Airline Sentiments: Clustering Twitter Posts about Airline Service based on their Sentiment

By: Shannon Kwok, Javier Chin, Tan Hon Jung and Vanisha Muthu

Introduction

With the rapid digitalisation and the rise of social media, overwhelming volumes of text data are generated daily through tweets, reviews, and online forums. Extracting sentiment from this vast, unstructured text has become a critical task in Natural Language Processing (NLP). In the case of Airline Sentiment Analysis (SA), mining customer feedback on platforms like Twitter can offer real-time insights into customer satisfaction, pain points, and service quality, and guide decisions in customer service, route management, and brand strategy (Chowdhury, 2024). This makes sentiment analysis a valuable tool for getting customer insights and decision-making.

However, the large scale of data production makes manual annotation time-consuming and impractical. Additionally, the informalities and inconsistencies of social media language makes sentiment classification tasks more complex. To address this, recent research has turned to unsupervised learning techniques, deep learning models, and transformer-based architectures (Albladi et al., 2025). These approaches can uncover sentiment patterns without relying on large volumes of labeled data.

Earlier methods such as Naïve Bayes, Support Vector Machines (SVM), and Latent Dirichlet Allocation (LDA) relied heavily on hand-crafted features like n-grams and part-of-speech tags. While effective for basic sentiment classification tasks, these approaches struggled with capturing nuanced semantics and context. Moreover, LDA assumes a bag-of-words model, ignoring word order and hence performing poorly in capturing nuanced semantics (Blei et al., 2003). As such, our project reinforced the model with the use of deep learning models. The introduction of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks marked a significant improvement by enabling the automatic learning of richer textual representations (Xu et al., 2015). Empirical studies have demonstrated the effectiveness of these methods, reporting sentiment classification accuracies of 86.40% on a Movie Review dataset using SVM, 80.70% on the Stanford Sentiment Treebank using deep learning, and 82.52% on Twitter data using a co-trained SVM model—highlighting their applicability across various text sources, including social media.

Moreover, the introduction of transformer models such as Bidirectional Encoder Representations from Transformers (BERT), which leverages self-attention mechanisms, revolutionised NLP by enabling better context understanding. Pre-trained models like BERT and GPT have set new benchmarks in NLP tasks like sentiment analysis, by enabling effective transfer learning with minimal task-specific data. BERT embeddings have significantly advanced NLP tasks as they reflect word meanings based on their usage in specific contexts, while conventional word embeddings such as Word2Vec and GloVe provide static, context-independent word representations (Gardazi et al., 2025).

However, using BERT for sentiment classification often requires fine-tuning with large amounts of labeled data. A study using BERT for sentiment analysis and classification of 1.6 million Twitter posts achieved 92% accuracy (Albladi et al., 2016). However, in cases where the dataset is much smaller, the accuracy of BERT may not be as high. To address this limitation, our project combines unsupervised clustering methods with BERT embeddings for scalable sentiment analysis. In our project, we implemented and compared the following approaches:

1. Latent Dirichlet Allocation (LDA) with Convolutional Neural Network (CNN)
2. Bidirectional Encoder Representations from Transformers (BERT) Embeddings with K-means Clustering
3. DistilBert Classification with Bayesian Optimisation
4. GMM with BERT Embeddings

Dataset

The dataset comprises tweets from users sharing their experiences with various airlines, each labeled as positive, negative, or neutral. While it offers valuable insights for sentiment analysis, it contains noise and inconsistencies in text formatting. To make the data suitable for sentiment analysis, we cleaned and processed the data by applying the following steps:

1. **Expanding Contractions**: Tweets often contain contractions such as “don’t”. These expressions were expanded into their full forms (e.g.,"do not”) using the contractions package to ensure consistency in the text.
2. **Removing Hashtags**: Hashtags, while commonly used for categorisation or emphasis, do not contribute meaningful information in sentiment analysis.

Therefore, hashtags were removed but the associated word was kept.

3. **Extracting and Removing Airline Mentions**: A regular expression was used to detect mentions of specific airlines (e.g., @united) and remove them. This separation ensures the model learns from the tweet's content and tone rather than relying on airline names, which may bias sentiment predictions. It enhances generalisation, allowing better performance on unseen data, including tweets about airlines not in the training set.
4. **Identifying and Removing URLs and User Mentions**: URLs and other user mentions present in the tweets were removed because they are not relevant for sentiment analysis.
5. **Converting Emojis to Text**: To capture the sentiment conveyed by emojis, we converted emojis into text using the emoji package.
6. **Residual Noise**: Tweets often contain slang and informal grammar. While preprocessing addressed some of this noise, residual inconsistencies were retained for the model to learn from, as deep learning models are generally robust to such variation.
7. **Visualizations and Analysis**: After the cleaning process, several visualizations were generated to better understand the distribution of sentiments across the dataset.

