Vietnam National University - Ho Chi Minh City
University of Information Technology
Faculty of Computer Science

FINAL PROJECT REPORT

Artificial Intelligence Thinking - AI002

Project:

Multimodal Sarcasm Detection on Vietnamese Social Media Posts

Instructor:

PhD. Ngo Duc Thanh

Group Members:

Ha Huy Hoang	22520460
Dang Vinh Hoi	22520490
Nguyen Duy Hoang	22520467
Tang Gia Han	22520394
Nguyen Xuan Bach	22520093

Ho Chi Minh City, January 2025

Contents

1. Problem

1.1. Introduction

In social media groups and communities, particularly those focused on education or specific topics, posting content with sarcastic intent can lead to undesirable outcomes. Such posts not only risk violating community standards but may also compromise the professionalism and constructive nature of the environment. This creates significant challenges for administrators and moderators, who must ensure the appropriateness of shared content, often relying on time-consuming manual review processes.

As the size of these groups increases, with a growing number of members and posts, manual moderation becomes inefficient and difficult to scale. To address this issue, we propose a solution in the form of a Multimodal Sarcasm Detection system. The goal of this system is to automatically classify multimedia posts on social media into two categories: **sarcasm** and **not-sarcasm**.

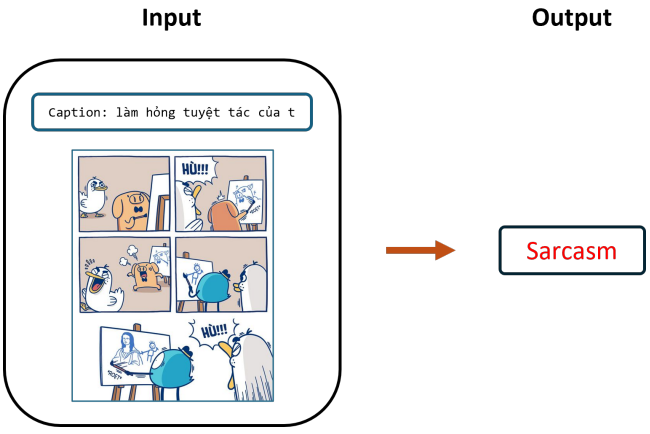
The proposed system operates as follows:

- Posts classified as **sarcasm** are automatically removed to reduce potential negative impacts.
- Posts classified as **not-sarcasm** are forwarded for manual review by administrators and moderators.

This solution aims to streamline the moderation process, significantly reduce the workload for content moderators, and enhance the management of social media groups and communities, particularly in high-volume environments.

1.2. Problem Statement

- **Input:** A post consists of the following two components:
  - **Caption:** Text associated with the post.
  - **Image:** A single picture (e.g., photo or graphic) associated with the post.
- **Output:** One of the following two labels:
  - **sarcasm:** If the post contains sarcasm based on the analysis of the caption and image.
  - **not-sarcasm:** If the post does not contain sarcasm, indicating a straightforward or literal interpretation of the caption and image.



- **Constraints:**
  - The image must have a minimum height and width of 224x224 pixels (i.e., height and width should be greater than or equal to 224 pixels).
  - If the image contains text, the text within the image must not exceed 256 units (word, character, or emoji).
  - The text (caption) is required to contain at least 1 unit (word, character, or emoji) and no more than 256 units (word, character, or emoji).
  - The text (caption) is recommended to be in standard Vietnamese but may include slang, teencode, or foreign language elements.
- **Requirements:**
  - Must achieve an accuracy of at least 75% as measured by the accuracy metric on the test set provided by the customer.
  - The total training and inference time must not exceed 2 hours, with an inference time not exceeding 0.05 seconds per sample, running on Kaggle's Tesla P100-PCIE-16GB GPU (15.89 GB memory, 56 multiprocessors).
  - The inference time, measured from submitting the complete input (image and caption) to receiving the result, must not exceed 5 seconds when deployed via a Streamlit web application on the specified configuration (Model name: AMD EPYC 7B13, CPU frequency: 2449.998 MHz, maximum 2 CPU cores, 2.7GB memory, and up to 50GB storage).

