EMOTION-DRIVEN MUSIC GENERATION FROM VIETNAMESE TEXT

GROUP 10

Member of Group

2025

Mac Pham Thien Long
22070503

Nguyen Duy Thuc
21070278

Nguyen Dinh Duy
22070367

Nguyen Thi Ngoc Anh
22070457

Vu Hong Thuy
22070360

Table of Content

2025

INTRODUCTION

PROBLEM STATEMENT

Music is becoming a powerful tool for **personalized experiences** in entertainment, content creation, and mental health support. However, generating **emotionally** aligned music from **Vietnamese text** remains a technical challenge due to the language's rich context and expressiveness.

The core issues lie in how to
(1) understand and accurately extract the **diverse emotional features** from **Vietnamese text**
(2) generate **novel, natural music** that truly reflects the desired emotions.

This project proposes to develop and evaluate an end-to-end system that generates emotion-driven music from Vietnamese text based on deep learning approaches.

LITERATURE REVIEW

1. Emotion Recognition methods

Method	Strengths	Limitations
Lexicon-based	Easy to implement, no training required	Poor at understanding context
Classical ML	Controllable, works well on small datasets	Limited semantic representation
Deep Learning	High accuracy, strong contextual understanding	Needs large labeled datasets & computing power

2. Music Generation methods

Method	Strengths	Limitations
LSTM	Simple, stable for sequential data	Weak at capturing long-term dependencies
Transformer	Excellent for long-range relationships	High resource demand, complex to optimize

Research Gaps:

- *No research on an end-to-end system for emotion-based music generation from Vietnamese text*

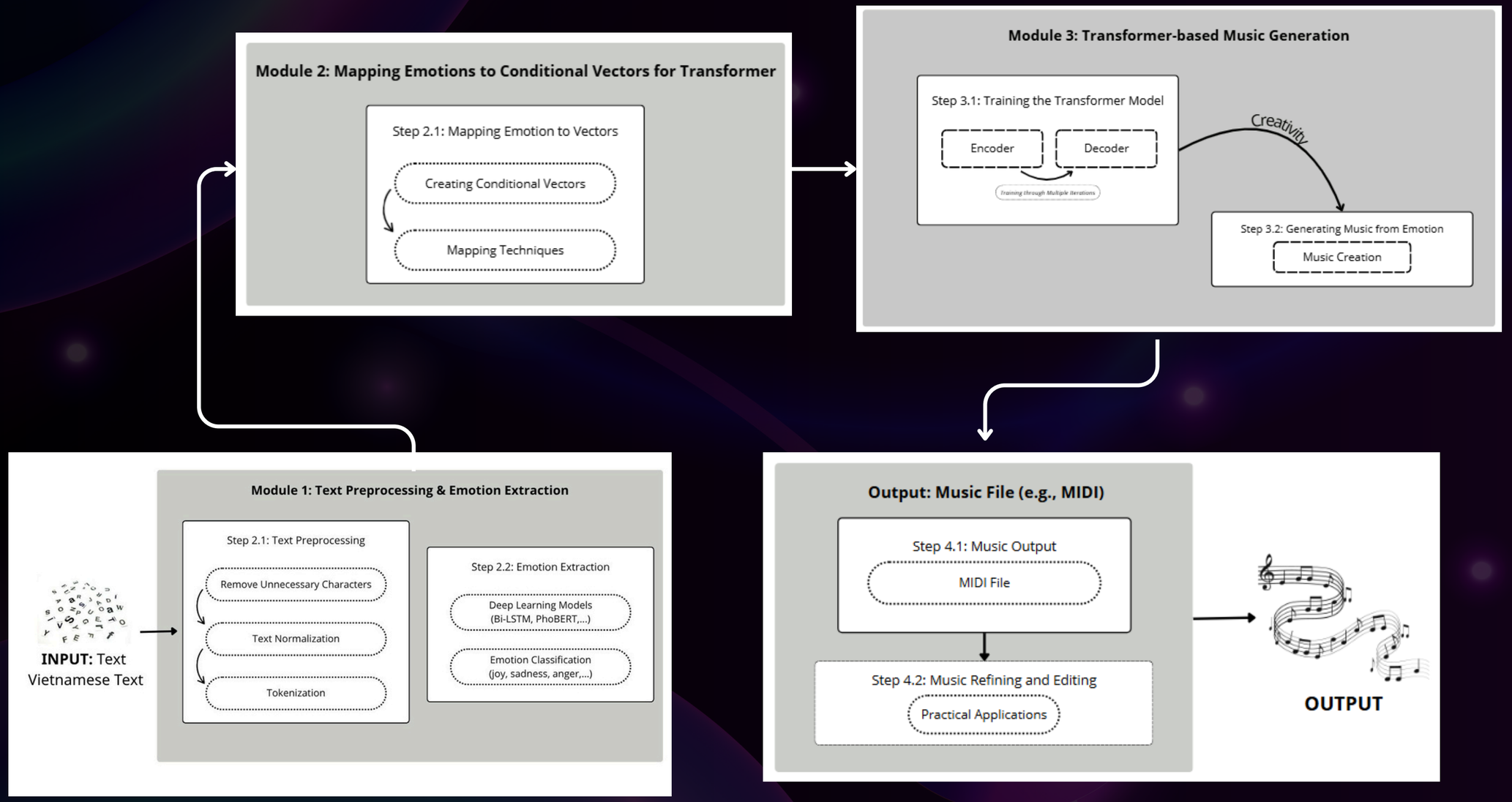
PROJECT GOALS

- Build a complete framework that transforms Vietnamese input text into emotional music output.
- Collect and construct a Vietnamese dataset for the emotion recognition task with 6 emotion labels.
- Develop a Vietnamese text-based emotion recognition model with at least 90% accuracy on the test dataset.
- Develop a Transformer-based model capable of generating emotion-conditioned music.

METHODOLOGY

2025

SYSTEM ARCHITECTURE - EMOTION-BASED MUSIC GENERATION



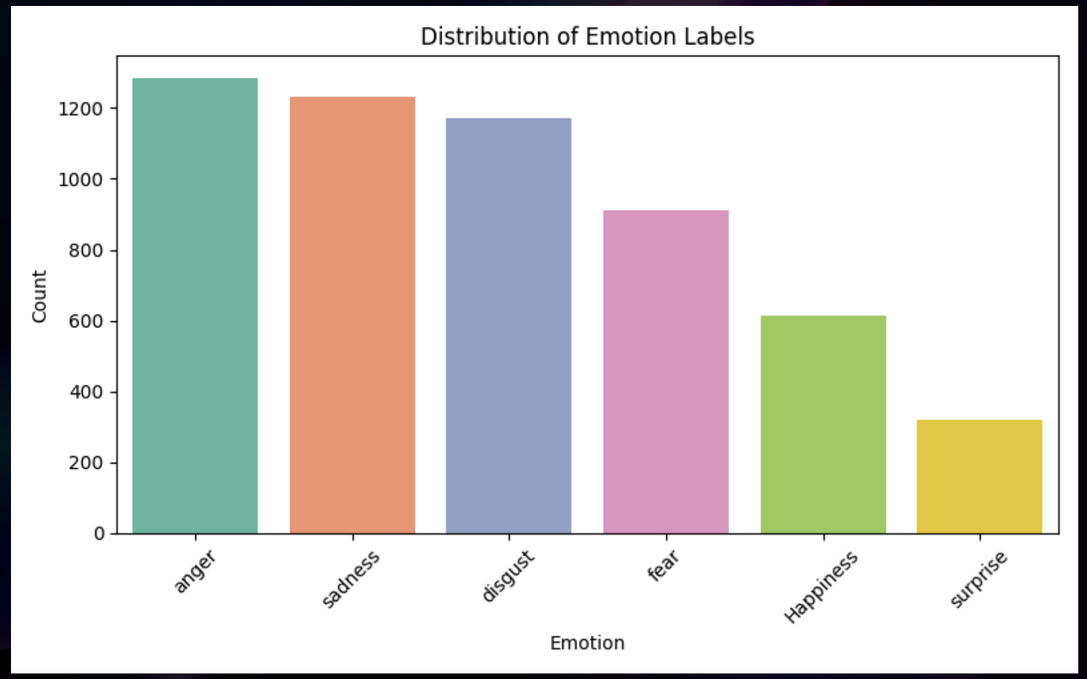
DATA COLLECTION

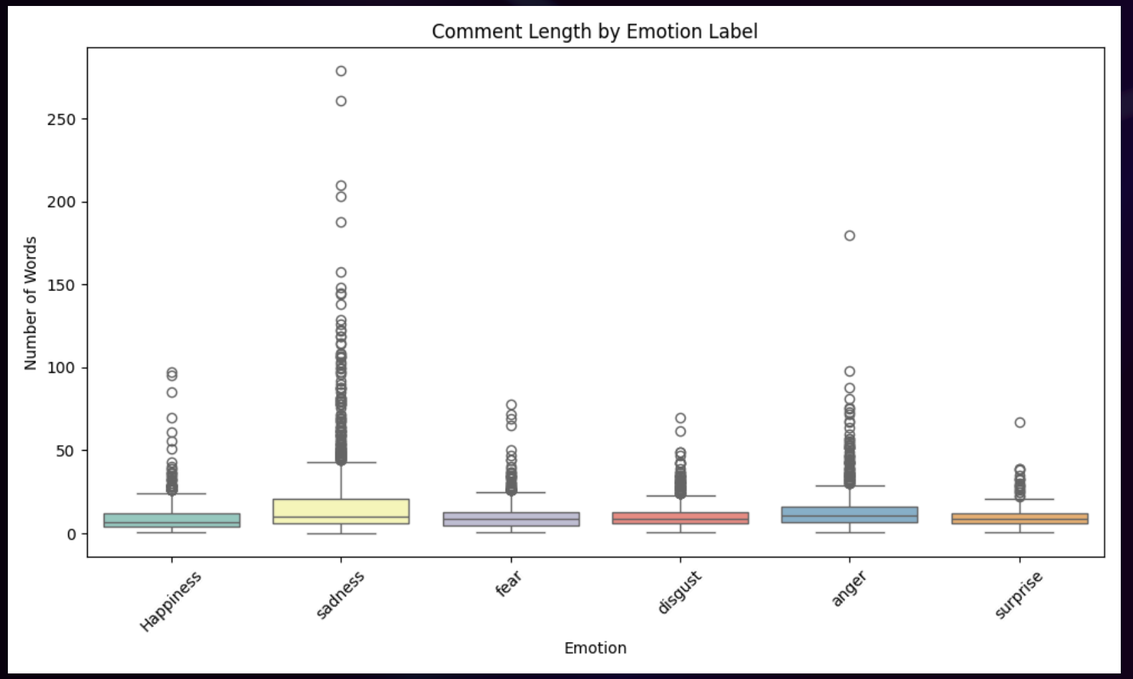
1. Data Source
   - Platform: YouTube
   - Content type: Vietnamese user comments from emotion-rich videos
   - Language: Vietnamese