Firstly, we looked at the distribution of airline sentiment and evidently with reference to Fig 1, a large proportion of the data suggests negative sentiments. This can lead to models that are biased towards the majority class, resulting in poor performance and inaccurate predictions for the minority class.

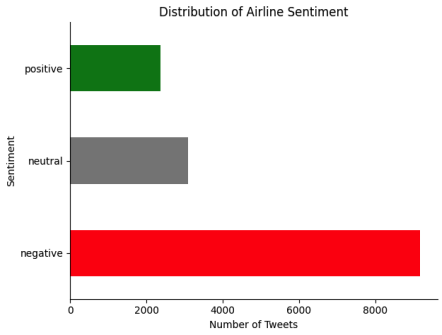

Fig 1: Distribution of airline sentiment

Thereafter, we visualized the distribution of sentiment by airline. With reference to Fig 2, it is clear that airlines such as AmericanAir, United and USAirways have a disproportionate amount of negative sentiments. This phenomenon further validates our approach to remove airline mentions in order to ensure the model learns from content and tone of the tweet as opposed to using the airline name as a predictive feature.

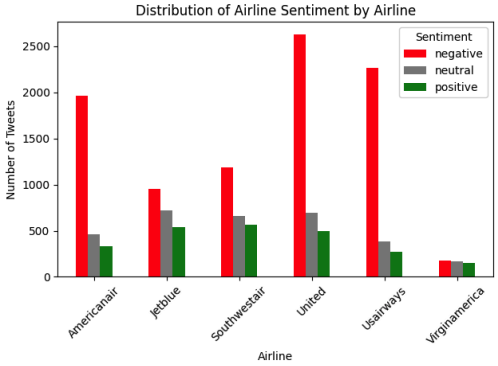

Fig 2: Distribution of Airline Sentiment by Airline

Fig 3: Positive Word Cloud

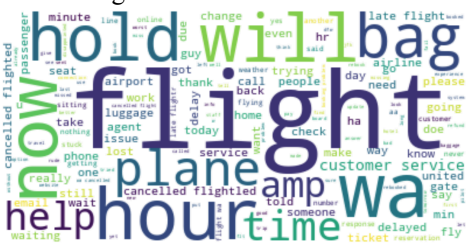

Fig 4: Negative Word Cloud

As seen from the word clouds, certain words such as “thank”, “flight” and “will” appear in high frequencies across multiple classes, making it a potential source for misclassification errors in our ML models.

Methods

The task is to perform sentiment analysis of tweets, where the goal is to classify tweets into three categories: positive, neutral, or negative. This is a multi-class classification task, mapping each tweet to a sentiment label $y \in \{0, 1, 2\}$, where 0 is negative, 1 is neutral, and 2 is positive.

Our first approach is to use Latent Dirichlet Allocation (LDA) to capture topic-level representations of the tweets before we use deep learning models. LDA assumes each tweet is a mixture of topics and identifies the underlying topics based on word distributions. LDA is mathematically modeled as:

$$P(w|z) = \frac{exp(\theta_z^T w)}{\Sigma_z exp(\theta_z^T w)}$$

where w is a word, z is a topic, and θ represents the topic-word distribution. Each tweet is modeled as a distribution over topics z, and the goal is to find a topic distribution θ for each tweet that best describes the content.The optimal number of topics was determined through coherence scoring, with 9 topics found to provide the best balance between model simplicity and feature richness.

We initially used a Long Short-Term Memory (LSTM) network to classify sentiment based on the LDA-extracted features. LSTMs are effective for sequence data, but this model showed a 75% accuracy with high validation loss of 1.2036, indicating overfitting. Overfitting suggests the model was learning noise instead of underlying patterns.

To mitigate overfitting, we switched to a Convolutional Neural Network (CNN), which is better at extracting local patterns from structured inputs like LDA features. This change improved the accuracy to 78% and reduced validation loss to 0.5648, indicating better generalization and improved model performance compared to the LSTM-based approach. To further combat overfitting, early stopping was implemented during training. Early stopping halts the training process once the validation loss stops improving, ensuring the model doesn't overfit to the training data. We set early stopping to trigger after 3 epochs without improvement, which helped stabilize the model and improve its generalization. We also explored a margin-based approach, which aims to increase the decision margin between sentiment classes during training. However, this approach did not lead to performance improvements and resulted in a marginal decrease in accuracy, suggesting it wasn't effective for this task.