2. Computational Thinking Process

2.1. Overview

Computational thinking is a problem-solving process that involves a set of techniques and approaches designed to break down complex problems, identify patterns, and devise effective solutions. It is a fundamental skill that bridges the gap between human thinking and computational systems. This process consists of the following four key steps:

- **Decomposition:** Breaking down a complex problem into smaller, more manageable components. This step helps to focus on individual parts of the problem, making it easier to analyze and solve.
- **Pattern Recognition:** Analyzing and identifying recurring patterns or similarities within the problem. Recognizing patterns helps to reuse existing solutions or insights, thereby improving efficiency and reducing redundancy.
- **Abstraction:** Identifying the essential details of the problem while ignoring irrelevant information. This step ensures that the problem is simplified without losing critical aspects required for finding a solution.
- **Algorithm Design:** Developing a step-by-step process or set of rules to solve the problem. This step ensures that the solution can be implemented systematically and is repeatable for similar problems in the future.

2.2. Hierarchical Structure

The sarcasm detection system is organized in a hierarchical structure that reflects the decomposition of the problem into manageable components. This structure consists of multiple levels, where each level represents a different degree of abstraction and complexity.

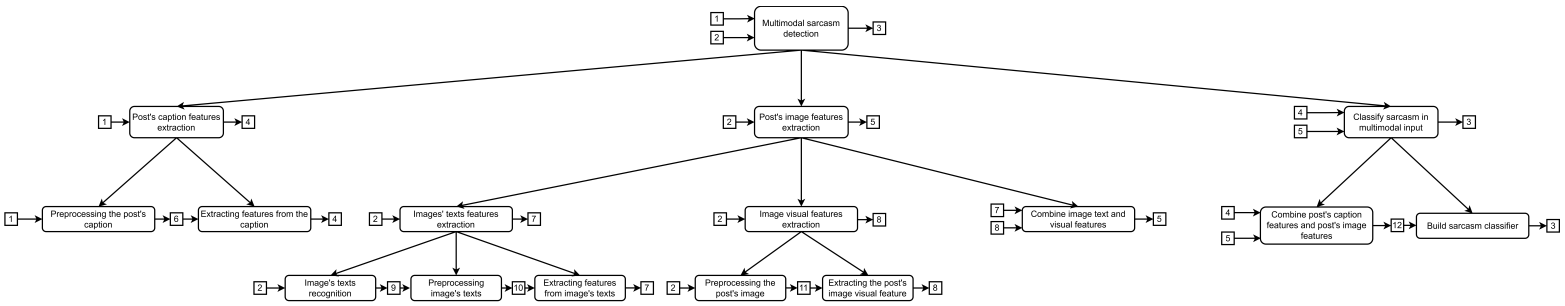


Figure 1: Hierarchical structure of the sarcasm detection system.

To provide further clarity, the nodes in the hierarchical decomposition tree are detailed in the following figure. Each node represents an input-output process, outlining the specific roles within the system.

1. Post's caption
2. Post's image
3. Labels: sarcasm, not-sarcasm
4. Post's caption features
5. Post's image features
6. Post's caption processing
7. Image's texts linguistic features
8. Post's image visual features
9. Texts from the post's image
10. Image's texts processing
11. Post's image processing
12. Fused multimodal features

Figure 2: Input-output node details in the hierarchical structure.

2.3. Application of Computational Thinking Components

The problem of sarcasm detection is decomposed into smaller, well-defined tasks to ensure a systematic and manageable approach. Each subproblem is detailed below along with its associated pattern recognition and abstraction steps:

1. Text Feature Extraction

- **Preprocessing Captions:**

  1. Remove noise such as special characters, hashtags, and hyperlinks.
  2. Perform tokenization using techniques like WordPiece or SentencePiece.
  3. Normalize text to handle variations in casing and spelling.

  Pattern Recognition:

  – Identify recurring linguistic patterns, such as frequent use of hyperbolic expressions (e.g., "Absolutely amazing!").
  – Recognize specific text markers, including idioms or slang, that often signal sarcasm.
  – Detect contradictions within a caption (e.g., positive sentiment followed by a negation clause).