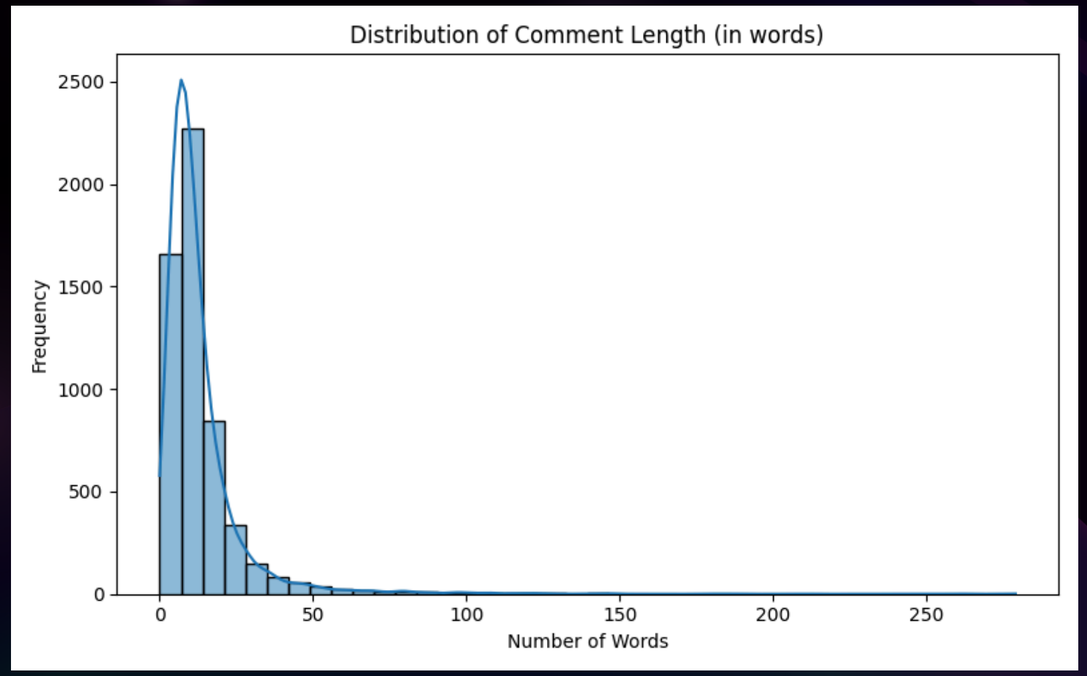
2. Tools & Collection Process
   - Language: Python
   - Libraries used:
   - ***selenium***: browser automation to scroll & extract comments
   - ***webdriver-manager***: manage compatible ChromeDriver
   - ***openpyxl***: export data to .xlsx

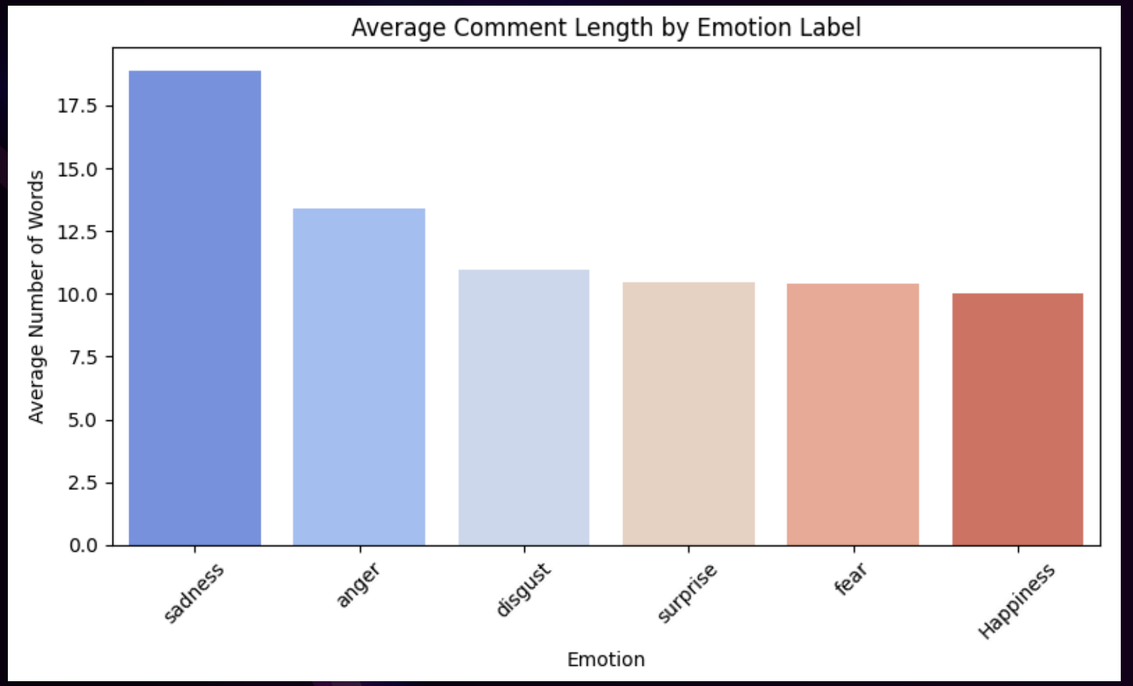
*__Steps:__
   - Open video in Chrome using Selenium
   - Scroll to load all comments
   - Extract Vietnamese-language comments
   - Save to Excel for manual labeling

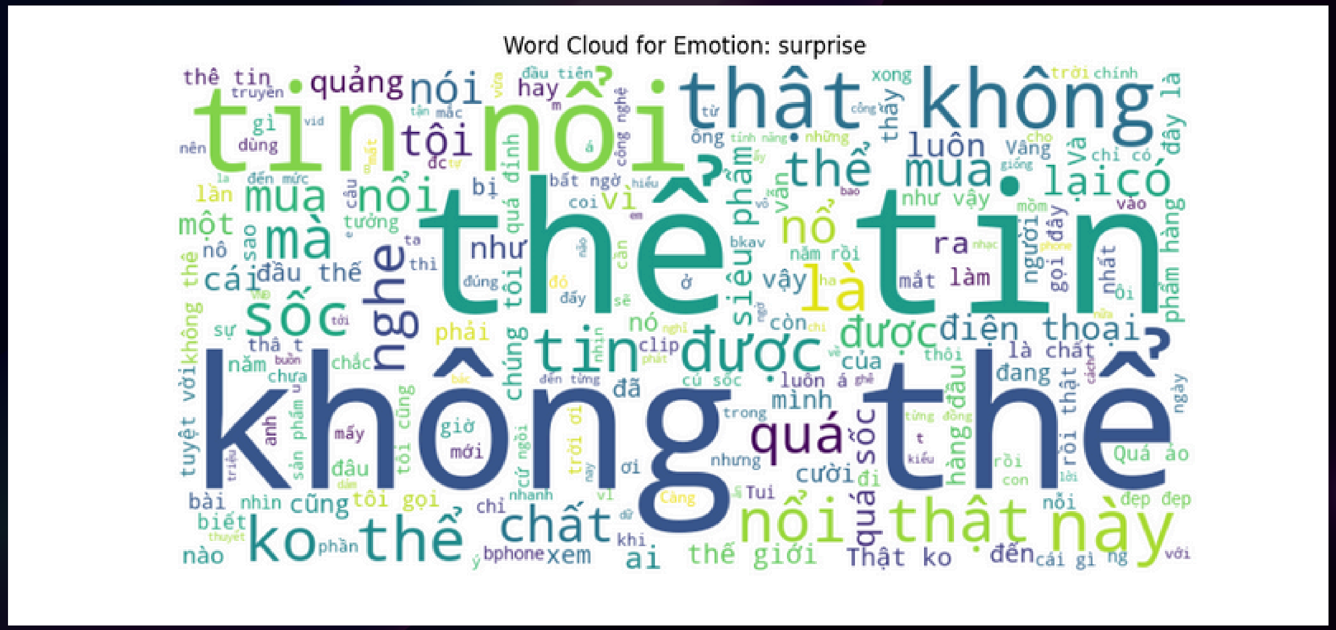
3. Emotion Labels (Ekman-6)
   - 36.000 data, 2 columns: comments & emotion labels
   - Happiness, sadness, anger, fear, surprise, disgust


Distribution of Emotion Labels
Count
Emotion
anger
sadness
disgust
fear
Happiness
surprise

