# Twitter Sentiment Analysis Using Classical & Neural Models

Name: HUANG YEN WEI

## 1. Problem Statement

Sentiment analysis on social media data is a crucial task for understanding public opinion and user behavior. This project aims to classify tweets from the Kaggle "Twitter Sentiment Dataset" into negative (-1), neutral (0), and positive (+1) sentiments. The challenge lies in handling large-scale textual data efficiently while comparing classical machine learning approaches with a neural network model (CNN–LSTM).

## 2. Research Questions

1. Which model achieves the best overall and per-class performance for predicting sentiment?
2. How do TF-IDF and Word2Vec features compare when used with classical models?
3. Does a CNN–LSTM hybrid outperform classical ensembles (Voting Classifier) on this dataset?

## 3. Dataset

- **Source:** Kaggle — Twitter Sentiment Dataset (Saurabh Shahane, 2021)
- **Columns:** clean_text (string), category (int: -1, 0, 1)
- **Size:** 162,980 tweets

## 4. Methods

1. **Exploratory Data Analysis (EDA) & Preprocessing**
    a. Class distribution, text length analysis, missing data handling
    b. Tokenization and stopword removal
    c. Sampling: due to memory constraints, we experimented with different sample sizes (1,000; 15,000; 20,000) to train models effectively
2. **Feature Engineering**
    a. **TF-IDF Features:** Classical baseline models
    b. **Word2Vec Features:** Average tweet embeddings for classical models
    c. **Neural Network Features:** Keras tokenizer with padding, feeding into CNN–LSTM
3. **Models**
    a. Classical: Decision Tree, KNN, Logistic Regression
    b. Ensemble: Voting Classifier combining best-performing classical models
    c. Neural: CNN–LSTM hybrid (embedding → convolution → LSTM → dense layers)
    d. **Best performing model:** CNN–LSTM
4. **Evaluation**
    a. Metrics: accuracy, precision, recall, macro F1, per-class F1
    b. Confusion matrices and ROC-AUC

## 5. Work Organization

- **Joaquin:** EDA & preprocessing, main pipeline, CNN–LSTM implementation
- **Emin:** TF-IDF feature engineering

- **Will:** Word2Vec feature engineering

**Workflow:** At the start, the team defined a clear structure and split tasks. Each member worked separately due to different schedules and other project commitments. Weekly meetings during the Data Science lecture were used to check progress, coordinate next steps, and resolve issues.

## 6. Reflexion

Working on the Word2Vec pipeline was one of the most technically challenging and insightful parts of this project for me. Compared with TF-IDF, which tends to behave more predictably, Word2Vec embeddings required a deeper understanding of how semantic representations are formed and how classical algorithms interpret dense vectors.

One of the first challenges I encountered was the mismatch between pre-trained embeddings and the unique style of Twitter language—informal expressions, sarcasm, abbreviations, emojis, and inconsistent grammar. Many words in the dataset did not exist in the pre-trained vocabulary, which weakened the quality of the averaged embeddings and reduced model performance. This issue helped me realize why domain-specific embeddings are often necessary for social-media tasks.

Another difficulty came from the nature of averaged embeddings themselves. Although averaging word vectors produces a compact representation, it removes important information about word order and context. Classical machine-learning models such as KNN and Decision Trees depend heavily on the structure and separability of feature space, and averaged embeddings do not provide enough signal for them to learn effectively. Logistic Regression was the only model that showed moderate stability ($F1 \approx 0.535$), confirming that linear models can still extract weak global patterns from dense vectors.

Additionally, due to memory constraints, we trained the Word2Vec models on multiple sample sizes. The performance fluctuated significantly, which taught me how sensitive dense embeddings are to dataset scale. This part of the project not only improved my understanding of representation learning but also strengthened my appreciation for deep-learning models. When I saw how much better the CNN–LSTM handled sequential patterns and semantic nuance, it became clear that modern NLP relies heavily on context-aware architectures to capture the expressive nature of human language.

Overall, this experience helped me grow technically and gave me a clearer picture of the gap between classical models and neural approaches. It also inspired me to explore more advanced embedding techniques in future projects.

Working on Word2Vec revealed both the power and limitations of dense embeddings. While TF-IDF performed more consistently, Word2Vec required more careful tuning. Classical models struggled with averaged embeddings because these vectors lose important context. Logistic Regression performed the best among Word2Vec models ($F1 \approx 0.535$), while KNN and Decision Trees performed significantly worse.

The main challenges I encountered were:

• Pre-trained embeddings mismatching informal Twitter language

• High-dimensional vectors causing instability

• Variability across different sampled dataset sizes

Despite these obstacles, implementing Word2Vec gave me insight into how crucial representation learning is. It also illustrated why models like CNN–LSTM, which capture sequential and contextual information, perform much better on short and noisy text such as tweets.

## 7. Conclusion and Future Work

The CNN–LSTM achieved the highest performance (F1 = 0.8459), outperforming all TF-IDF and Word2Vec models. Among classical approaches, TF-IDF + Logistic Regression was the strongest, while Word2Vec models lagged. Overall, deep learning proved more capable of capturing the contextual nuances needed for sentiment analysis.

Future improvements may include:

- Training on the full dataset using more computational resources
- Experimenting with transformer-based models such as BERT
- Building Twitter-specific embeddings
- Conducting systematic hyperparameter tuning
- Exploring multilingual sentiment classification