  Abstraction:

  – Transform raw captions into semantic embeddings using pretrained language models like VS-BERT or Multilingual-CLIP.

– Abstract away syntactic noise and focus on core linguistic features such as sentiment, emotion, and contextual relationships.

- **Generating Semantic Embeddings:**

1. Pass preprocessed text through pretrained models to generate dense feature vectors.
2. Capture contextual information, word dependencies, and sentiment markers within the embeddings.

Pattern Recognition:

– Learn text representations that highlight sarcasm-inducing constructs, such as irony or satire.
– Focus on embeddings that emphasize contrasts between emotions or sentiments.

Abstraction:

– Represent text as fixed-dimensional vectors, removing low-importance tokens or features.
– Simplify embeddings to focus on sarcasm-relevant characteristics, such as contextual polarity shifts.

2. Image Feature Extraction

- **Recognizing Embedded Text:**

1. Use OCR tools to detect and extract text embedded within images.
2. Preprocess the extracted text to remove noise and tokenize it for further analysis.

Pattern Recognition:

– Identify consistent patterns in embedded text, such as sarcastic captions added to memes.
– Recognize visual layouts that frequently contain sarcasm-related text (e.g., text overlays on humorous images).

Abstraction:

– Represent embedded text as high-dimensional vectors after preprocessing.
– Disregard image regions without relevant text content.

- **Extracting Visual Features:**

1. Preprocess images by resizing, normalizing pixel values, and applying augmentation techniques.
2. Use pretrained models such as Vision Transformers (ViT) to extract high-level visual features.

Pattern Recognition:

– Recognize facial expressions, gestures, or exaggerated poses indicative of sarcasm.
– Detect inconsistencies or ironic elements within the visual context (e.g., a happy face in a negative scenario).

Abstraction:

– Convert images into fixed-length feature vectors while discarding unnecessary background details.
– Focus on sarcasm-relevant visual features, such as facial expressions or central objects in the image.

3. **Multimodal Integration**

- **Feature Alignment and Fusion:**

  1. Align text and image features to ensure consistent representation dimensions.
  2. Use dense layers or attention mechanisms to fuse multimodal features into a unified representation.

  Pattern Recognition:

  – Identify cross-modal patterns, such as contradictions between textual and visual content.

  – Recognize synergistic relationships where sarcasm emerges from the combination of modalities (e.g., a serious image paired with a sarcastic caption).

  Abstraction:

  – Create a lower-dimensional representation of fused features to improve efficiency and reduce noise.

  – Focus on abstracted multimodal features that encapsulate sarcasm cues across both text and image modalities.

4. **Classification**

- **Designing and Training the Classifier:**

  1. Input the unified multimodal representation into dense layers for classification.
  2. Train the model to output probabilities for each class (e.g., sarcasm vs. non-sarcasm).

  Pattern Recognition:

  – Detect consistent predictive patterns in the combined feature space for sarcasm detection.

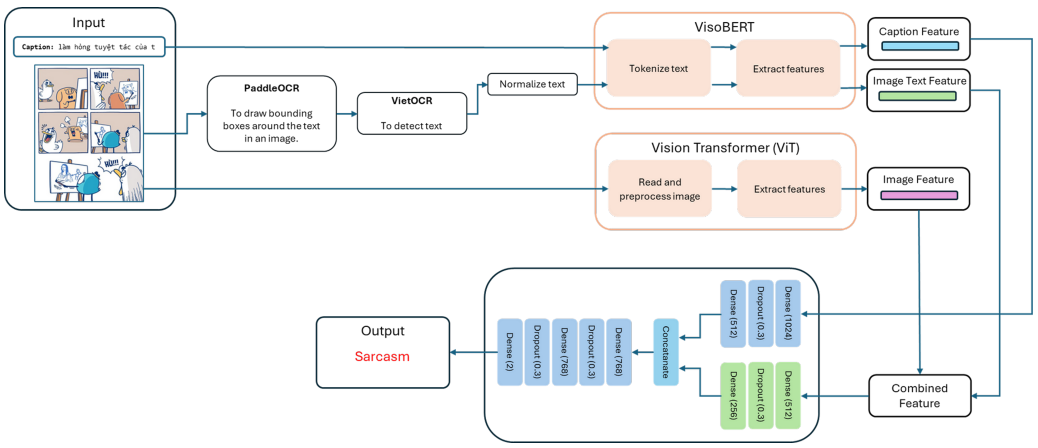
  – Focus on high-impact feature interactions that drive the classifier's decisions.