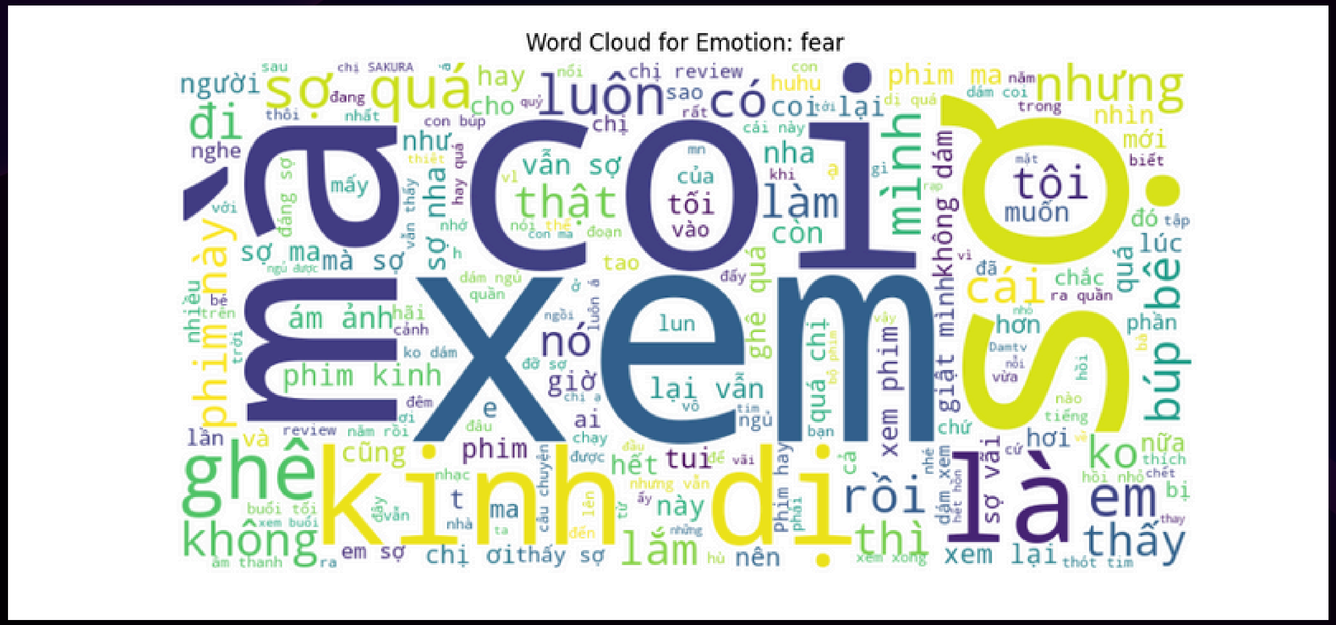

Comment Length by Emotion Label
Number of Words
Emotion
Happiness
sadness
fear
disgust
anger
surprise

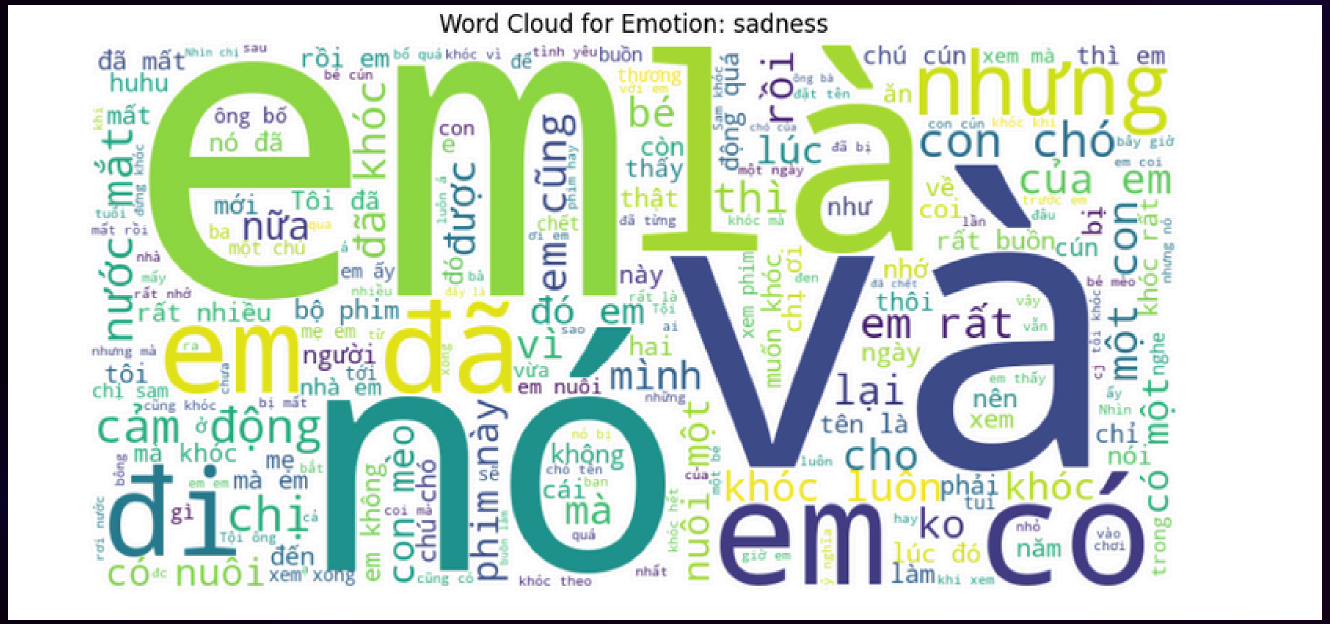

Distribution of Comment Length (in words)
Frequency
Number of Words

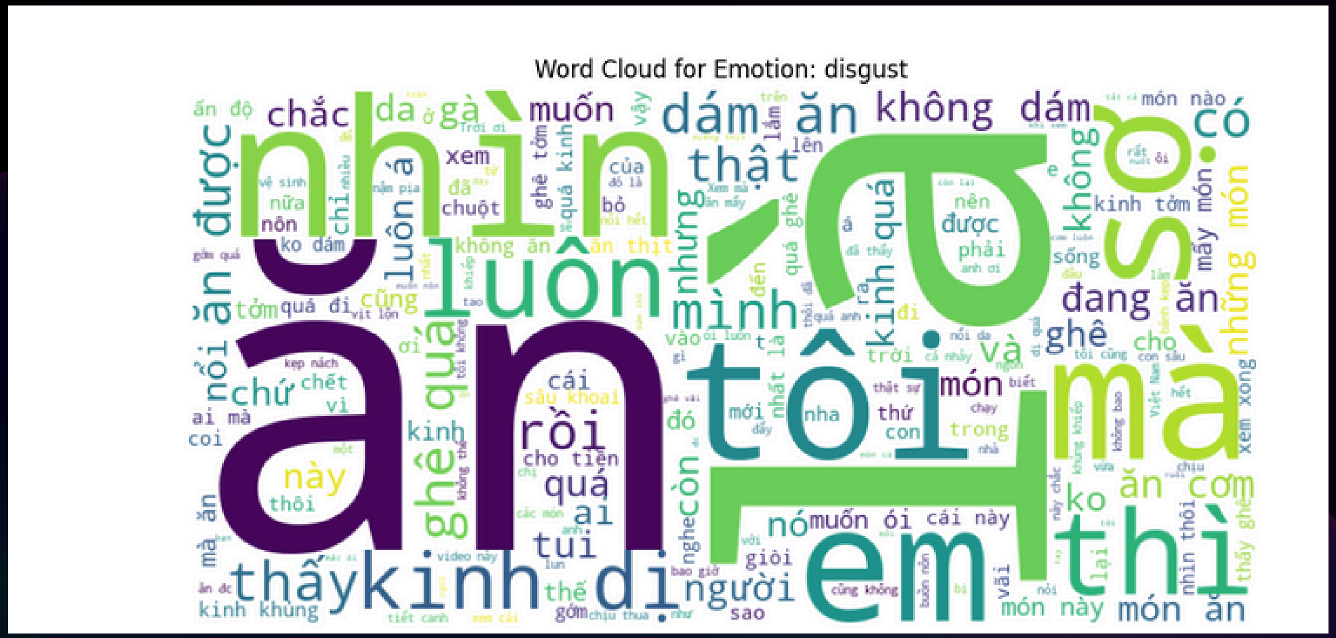

Average Comment Length by Emotion Label
Average Number of Words
Emotion
sadness
anger
disgust
surprise
fear
Happiness

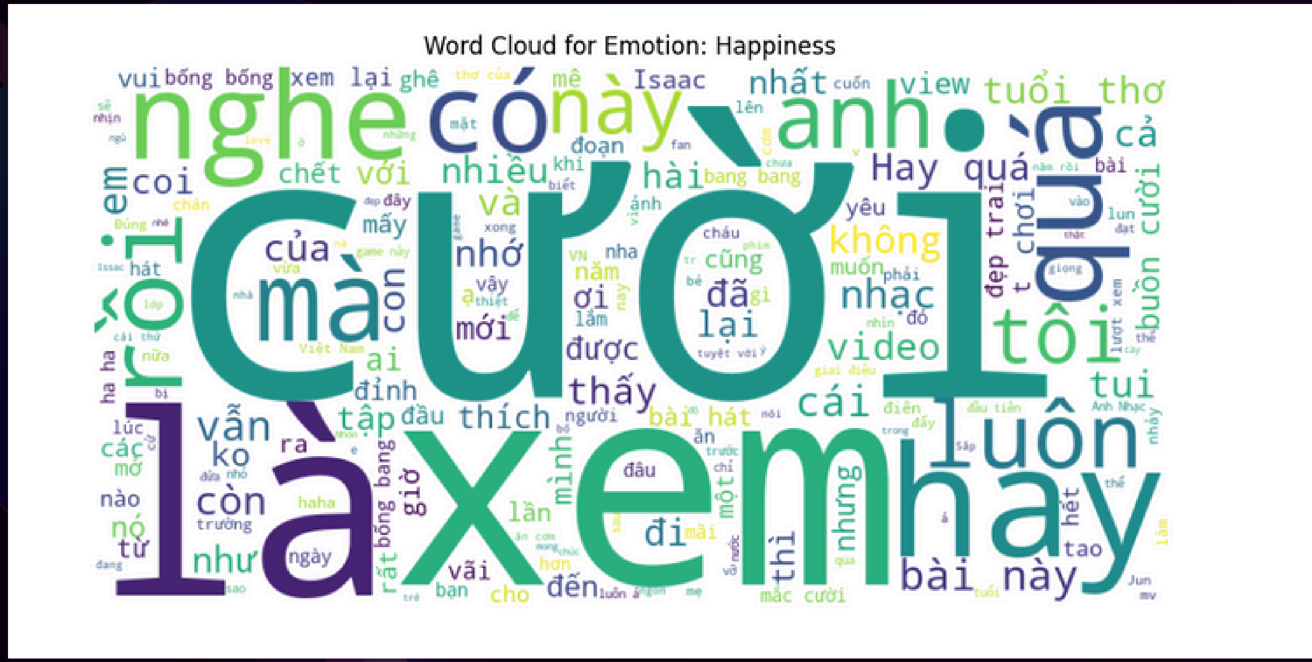

Word Cloud for Emotion: surprise
thể tin
không thể
tin nổi
thật
không
được
nổi
quá