In our second approach we explored Bidirectional Encoder Representations from Transformers (BERT) Embeddings and K-Means Clustering. Clustering is a fundamental unsupervised learning technique used to group similar data points together. This model aimed to classify the text data based on sentiment by leveraging pre-trained language models and clustering algorithms. To obtain vectorised representations of text, we used DistilBERT Base Uncased Fine-tuned SST-2, a variant of BERT pre-trained for sentiment analysis. The model converts text into 768-dimensional embeddings while preserving semantic meaning. Using Uniform Manifold Approximation and Projection (UMAP), we reduced the embeddings' dimensionality from 768 to 15 dimensions, which helped retain significant information while minimising noise and improving clustering efficiency.

To determine the optimal number of clusters (K) for K-Means clustering, we employed the Elbow Method, where we plotted the Sum of Squared Errors (SSE) against values of K. The point of inflection in the graph indicated that the optimal number of clusters was K = 6. Additional validation was performed using trial and error to confirm the choice. We applied K-Means clustering with K = 6, using K-Means++ initialisation to optimise the placement of initial centroids and prevent convergence to suboptimal solutions. The dataset was split into an 80-20 train-test ratio, where the model was trained and tested on separate subsets of data. After clustering, we computed the average sentiment score for each cluster using sentiment analysis. Clusters were then classified as "positive," "negative," or "neutral" based on their mean sentiment score.

The next method we used was transfer learning using a pre-trained model. We used the DistilBertForSequenceClassification model from the Transformers package with fine tuning. DistilBert is a fast, cheap and light transformer model based on the Bert architecture (Sanh et al., 2019), making it suitable as an NLP model for sentiment classification. To use such a model, we first have to tokenize the data using DistilBertTokenizer which converts raw text in the form of Tweets into a format that the DistilBert model can understand. We then used AdamW instead of Adam as the optimizer for this model as it decouples weight decay from the gradient updates, leading to better generalization and stability, particularly in large-scale models (Zhou et al., 2024). Before hyperparameter tuning, we were only able to achieve an F1 score of 81-82%. Using the Bayesian Optimization method, powered by the Optuna package, for maximising F1 score, we eventually are able to obtain the highest F1 score of 83.59%, a marginal improvement to our current model. The best hyperparameters chosen were learning rate = 3.1646e-05, batch size = 21 and epochs = 5. We used Bayesian Optimization as a hyperparameter tuning method as it uses an acquisition function to decide which hyperparameter settings to evaluate next, thus making it more efficient than Grid Search or Random Search.

In our last approach, we employed a transfer learning approach using the stsb-roberta-large variant of the SentenceTransformer model to convert tweets into dense, semantically rich embeddings. These embeddings were then reduced to two dimensions using Principal Component Analysis (PCA) to simplify visualization and processing. We trained a Gaussian Mixture Model (GMM) on the reduced training embeddings, allowing it to form probabilistic clusters based on sentiment-related patterns. To label these clusters, we used majority voting based on the true sentiment labels within each cluster. The GMM then predicted the sentiment clusters for test tweets, which

we mapped to sentiment labels using the learned cluster-label associations.

To optimize the pipeline, we experimented with different embedding models such as all-MiniLM-L6-v2 and RoBERTa. Initially, MiniLM outperformed RoBERTa (0.66 vs. 0.64 accuracy), possibly due to RoBERTa's increased complexity leading to overfitting. However, after refining the preprocessing step—specifically by removing stopwords and removing very short words—RoBERTa's performance improved to 0.69 accuracy, surpassing MiniLM. To visualize clustering performance, we plotted the test data in 2D, colored by predicted sentiment, revealing how the GMM differentiated between sentiment clusters.

Method	Accuracy	Precision	F1-score	Recall
LDA with CNN	78.86%	78%	78%	79%
BERT with K-Means Clustering	70.90%	67.10%	67.60%	70.80%
GMM with BERT Embeddings	69.82%	57.67%	62.13%	69.42%
DistilBert with Bayesian Optimization	82.38%	82.54%	82.33%	82.38%

Results & Discussions

We used F1 scores, precision and accuracy scores to measure the model's performance and compare them across the board.