  Abstraction:

  – Abstract the classification process into probabilities for each label.

  – Generalize the model to work across diverse examples by emphasizing robust patterns in the training data.

2.4. Algorithm Design



Here is a detailed breakdown of the algorithm design:

- **PaddleOCR[2]**: This component uses optical character recognition (OCR) to detect and draw bounding boxes around the text present in the input image.
- **VietOCR[4]**: This module extracts the text contained within the bounding boxes identified by PaddleOCR. It applies advanced OCR methods to accurately transcribe the text.
- **Normalize Text**: This module for normalize the text detection from VietOCR by PyVietnameseTextNormalizer[3]
- **VisoBERT[5]**: This component processes the text extracted by VietOCR as well as the caption. It tokenizes, normalizes the text, and extracts linguistic features to represent the textual information effectively.
- **Vision Transformer (ViT)[1, 6]**: This module analyzes the input image itself in parallel. It preprocesses the image and uses a vision transformer model to extract visual features from the image.
- **Caption Feature**: This is the feature representation of the input caption extracted using VisoBERT.
- **Image Text Feature**: This is the feature representation of the text extracted from the image using VisoBERT.
- **Image Feature**: This is the feature representation of the visual content of the image, extracted using the Vision Transformer (ViT).
- **Combined Feature**: This component combines the *Image Text Feature* and *Image Feature* into a single feature vector, capturing both textual and visual aspects of the input image.
- **Neural Network**: A neural network model processes the *Combined Feature* along with the *Caption Feature*. This model classifies the input into one of two classes:
  - **Sarcasm**: Indicates sarcasm in the input image-caption pair.
  - **Not-Sarcasm**: Indicates no sarcasm in the input image-caption pair.

This hybrid approach, combining rule-based computer vision/NLP techniques with a neural network component, allows the system to leverage the strengths of both paradigms to robustly analyze the input and generate the appropriate output. In summary, the full algorithm design includes:

- **PaddleOCR** for text bounding box.
- **VietOCR** for text detection.
- **Normalize Text** for normalize text
- **VisoBERT** for text feature extraction.
- **Vision Transformer (ViT)** for image feature extraction.
- **Caption Feature** for representing caption-based features.
- **Image Feature** for representing visual features.
- **Image Text Feature** for representing text features extracted from the text in image.
- **Combined Feature** for integrating visual and textual features.
- **Neural Network** for final classification into *Sarcasm* or *Not Sarcasm*.

This multimodal approach enables the system to effectively classify whether the input (caption and image) conveys sarcasm or not.

3. Evaluation

3.1. Data

The dataset used for sarcasm detection on Vietnamese social media posts is sourced from the UIT Data Science Challenge 2024 (DSC). This dataset contains a collection of posts from social media platforms, each consisting of two components: an image and a caption. The primary task is to classify these posts as either **sarcasm** or **not-sarcasm**, based on the analysis of both the visual and textual elements.

- **Label Distribution:**
  - **not-sarcasm:** 6011 samples
  - **sarcasm:** 5896 samples
- **Dataset Split:**
  - **Training Set:** 9525 samples (80% of the total dataset).
  - **Test Set:** 2382 samples (20% of the total dataset).
- **Dataset Split Considerations:**
  - The dataset is split in a way that ensures both subsets maintain the original label distribution. This prevents bias and ensures that both the training and test sets are representative of real-world conditions.
- **Sample Components:** Each sample represents a single post, which consists of:
  - **Image:** A single picture (e.g., photo or graphic) associated with the post. The image must satisfy the constraints specified in the Problem Statement.
  - **Caption:** Text associated with the post. The text must adhere to the constraints specified in the Problem Statement.