Word Cloud for Emotion: anger
ức chế
mà
là
không
cái
thì
cho
quá
người


Word Cloud for Emotion: fear
coi
mà
xem
sợ
kinh
đi
là
tôi
ghê


Word Cloud for Emotion: sadness
em
là
và
nó
đã
em
có
nhưng


Word Cloud for Emotion: disgust
ăn
ghê
nhìn
sợ
tôi
mà
thì
em
luôn
kinh


Word Cloud for Emotion: Happiness
cười
xem
là
hay
nghe
có
ảnh
quá
luôn
tôi

MUSIC DATASET

- **MAESTRO Dataset (for Pre-training):**
  - A large, unlabelled dataset containing over 200 hours of virtuosic classical piano performances. It was used to teach the model the fundamental "grammar" of music, such as melody, harmony, and rhythm, providing a robust musical foundation.

- **VGMIDI Dataset (for Fine-tuning):**
  - A smaller, curated dataset of 200 MIDI pieces from video game soundtracks. This dataset is emotionally labelled, with each piece assigned specific Valence and Arousal scores, which was used to fine-tune the model to generate music corresponding to specific emotional states.

PREPROCESSING

- Stopword list created to remove semantically irrelevant Vietnamese words.
- Teencode dictionary built to normalize slang (e.g., "k" → "không", "iu" → "yêu").
- Regex patterns used for efficient, context-aware slang replacement.

```python
# --- VIETNAMESE STOPWORDS ---
VIETNAMESE_STOP_WORDS = set([
    "và", "hoặc", "là", "có", "của", "trong", "theo", "này", "đây", "với", "cho", "mà", "được",
    "cùng", "bởi", "từ", "nếu", "cũng", "sẽ", "khi", "không", "để", "đi", "vì", "mới", "cả",
    "hơn", "nhiều", "ít", "thì", "như", "các", "vào", "bằng", "ra", "lên", "xuống", "qua", "lại",
    "anh", "em", "chị", "bạn", "tôi", "mình", "nó", "họ", "chúng ta", "chúng tôi", "chúng nó",
    "ai", "gì", "đâu", "nào", "sao", "bao nhiêu", "lúc nào", "tại sao", "ở", "tại", "trên",
    "dưới", "trước", "sau", "ấy", "những", "một", "hai", "ba", "vài", "rằng", "ạ", "à", "ừ",
    "dạ", "vâng", "ơi", "nhỉ", "nhé", "nha", "đó", "đây", "kia", "ấy"
])

# --- TEENCODE DICTIONARY ---
TEENCODE_MAP = {
    "k": "không", "ko": "không", "khum": "không", "hok": "không", "hem": "không", "hong": "không",
    "j": "gì", "g": "gì", "z": "gì", "zậy": "vậy", "zay": "vậy", "v": "vậy", "zô": "vào", "zo": "vào",
    "r": "rồi", "roi": "rồi", "wá": "quá", "wa": "quá", "iu": "yêu", "luv": "yêu",
    "thks": "cảm ơn", "tks": "cảm ơn", "thanks": "cảm ơn", "ty": "cảm ơn",
    "ok": "được", "oke": "được", "oki": "được", "okie": "được", "dc": "được", "đc": "được",
    "vl": "rất", "vkl": "rất", "vcl": "rất", "vch": "rất", "vs": "với", "mn": "mọi người",
    "bik": "biết", "bjt": "biết", "bit": "biết", "bb": "tạm biệt", "bye": "tạm biệt",
    "h": "giờ", "hjo": "giờ", "ng": "người", "nguoi": "người", "ntn": "như thế nào",
    "a": "anh", "e": "em", "ib": "nhắn tin", "inbox": "nhắn tin", "s": "sao",
    "đk": "được không", "dk": "được không", "t": "tôi", "b": "bạn", "m": "mày",
}

# Compile regex for efficiency
teencode_pattern = re.compile(r'\b(' + '|'.join(re.escape(key) for key in TEENCODE_MAP.keys()) + r')\b', re.IGNORECASE)

print("Stopwords and teencode dictionaries are loaded.")
```

PREPROCESSING

- Custom function replaces slang with standard Vietnamese using regex and predefined mapping.
- Enhances text consistency and model performance by handling informal language.
- Utility function detects presence of Vietnamese diacritics via regex.
- Texts without diacritics (often noisy/slang) are removed to preserve data quality.

```python
def normalize_teencode(text: str) -> str:
    """Standardizes teencode words to their proper form."""
    return teencode_pattern.sub(lambda m: TEENCODE_MAP[m.group(0).lower()], text)

def has_vietnamese_diacritics(text: str) -> bool:
    return re.search(r'[áàảãạăắằẳẵặâấầẩẫậđéèẻẽẹêếềểễệíìỉĩịóòỏõọôốồổỗộơớờởỡợúùủũụưứừửữựýỳỷỹỵ]', text, re.IGNORECASE) is not None
```

PREPROCESSING

- Lowercasing, HTML/URL removal, Unicode normalization, punctuation & whitespace trimming.
- Normalize slang, tokenize Vietnamese text, and reconstruct clean phrases.
- Remove nulls, duplicates, and texts without diacritics to ensure clean, analyzable input.

```python
def preprocess_pipeline(text: str) -> str:
    if not isinstance(text, str):
        return ""

    # Convert to lowercase
    s = text.lower()
    # Normalize teencode
    s = normalize_teencode(s)
    # Remove HTML tags and URLs
    s = re.sub(r'<[^>]+>', ' ', s)
    s = re.sub(r'http\S+|www.\S+', ' ', s)
    # Normalize Unicode
    s = unicodedata.normalize("NFC", s)
    # Remove special characters, keeping letters, numbers, whitespace, and standard punctuation
    vietnamese_chars = 'áàảãạăắằẳẵặâấầẩẫậđéèẻẽẹêếềểễệíìỉĩịóòỏõọôốồổỗộơớờởỡợúùủũụưứừửữựýỳỷỹỵ'
    allowed_chars = r'[^a-z0-9' + vietnamese_chars + r'\s.,?!:;\'"]'
    s = re.sub(allowed_chars, ' ', s)
    # Remove extra whitespace
    s = re.sub(r'\s+', ' ', s).strip()
    # Remove any leading/trailing punctuation and whitespace
    punctuation_to_trim = r'.,?!:;\'"\s'
    s = re.sub(r'^[' + punctuation_to_trim + r']+|[' + punctuation_to_trim + r']+$', '', s)

    # Tokenization
    tokens = word_tokenize(s)
    s = " ".join(tokens) # Join tokens back into a string

    return s
```