For LDA, the model performed better after switching to CNN from LSTM as it can extract n-grams features more efficiently through convolutional filters. Moreover, CNN's focus on local features further improved the model's performance. In addition, the use of the early stopping technique helped to moderate the learning of the model to reduce overfitting and prevent the model from memorising train data. Further tweaks in batch size, dropout and epochs yielded a F1 score of 0.78.

For K-Means Clustering using DistilBert embeddings, the model had a low-moderate F1 score of 67.6%. The positive and negative clusters had much higher F1 scores (68% and 83%) than the neutral cluster (22%). Thus, the model is lacking in neutral classification, but can classify positive and negative tweets well. We found that many of the misclassified tweets had a lower sentiment confidence in the dataset, hence they are not as strongly positive or negative in sentiment. This could explain the misclassification rate. To finetune the model while generating the BERT embeddings, the batch size and number of dimensions were adjusted to find a good balance between accuracy and runtime.

For DistilBert Classification with Bayesian Optimisation, the model had the best weighted F1 score of 82-83%, indicating a good balance between precision and recall. This is likely due to its ability to learn complex language representations. While this model gives the best results out of the 4 presented in this report, it comes with its own set of challenges, such as it being computationally intensive. Thus, any methods to address the imbalance in the dataset to make its predictions more accurate such as using Synthetic Minority Over-Sampling Technique (SMOTE) or Stratified K Cross Validation would cause further computational strain and result in longer runtimes.

For GMM using BERT Embeddings, the model had a low-moderate F1 score ranging between 64%-66% when using the different Sentence Transformer initialisations, specifically 'all-MiniLM-L6-v2' and ‘'stsb-roberta-large'. The scores improved to range 66%-69% when stop words were removed in further data cleaning. The neutral sentiment cluster consistently showed very low F1-scores, often near zero, indicating the model's difficulty in accurately classifying neutral statements—likely due to their ambiguity and overlap with other sentiments. In contrast, negative sentiment consistently outperformed positive (e.g., 0.81 vs. 0.44 or 0.84 vs. 0.57), suggesting the model found negative expressions easier to detect, possibly due to their more extreme language. This performance gap may also be influenced by the overrepresentation of negative sentiment in the dataset, giving the model more training examples to learn from. These findings highlight the importance of addressing class imbalance and improving the model's ability to handle subtle or ambiguous sentiments.

The F1 score of the models ranges from 69% to 83%, which is relatively lower than human performance in text classification tasks, where accuracy typically ranges from 80% to 90%. However, a model doesn't always need to outperform humans to be valuable. Its primary benefit lies in supplementing human efforts by automating repetitive tasks, particularly in high-volume areas such as sentiment analysis of social media or customer feedback. This allows humans to shift their focus to more meaningful or complex tasks that require creativity, strategic thinking, or emotional intelligence—areas where machines still face limitations. Furthermore, this project raises privacy concerns in the data being used. Sensitive information could be inadvertently exposed without proper anonymization. It is crucial to protect personally identifiable information (PII), obtain users' consent and follow ethical data usage protocols to prevent misuse.

References

1. Albladi, A., Islam, M., & Seals, C. (2025). Sentiment analysis of Twitter data using NLP models: A comprehensive review. *IEEE Access*, *13*, 30444–30468. https://doi.org/10.1109/access.2025.3541494
2. Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. Journal of Machine Learning Research 3 , 3. https://doi.org/https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf
3. Chowdhury, R. H. (2024). Sentiment analysis and social media analytics in Brand Management: Techniques, trends, and implications. *World Journal of Advanced Research and Reviews*, *23*(2), 287–296. https://doi.org/10.30574/wjarr.2024.23.2.2369
4. Gardazi, N. M., Daud, A., Malik, M. K., Bukhari, A., Alsahfi, T., & Alshemaimri, B. (2025). BERT applications in natural language processing: A review. *Artificial Intelligence Review*, *58*(6), 166. https://doi.org/10.1007/s10462-025-11162-5
5. Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019, October 2). *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. arXiv.org. https://arxiv.org/abs/1910.01108v4
6. Xu, J., Wang, P., Tian, G., Xu, B., Zhao, J., Wang, F., & Hao, H. (2015). Short text clustering via Convolutional Neural Networks. Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing. https://doi.org/10.3115/v1/w15-1509
7. Zhou, P., Xie, X., Lin, Z., & Yan, S. (2024). Towards understanding convergence and generalization of AdamW. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1-8. https://ink.library.smu.edu.sg/sis_research/8986