3.2. Metrics

The evaluation of the sarcasm detection system is based on the following metrics:

3.2.1. Customer-Centric Metrics

The two core metrics used to assess the model's performance, as per the customer's requirements, are:

- **Accuracy:**
  - Measures the percentage of correctly classified samples out of the total samples.
  - Formula:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}}$$

- **Average Inference Time:**
  - Measures the average time taken to process an input post (image and caption) and produce a prediction.
  - Formula:

$$\text{Average Inference Time} = \frac{1}{m} \sum_{i=1}^{m} (\text{End Time}_i - \text{Start Time}_i)$$

where m is the total number of samples.

3.2.2. Training Evaluation Metrics

In addition to the customer-centric metrics, the following metrics are used during model training and evaluation to gain deeper insights into performance:

- **Precision:**
  - Indicates the proportion of posts predicted as sarcasm that are actually sarcasm.
  - Formula:
$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$
- **Recall:**
  - Measures the proportion of sarcasm posts that are correctly identified.
  - Formula:
$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$
- **F1-Score:**
  - A harmonic mean of precision and recall, providing a balance between the two.
  - Formula:
$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$
- **F1-Macro:**
  - Computes the F1-score for each class (**sarcasm** and **not-sarcasm**) independently and then averages them.
  - Formula:
$$\text{F1-Macro} = \frac{\text{F1-Score for Sarcasm} + \text{F1-Score for Not-Sarcasm}}{2}$$

3.3. Experimental Results

The following section presents the experimental results of the sarcasm detection model, including performance statistics for processing and prediction times, as well as the classification performance evaluated on the test set. The evaluation was conducted using a Tesla P100 GPU in the Kaggle environment and the code training can see at here.

Number of test	Processing Time	Prediction Time
2382 (pair image and caption)	109.56s	0.88s

Figure 3: Performance statistics for processing and prediction times on the test set, evaluated using a Tesla P100 GPU in the Kaggle environment.

	precision	recall	f1-score	support
not-sarcasm	0.70	0.78	0.74	1071
sarcasm	0.80	0.72	0.76	1311
accuracy			0.75	2382
macro avg	0.75	0.75	0.75	2382
weighted avg	0.76	0.75	0.75	2382

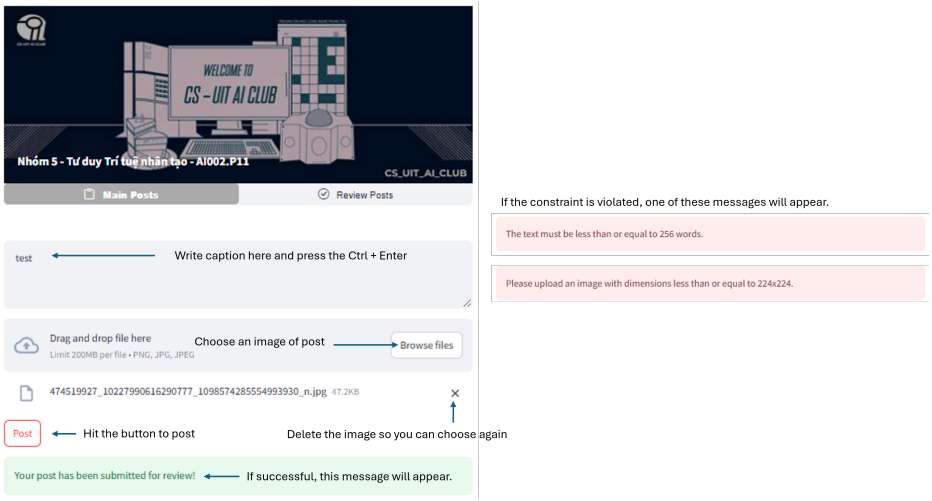
Figure 4: Classification report for the model evaluated on the test set.