STUDIO
SHODWE

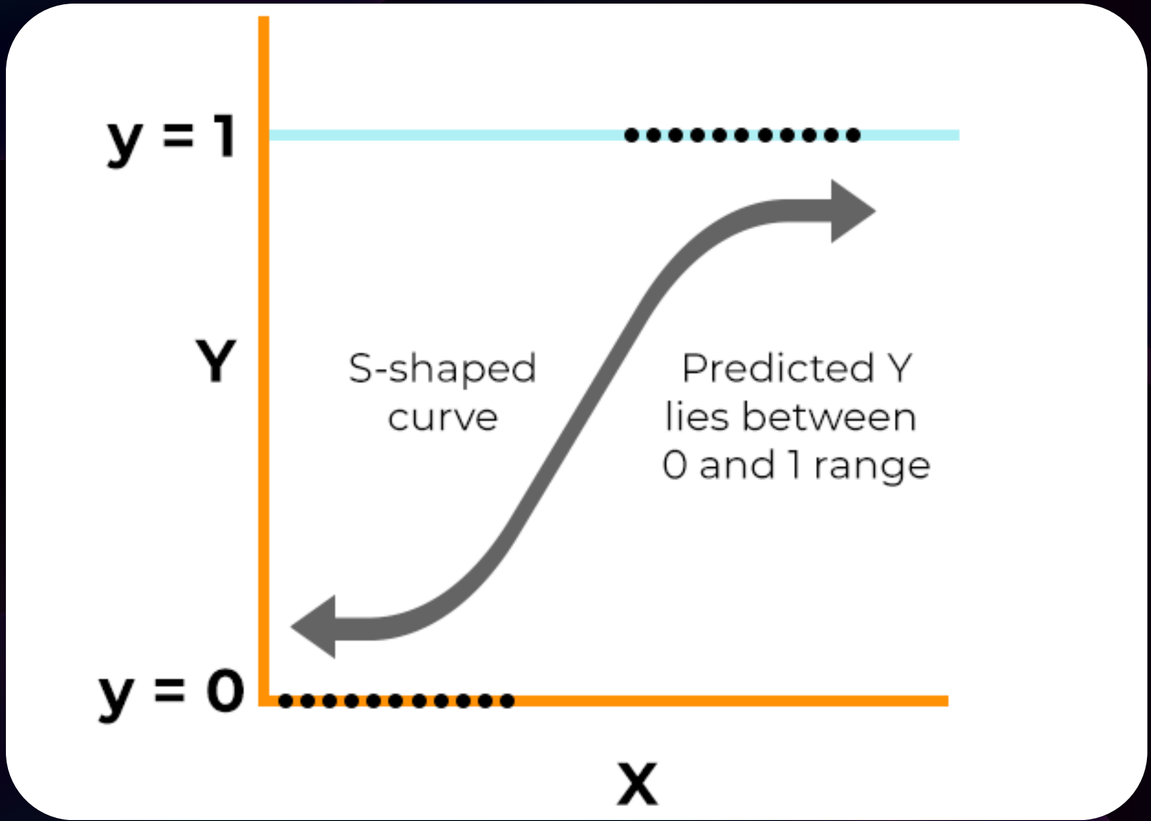
OUR MODEL

PhoBERT → Bi-LTSM → Multinomial Logistic Regression

STUDIO
SHODWE

LOGISTIC REGRESSION

Multinomial logistic regression models the relationship between a categorical dependent variable with more than two outcomes and one or more independent variables.



The model creates K−1 equations comparing each class to the reference class

$$\log\left(\frac{P(Y=k)}{P(Y=K)}\right) = \beta_{0k} + \beta_{1k}X_1 + \cdots + \beta_{pk}X_p$$

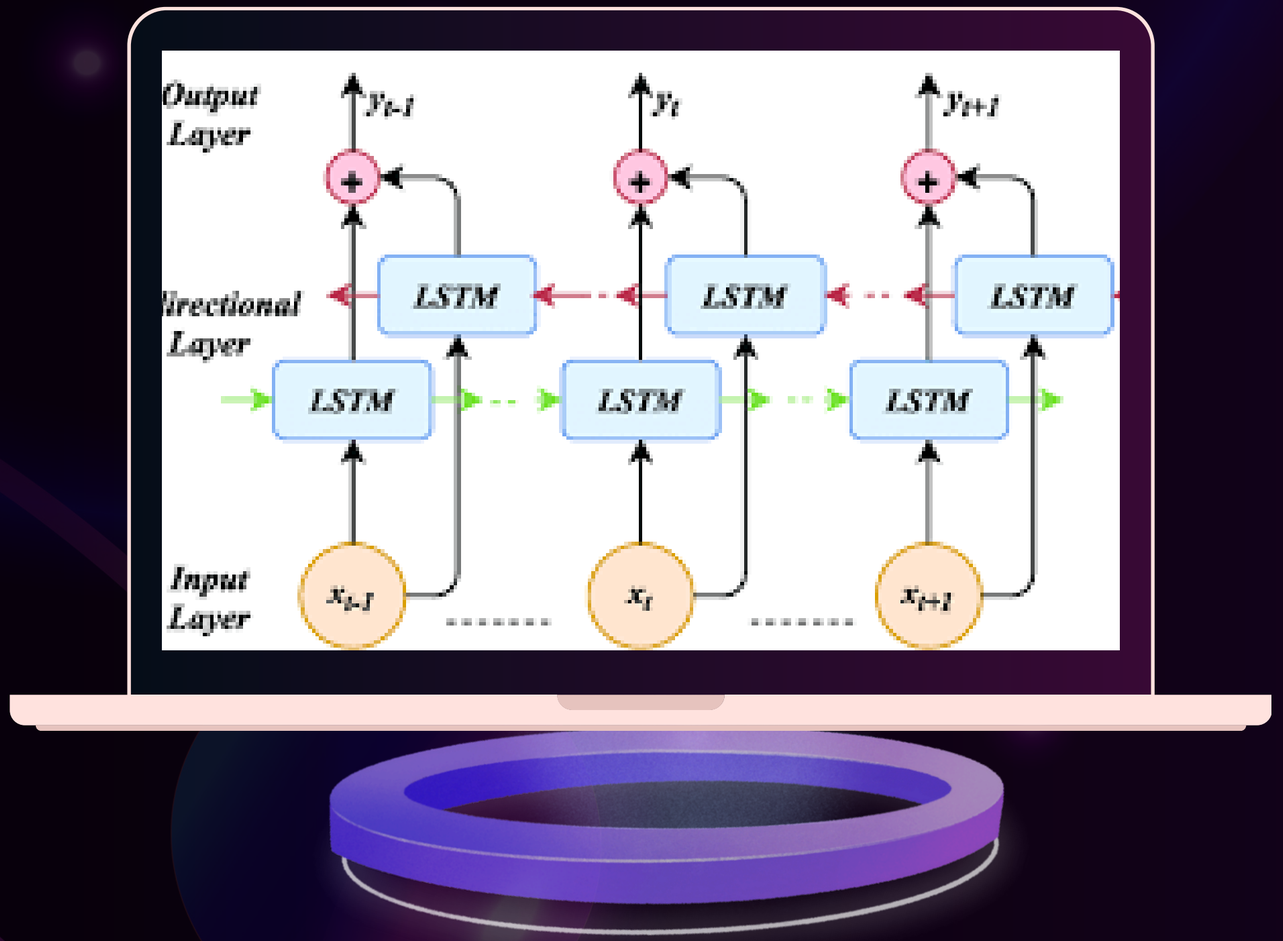
The predicted probabilities is computed via the softmax function

$$P(Y=k) = \frac{e^{X\beta_k}}{\sum_{j=1}^{K} e^{X\beta_j}}$$

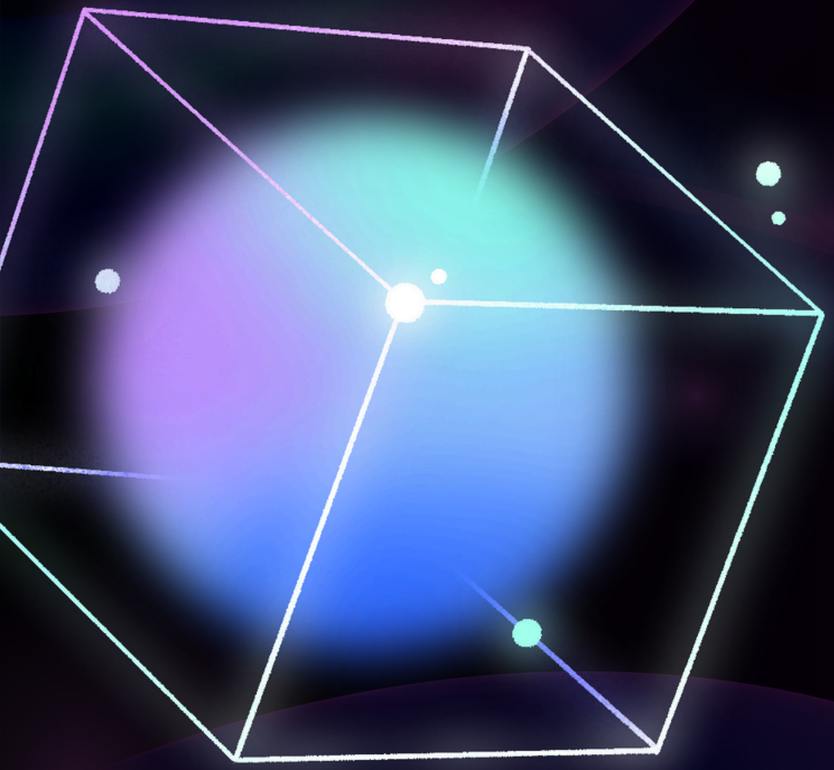
STUDIO
SHODWE

BI-LSTM

A type of recurrent neural network (RNN) that extends the standard LSTM (Long Short-Term Memory) network by processing input sequences in both forward and backward directions



STUDIO
SHODWE



PHOBERT

- A pre-trained transformer-based model specifically designed for Vietnamese text
- Based on the BERT architecture but optimized to suit the linguistic and structural characteristics of Vietnamese.
- Ultilizing the **Transformer architecture**

