiency.
Number of Layers	1, 2	2	Improved model depth for long-term dependency handling.
Dropout Probability	0.3, 0.5, 0.7	0.5	Prevented overfitting without harming validation performance.
Learning Rate	0.001, 0.0005, 0.0001	0.001	Ensured stable convergence during training.
Batch Size	32, 64, 128	64	Optimal trade-off between memory usage and model updates.
Epochs	5, 10, 20	10	Prevented overfitting while ensuring adequate training.

6 Results

1. Results of different models on dev data in three phases in tabular format. If you didn't take part in any phase leave it blank.
   Solution:

Phase	Model	F1 Score
Phase 1	Logistic	0.62
Phase 2	Simple LSTM	0.643
Phase 3	BiLSTM	0.656
Phase 4	BiLSTM with Attention	0.649

2. In the results table show all metrics, as provided in the evaluation scripts. You can also have a column for rank on the leaderboard if you like.
   Solution: F1-Score = Harmonic mean of precision and recall.

$$F_1 \text{ Score} = \frac{2TP}{2TP + FP + FN}$$

3. Results of different models on the test data in tabular format.

Phase	Model	F1 Score
Phase 1	Logistic	0.62
Phase 2	Simple LSTM	0.61
Phase 3	BiLSTM	0.662

4. Briefly explain the results, e.g., what was the best performing model on dev set, test set. Why do you think this model worked better than all other models?

   Solution:

   - Best Performaing Model: BiLSTM model on both dev data and test data
   - Why BiLSTM works better:
     - BiLSTM captures forward and backward dependencies for better context
     - Handles long-term relationships across text spans effectively
     - GloVe embeddings enhance generalization and semantic understanding

7 Error Analysis

1. Error analysis of your different models both on the dev set and the test set. What models worked in general and in what kind of setting? What was the reason for it?

   Solution:

   - Logistic Regression (Phase 1):

     Strengths: Performed relatively well on simpler patterns where contextual understanding was not required

     Weaknesses: Struggled with handling long sequences or complex relationships in text. Misclassified instances where word order and context were critical
   - Simple LSTM (Phase 2):

     Strengths: Improved performance due to sequential modeling. Better at recognizing context over short spans

     Weaknesses: Limited in capturing bidirectional dependencies, leading to errors in tasks requiring comprehensive sequence understanding
   - BiLSTM (Phase 3):

     Strengths: Significantly improved handling of complex dependencies by processing sequences bidirectionally

     Weaknesses: Made errors on ambiguous sequences where external knowledge or domain-specific understanding was required
   - BiLSTM with Attention (Phase 4):

     Strengths: Added interpretability with attention, focusing on key parts of sequences

     Weaknesses: Slightly overfitted to the dev set, possibly due to an imbalance in focusing on certain tokens excessively
2. Error analysis includes analysis of the model, you can use visualizations for this. For e.g., if you used attention mechanism you can use, attention heat maps, etc.

   Solution:

   - Confusion matrix in BiLSTM model
   - The confusion matrix revealed that errors were concentrated in closely related classes, indicating unclear decision boundaries
   - These errors suggest that the model struggles to differentiate between similar classes
3. Analyze why models are not perfect? E.g., what kind of mistakes are made by the best model, how could these be overcome?

   Solution:

   - Ambiguity in Data
   - Out of Vocabulary issues
   - Handling long context

4. Any interesting insights!

   Solution:

   - Logistic Regression: Effective for linear relationships; less suited for tasks requiring contextual understanding
   - BiLSTM: Improved performance on tasks needing nuanced understanding; bidirectional nature reduced errors
   - BiLSTM with Attention: Enhanced interpretability; performance depended on proper attention tuning

8 Conclusion

- Key Findings:

  BiLSTM achieved the highest F1-score (0.662), outperforming Logistic Regression (0.62) and Simple LSTM (0.61)

  Errors were frequent in closely related classes; the attention mechanism showed limited improvement due to suboptimal tuning

  Preprocessing and embeddings played a critical role in model stability

- Recommendations:

  Use hierarchical fine-tuning, data augmentation, and hyperparameter optimization to improve model performance

  Incorporate contextual embeddings like BERT for deeper semantic understanding

- Future Directions:

  Experiment with transfer learning using transformers and ensemble models for better robustness

  Introduce explainability tools (e.g., attention heatmaps) and class-balancing techniques (e.g., focal loss)

- Impact:

  Addressing these points will improve the model's scalability, accuracy, and applicability in real-world tasks

References

1. "Long Short-Term Memory" by Hochreiter and Schmiddhuber(1997)
2. Ashutosh Modi sir's notes