3.4. Web demo

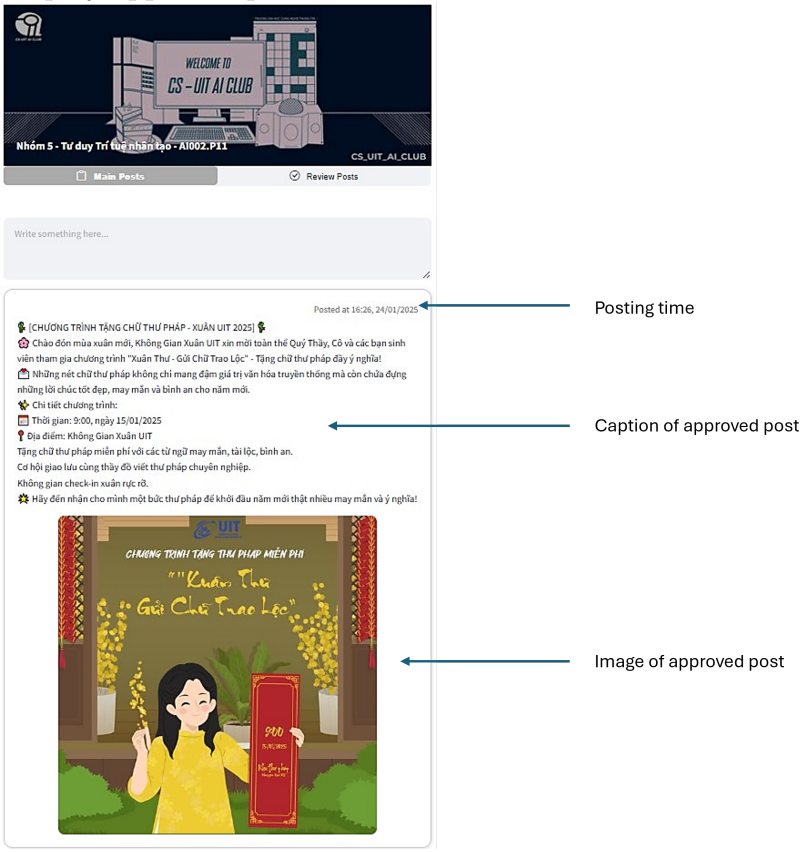
Web written using Streamlit (a Python frontend framework), simulating the basic features of a Facebook-like a Group. The web will have two main tabs:

- **Main Posts**: Used to post and view approved posts

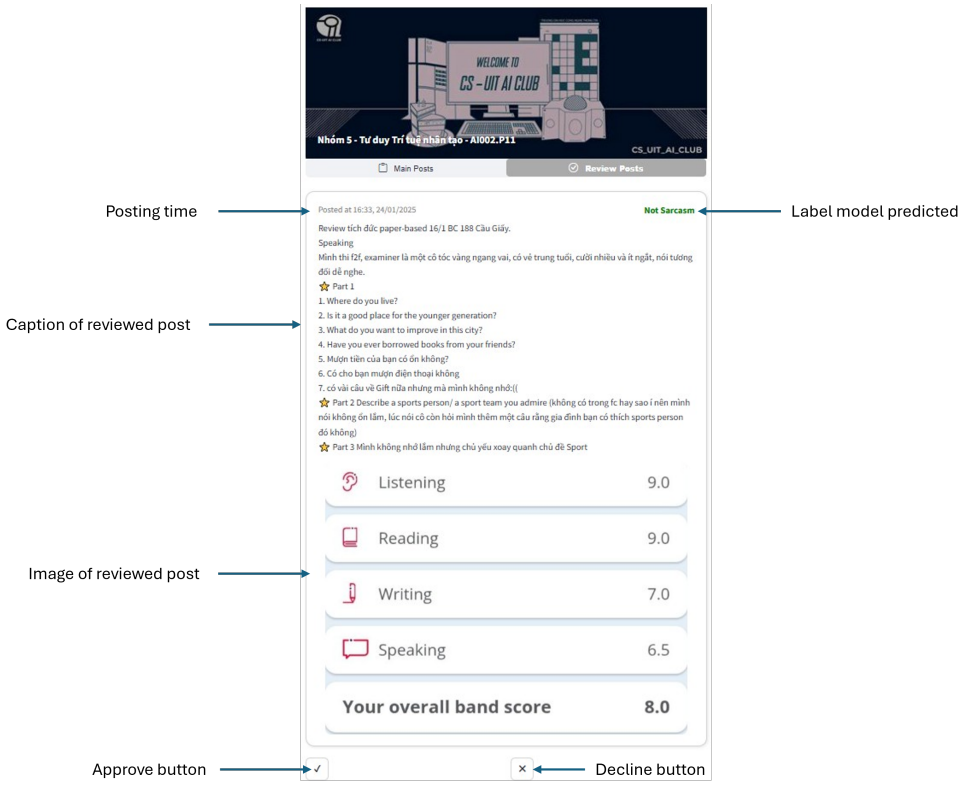
– Post a post: The post consists of only a caption and an image (satisfying the constraints in the section 1.2.)