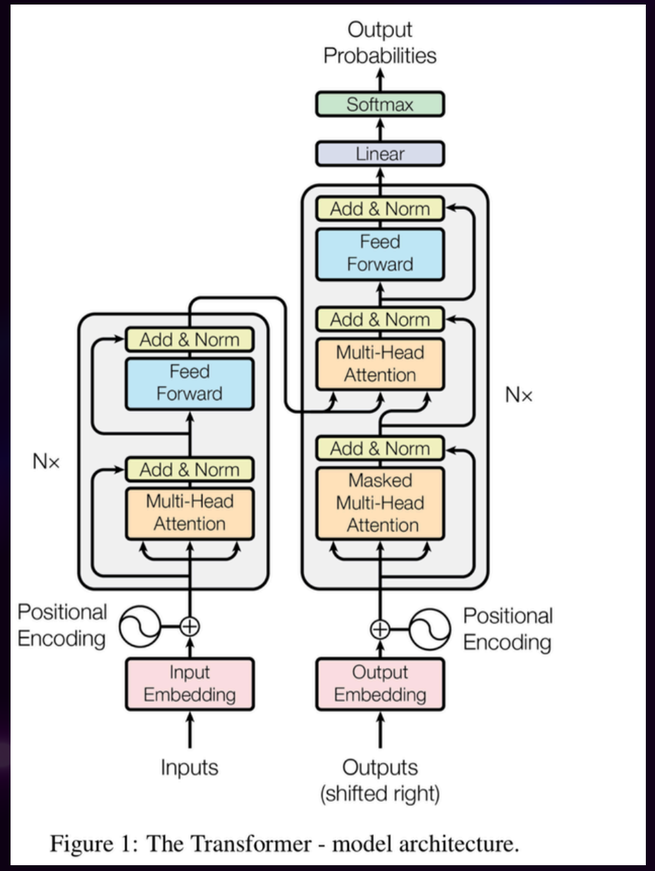
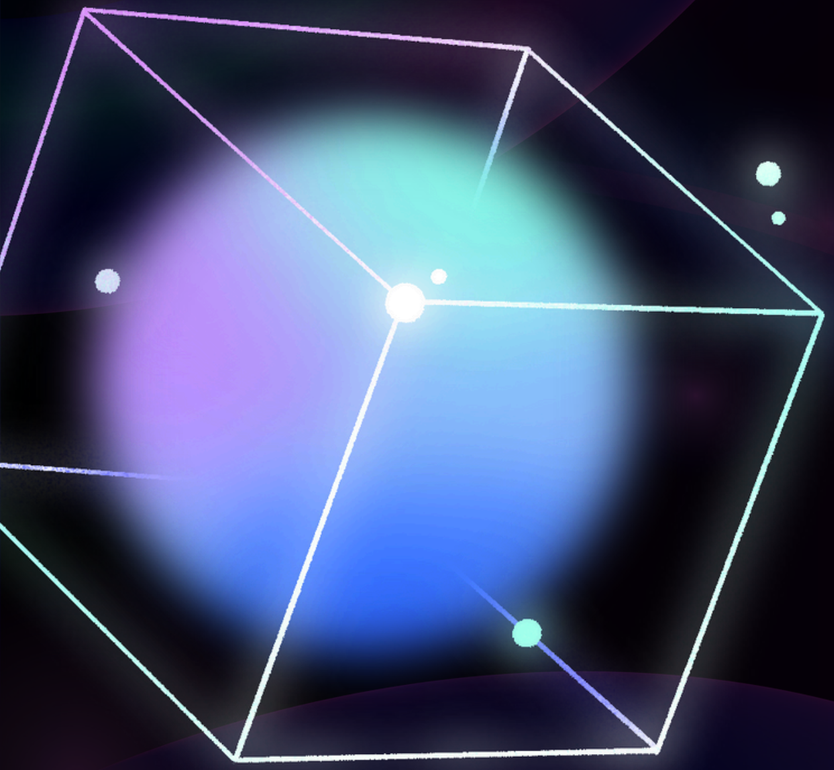


Figure 1: The Transformer - model architecture.

STUDIO
SHODWE



KEY FEATURES

Specialized for Vietnamese

Trained on a large Vietnamese corpus, → optimized for handling the unique characteristics of Vietnamese

Self-Attention Mechanism

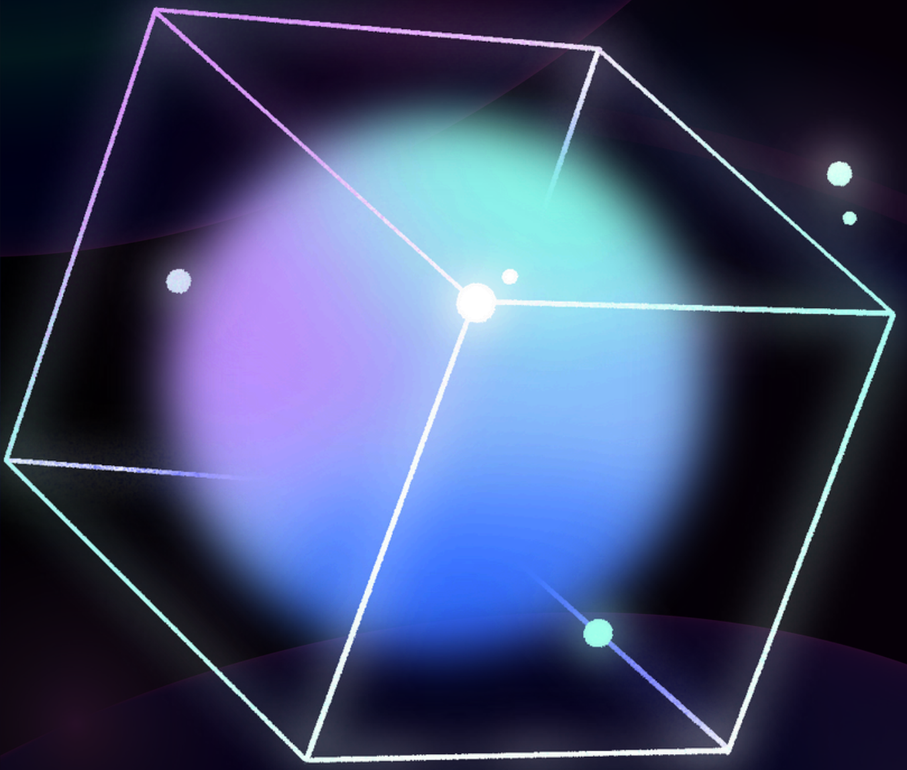
Allows to capture long-range dependencies in text → essential for emotion detection where context across the sentence matters.

Model Architecture

12 layers in based model
12 attention heads per layer → focus on different aspects of input text
768 hidden units per layer → represent complex emotion features

STUDIO
SHODWE

PROCESSING STEPS



Tokenization

Breaks down text into subwords/tokens.

Self-Attention

Evaluates each token's importance in relation to the others.

Feedforward Network

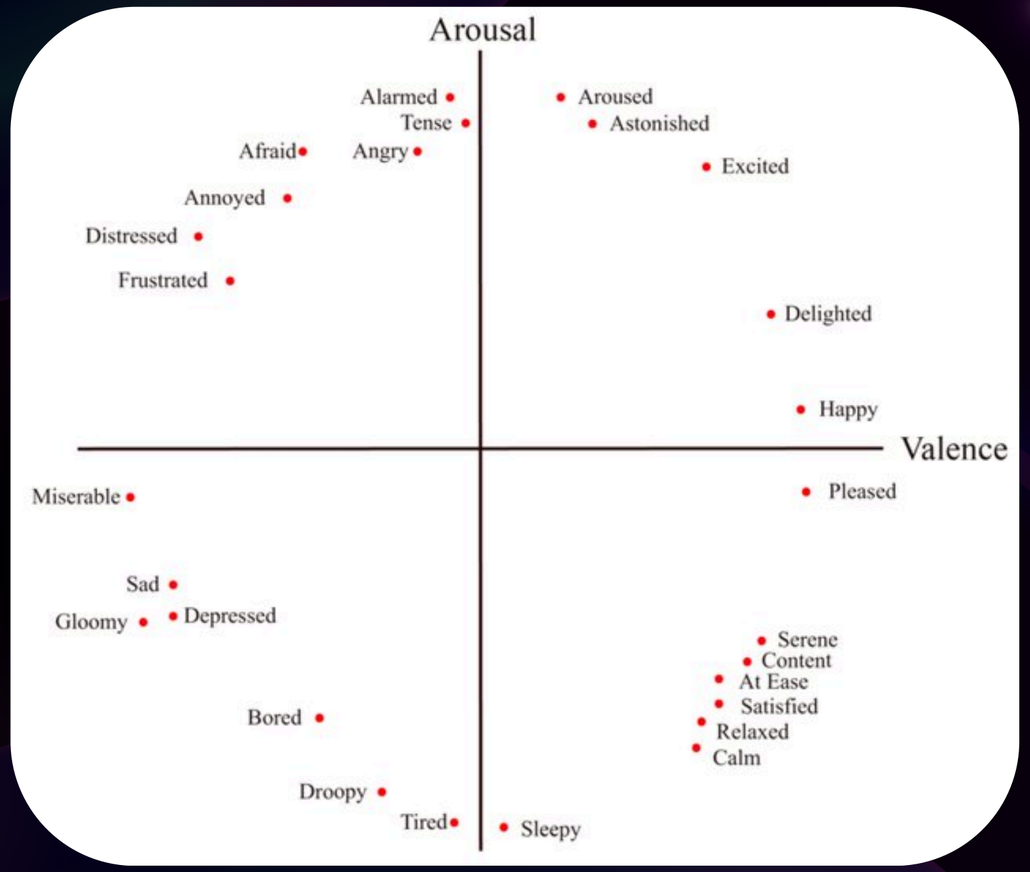
Further processes token representations to learn complex relationships.

Classification

Uses a softmax activation technique to generate emotion probabilities.

STUDIO
SHODWE

EMOTION MAPPING



Emotion is mapped into **Valence** and **Arousal** follows Russell's circumplex model.

Valence is the emotional value of a feeling—how positive (pleasant) or negative (unpleasant) it is.

Arousal is the level of physical or mental activation—ranging from calm (low) to excited (high).

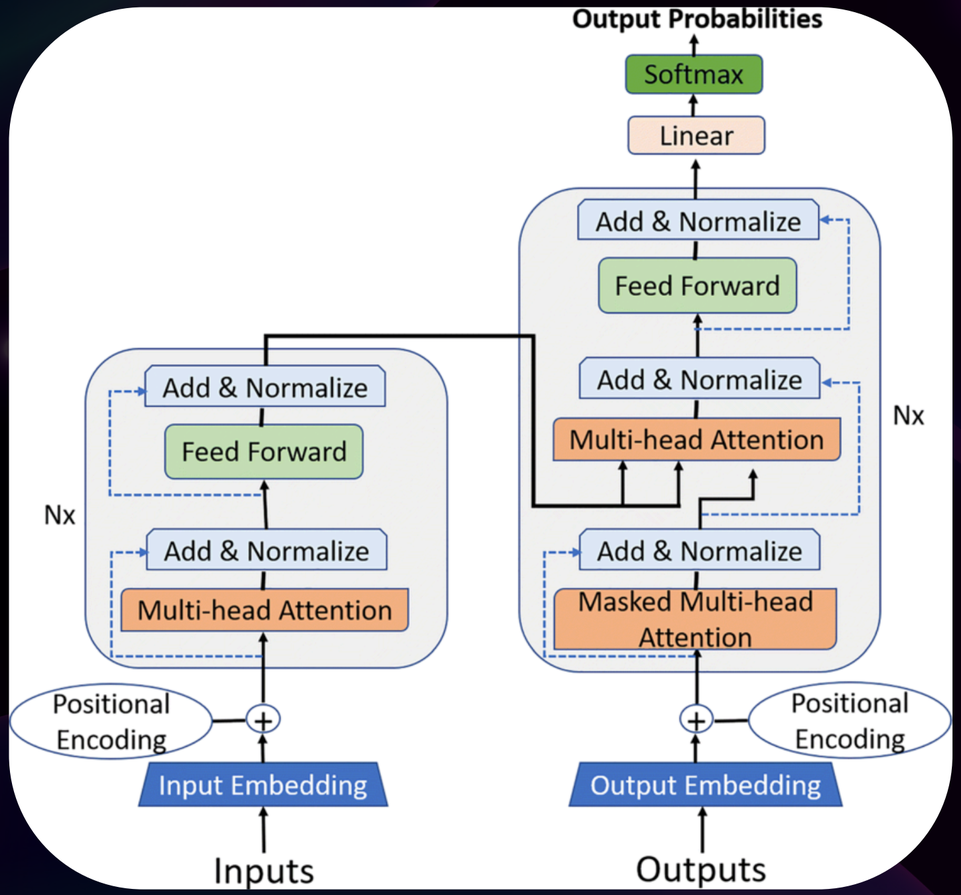
$$A = \sum_{i=0}^{5} p_i y_{Ai} \quad \text{and} \quad V = \sum_{i=0}^{5} p_i y_{Vi}$$

A = Σ_(I=0)^5 [P_I X_AI]

V = Σ_(I=0)^5 [P_I X_VI]

STUDIO
SHODWE

TRANSFORMER



Transformers generate music by first converting musical scores into a sequence of "tokens" representing notes, timing, and dynamics, much like words in a sentence. .

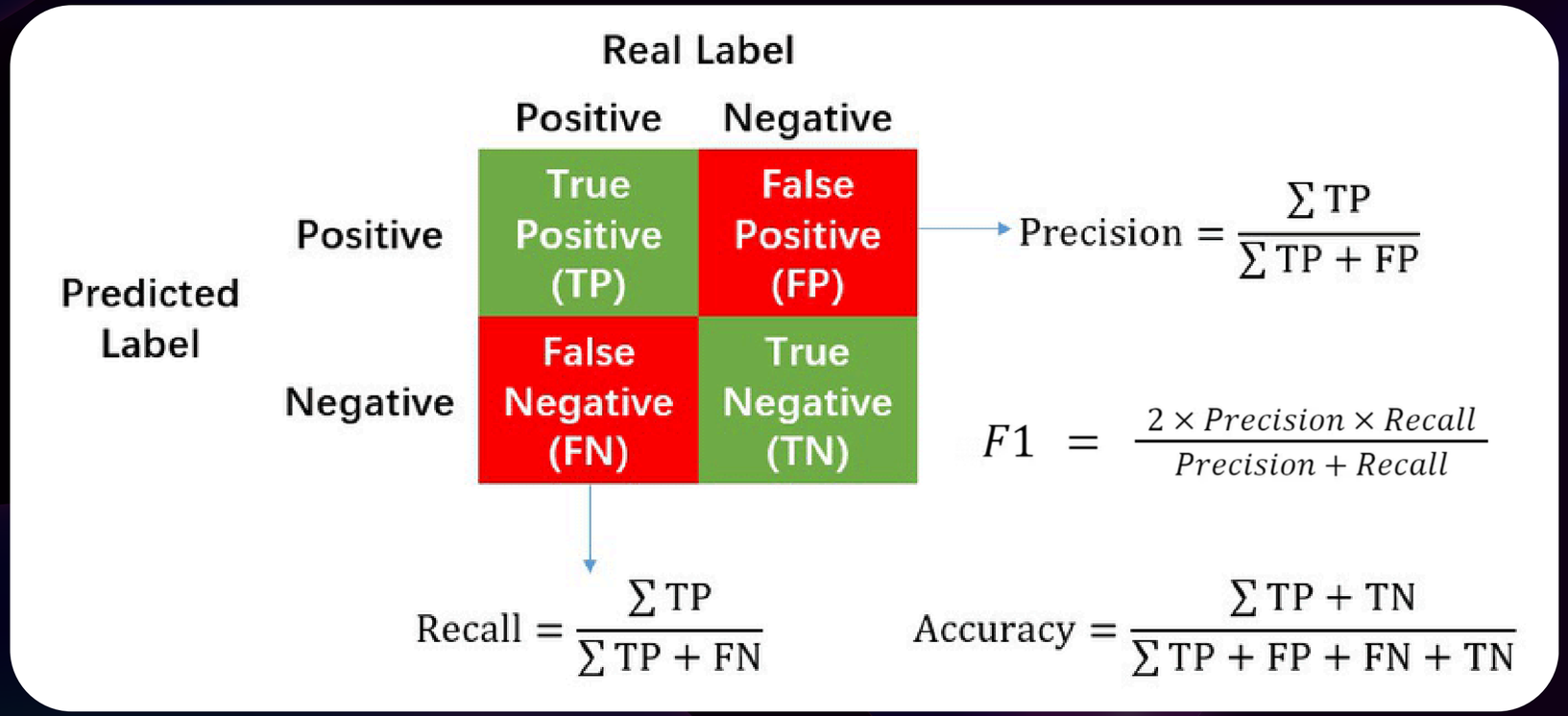
The model's self-attention mechanism then processes this entire sequence at once, allowing it to learn complex, long-range musical structures and dependencies.

To create emotion-driven music, special tokens representing emotional states like valence and arousal are added as a condition, guiding the Transformer to generate a new musical piece that reflects the specified feeling

STUDIO
SHODWE

EVALUATION METRICS

Emotion Recognition



EVALUATION METRICS

Music Generation

Quantitative approach: Inspect the raw token output to identify technical failures like repetitive patterns and lack of diversity

Qualitative approach: Assess the final audio for musical coherence, creativity, and its ability to evoke the intended emotion

EXPERIMENTAL RESULTS

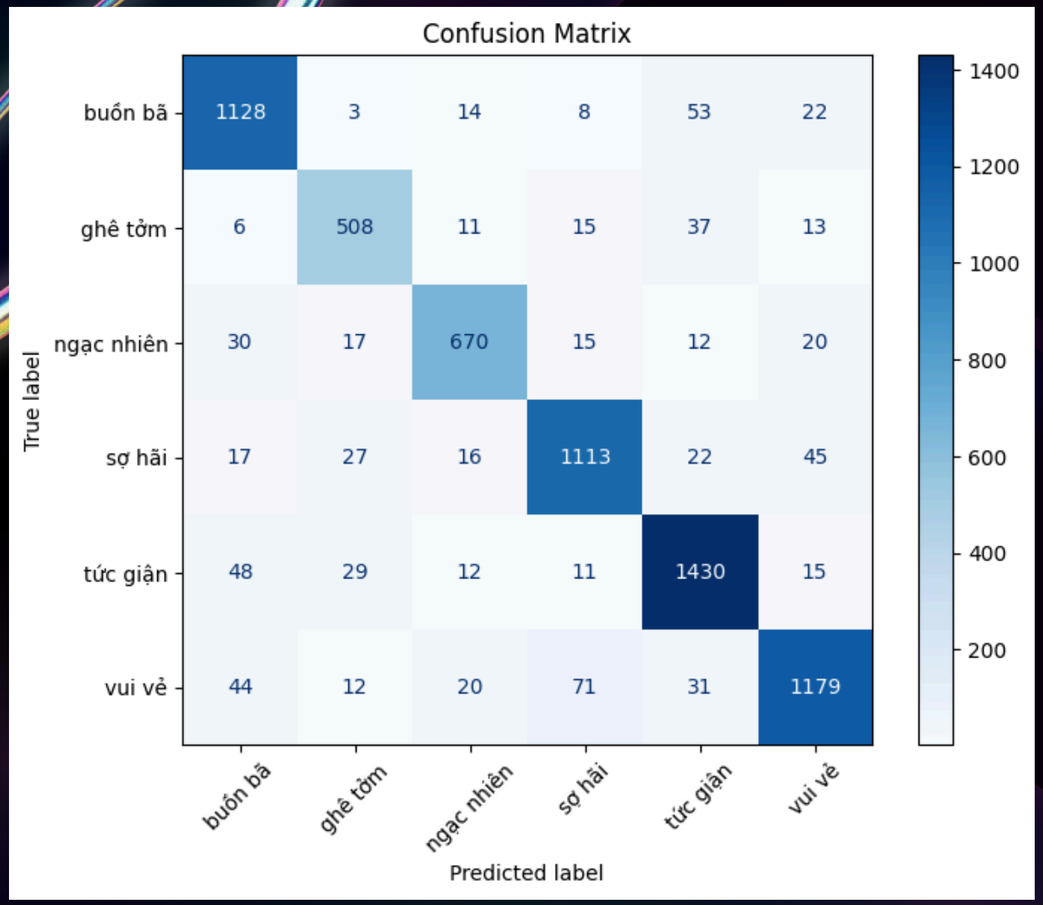
RESULTS

- **Accuracy (90%):** PhoBERT correctly predicts the emotion (regardless of type) in 90% of cases.
- **Precision (89%):** When PhoBERT identifies a text as expressing emotion X, it is correct 94% of the time.
- **Recall (89%):** Of all texts actually expressing emotion X, PhoBERT correctly identifies 93%. This helps the GAN not miss opportunities to generate music that aligns with the user's actual emotion.
- **F1-Score (90%)**: The balanced measure of Precision and Recall, at 94%.