– Display approved posts in the "Review Posts" tab



- **Review Posts**: All posts must be reviewed through this page. The displayed information includes the caption, image of the post waiting for review, posting time, model classification label, and two buttons to approve or reject the post. We do not perform automatic post approval because we want the model to only support classification; the final step must still be human classification.



You can view the original website code and data here, or directly access the website here. (Since it's hosted on Streamlit Cloud, the app might automatically shut down at times. If this happens, please wait a few minutes for it to restart. Note that performance may be slightly limited.)

4. Ethics and Social

The deployment of a sarcasm detection system involves both ethical considerations and potential social impacts. Addressing these factors ensures responsible implementation while maximizing the system's benefits for online communities.

4.1. Ethical Considerations

4.1.1. Risk of Misclassification

- **Challenge:** The system may incorrectly classify legitimate posts as sarcastic (false positives), leading to unnecessary removal and reducing user trust.
- **Proposed Solutions:**
  - Apply a high confidence threshold for automatic classifications to reduce errors.
  - Temporarily store flagged posts for manual review and restoration if necessary.
  - Establish a user appeal process to handle wrongful deletions and restore posts if appeals are valid.
  - Regularly refine the model using real-world feedback and misclassification cases.

4.1.2. Privacy Concerns in Moderation

- **Challenge:** The analysis of images and text may inadvertently expose personal or sensitive information, raising privacy concerns.
- **Proposed Solutions:**

  - Comply with privacy regulations (e.g., GDPR) to handle user data securely and transparently.
  - Discard original content after processing, retaining only classification results to minimize data retention risks.
  - Anonymize personal information (e.g., names, faces) during analysis.
  - Conduct routine audits to identify and mitigate vulnerabilities in data handling practices.

4.1.3. Misuse of the System

- **Challenge:** In groups centered around specific themes, such as memes or humor, over-reliance on sarcasm detection may suppress legitimate discussions or humor, contradicting the group's purpose.
- **Proposed Solutions:**
  - Customize detection thresholds and moderation rules to fit the context and culture of specific groups.
  - Allow group administrators to adjust or override automatic classifications based on community needs.
  - Educate users about the system's limitations to manage expectations and reduce potential frustration.

4.2. Social Impact

The development and deployment of an AI-powered sarcasm detection system can have significant societal implications, both positive and negative, depending on its implementation and usage.

4.2.1. Positive Impacts

- Improved Communication: The system can help reduce misunderstandings by accurately identifying sarcasm, especially in professional, academic, or multicultural discussions, fostering clearer and more constructive interactions.
- Efficient Content Moderation: Sarcasm detection can assist moderators by identifying potentially harmful or disruptive sarcasm, reducing their workload and allowing them to focus on more complex issues.
- Protection of Vulnerable Users: By detecting harmful or abusive sarcasm, the system can protect vulnerable users from harassment or cyberbullying, creating a safer online environment for everyone.
- Early Detection of Harmful Trends: The system can identify trends in sarcastic or satirical content that might be used to spread misinformation or incite division, allowing for timely intervention.

4.2.2. Negative Impacts

- Over-Censorship: Over-reliance on the system may lead to excessive moderation, stifling legitimate expressions of humor or creativity and discouraging open discussions.
- Cultural and Contextual Misinterpretation: Sarcasm often relies on cultural or contextual cues, which the system might misinterpret, leading to inaccurate moderation decisions.