RESULTS OF EACH LABELS

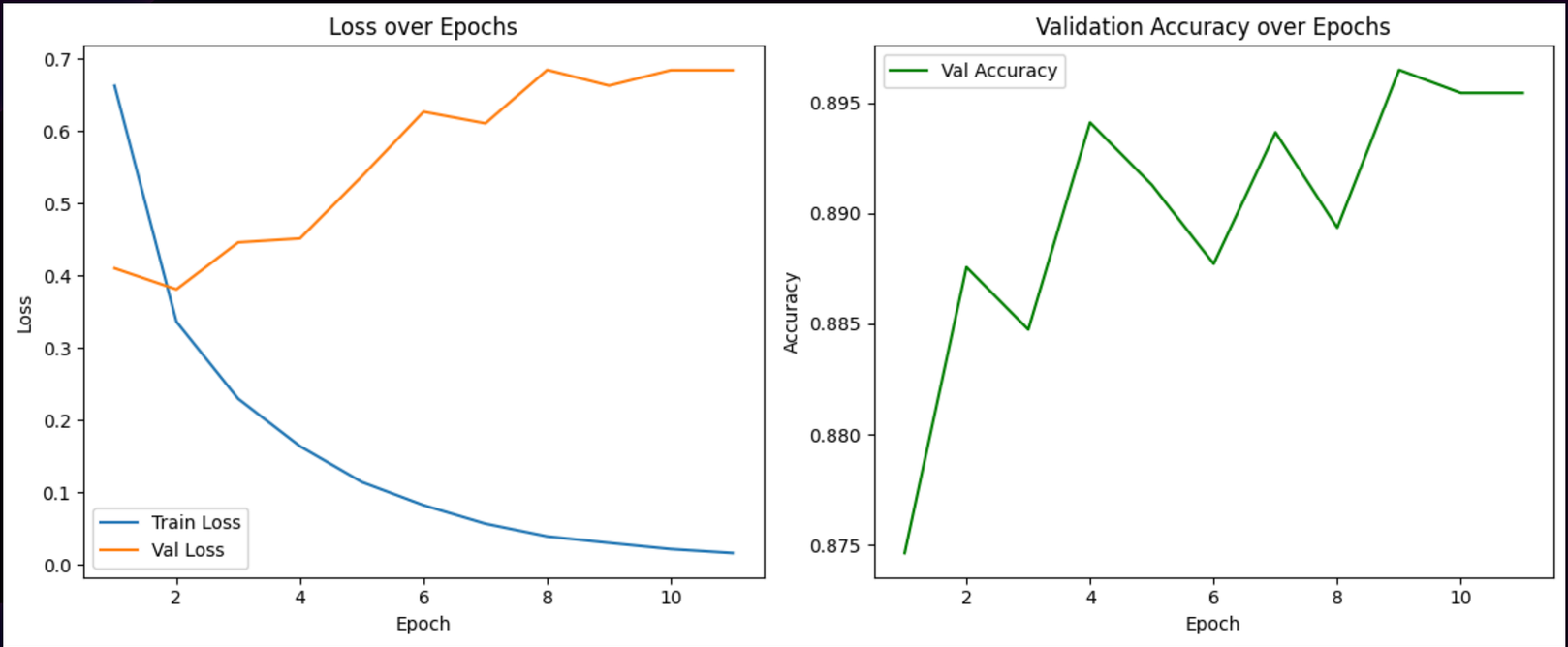
Result parameters	Sadness	Disgust	Surprise	Fear	Anger	Happiness
Precision	89%	85%	90%	90%	90%	91%
Recall	92%	86%	88%	90%	93%	87%
F1-Score	90%	86%	89%	90%	91%	89%

CONFUSION MATRIX



- The matrix shows good diagonal dominance, with high correct predictions for labels like "angry" (1430), "happy" (1179), and "sad" (1128).
- Notable misclassification: The most frequent error was between "happy" and "fearful", with 71 "happy" cases misclassified.
- Mislabels occurred between "sad" and "angry" (53 ↔ 48), and from "happy" to "sad" (44 cases).
- These errors are understandable, given overlapping emotional expressions and lexical similarities in sentiment analysis.

LEARNING CURVES



- TRAINING LOSS DECREASED STEADILY → EFFECTIVE LEARNING.
- VALIDATION LOSS ROSE AFTER EPOCH 8 → OVERFITTING → EARLY STOPPING USED.
- BEST ACCURACY AT EPOCH 9 → MODEL SAVED AT OPTIMAL POINT.

MUSIC GENERATION

```
Starting fine-tuning data preparation with chunk size 1024...
Processing labelled MIDI files: 100% ████████████████████ 204/204 [00:05<00:00, 39.20it/s]
Fine-tuning data preparation complete. Total chunks: 1012
Successfully loaded pre-trained weights for fine-tuning.
--- Starting Fine-tuning (1012 samples) for 4 epochs ---
Fine-tuning Epoch 1: 100% ████████████████████ 127/127 [00:33<00:00,  4.33it/s]
/usr/local/lib/python3.10/dist-packages/torch/nn/functional.py:5849: UserWarning: Support for mismatched key_padding_mask
  warnings.warn(
Fine-tuning Epoch 1 | Avg Loss: 0.0261
Model saved after epoch 1 to /kaggle/working/models/music_transformer_finetuned.pth
Fine-tuning Epoch 2: 100% ████████████████████ 127/127 [00:32<00:00,  4.59it/s]
Fine-tuning Epoch 2 | Avg Loss: 0.0150
Model saved after epoch 2 to /kaggle/working/models/music_transformer_finetuned.pth
Fine-tuning Epoch 3: 100% ████████████████████ 127/127 [00:32<00:00,  4.50it/s]
Fine-tuning Epoch 3 | Avg Loss: 0.0121
Model saved after epoch 3 to /kaggle/working/models/music_transformer_finetuned.pth
Fine-tuning Epoch 4: 100% ████████████████████ 127/127 [00:33<00:00,  4.49it/s]
Fine-tuning Epoch 4 | Avg Loss: 0.0110
Model saved after epoch 4 to /kaggle/working/models/music_transformer_finetuned.pth
--- Fine-tuning Complete ---
```

STUDIO
SHODWE

DEPLOYMENT

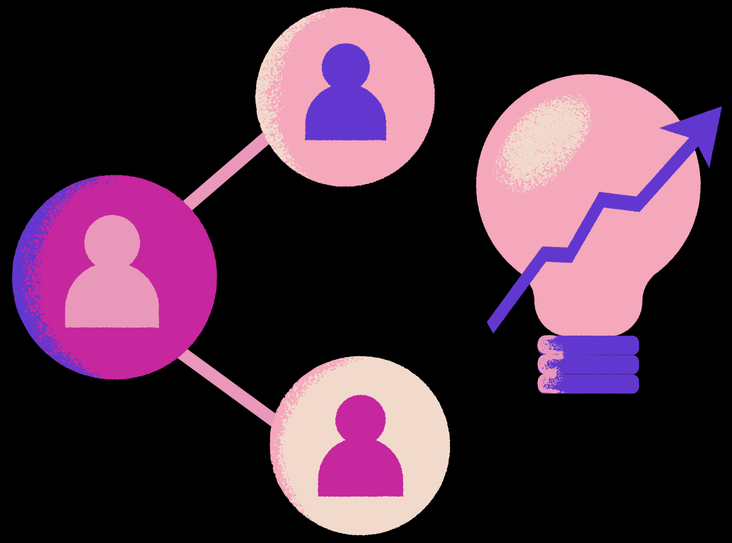
STUDIO
SHODWE

DEPLOYMENT

- **Frontend** : HTML&CSS
- **Backend** : Flask
- **Input** : Vietnamese text
- **Output** : Predicted Emotion + Music

2025

KEY FINDINGS



1. The system successfully identified **6 types** of emotions.
2. PhoBERT gives the best results in Vietnamese emotion recognition.
3. Multinominal Logistic Regression and LSTM works well but needs improvement to increase accuracy.
4. Developing a novel dataset of Vietnamese text with six main emotions labeled manually
5. Automatic pipeline completion: text → emotion → music.

2025

Conclusion

LIMITATIONS

1. The emotion dataset is quite small, unbalanced between classes.
2. The model cannot handle complex emotions (anxiety, regret, sarcasm...)
3. The core GAN-based music generation model is still under development

FUTURE WORK

1. Expand and improve the emotion dataset.
2. Training and evaluating conditional GAN model.
3. Complete a user-friendly web interface for emotion-driven music generation from Vietnamese text.

REFERENCES

NANDWANI, P. & VERMA, R. 2021. A review on sentiment analysis and emotion detection from text. Social network analysis and mining, 11, 81.

MEDHAT, W., HASSAN, A. & KORASHY, H. 2014. Sentiment analysis algorithms and applications: A survey. Ain Shams engineering journal, 5, 1093-1113.

DONG, H.-W., HSIAO, W.-Y., YANG, L.-C. & YANG, Y.-H. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. Proceedings of the AAAI conference on artificial intelligence, 2018.

NGUYEN, D. Q. & NGUYEN, A. T. 2020. PhoBERT: Pre-trained language models for Vietnamese. arXiv preprint arXiv:2003.00744.

BAYAGA, A. 2010. Multinomial Logistic Regression: Usage and Application in Risk Analysis. Journal of applied quantitative methods, 5.

HUANG, Z., XU, W. & YU, K. 2015. Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991.

THANK YOU!