- Dependence on Automation: Over-dependence on the system for sarcasm detection might reduce the role of human judgment, which is essential for addressing nuanced or ambiguous cases.
- Risk of Misuse: In certain contexts, the system could be exploited to suppress dissenting opinions or restrict freedom of speech, particularly in environments with limited transparency.

5. Conclusion

5.1. Summary

In this study, we proposed a multimodal approach for detecting sarcasm in Vietnamese social media posts by integrating both image and text data. This approach aimed to leverage visual and textual cues to accurately identify sarcastic intent. Real-world applications of this system, such as automated content moderation or sentiment analysis on Vietnamese social media platforms, would require overcoming challenges like low image quality, inconsistent resolution, and language variability, including the use of slang and non-standard language. Additionally, addressing complexities in text recognition within images and refining the system to handle diverse cultural contexts are crucial steps to enhance its practical usability and reliability.

5.2. Future Directions

While the current study lays a strong foundation for sarcasm detection, several areas remain open for exploration and improvement:

- **Advanced Feature Interaction:** Future work could explore the use of advanced techniques such as cross-attention mechanisms or multi-head cross-attention models to better capture the relationships between image features and textual features. These methods could enhance the system's ability to understand and integrate multimodal information, leading to more accurate sarcasm detection.
- **Improved Text Recognition:** A significant limitation of the current approach lies in handling text within images. Collecting more diverse datasets or manually annotating text in images could improve the quality of text recognition, particularly for non-standard fonts, handwriting, or low-resolution images. This would allow the system to better analyze the interplay between visual and textual elements.
- **Data Enrichment and Annotation:** Expanding the dataset to include more examples of sarcastic content, particularly in underrepresented contexts, would enable more comprehensive training. Leveraging community-based annotation or expert labeling could also improve the quality and diversity of the training data.
- **Deployment and Real-World Testing:** To evaluate the system's practical utility, real-world deployment should be pursued. This could include developing browser extensions or social media bots for platforms such as Facebook, Discord, or Twitter. These tools could automate content moderation or provide sarcasm-related insights, facilitating broader adoption and feedback for refinement.
- **Real-Time Optimization:** Optimizing the model for real-time performance, especially on devices with limited computational resources, would enhance its usability in various applications. This includes reducing inference time while maintaining accuracy, ensuring the system remains responsive and efficient.
- **User Engagement and Feedback Integration:** Implementing mechanisms for users to provide feedback on system predictions could improve trust and performance over time.

Such interactive features could allow the model to learn dynamically from misclassifications and adapt to evolving user needs.

These directions aim to enhance the robustness, adaptability, and practicality of sarcasm detection systems. By addressing these challenges and opportunities, future research can build on the foundation established in this study, contributing to healthier and more inclusive online environments.

References

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2020. arXiv: 2010.1192 [cs.CV]. URL: https://arxiv.org/abs/2010.11929.

[2] Yuning Du, Chenxia Li, Ruoyu Guo, Xiaoting Yin, Weiwei Liu, Jun Zhou, Yifan Bai, Zilin Yu, Yehua Yang, Qingqing Dang, and Haoshuang Wang. *PP-OCR: A Practical Ultra Lightweight OCR System*. 2020. arXiv: 2009.09941 [cs.CV]. URL: https://arxiv.org/abs/2009.09941.

[3] Xuan Hoang. *Vietnamese Text Normalizer*. https://github.com/Xuanhoang214/VietnameseTextNormalizer. 2021.

[4] JaidedAI. *EasyOCR - Ready-to-use OCR with 80+ supported languages*. 2020. URL: https://github.com/JaidedAI/EasyOCR.

[5] Nam Nguyen, Thang Phan, Duc-Vu Nguyen, and Kiet Nguyen. *ViSoBERT: A Pre-Trained Language Model for Vietnamese Social Media Text Processing*. 2023. URL: https://aclanthology.org/2023.emnlp-main.315/.

[6] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. *How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers*. 2021. arXiv: 2106.10270 [cs.CV]. URL: https://arxiv.org/abs/2106.10270.