:

ANALYSIS AND BUILDING CLASSIFICATION FOR VIBLO POSTS

Pham Quoc Thai (20235558) Nguyen Khac Viet Anh (20235471)
Nguyen Duc Manh (20235525) Tran Quang Trong (20235565)
Hoang Van Nhan (20235542) Le Nguyen Phuoc Thanh (20235560)
Nguyen Ngoc Linh (20235520)

January 2, 2026

Abstract

This report presents a comprehensive analysis of 4000 technical articles collected from the Viblo platform. Our objective is to analyze the data, preprocess and build a classification model for the data.

Contents

1 Introduction

1.1 Background

Viblo is a prominent knowledge-sharing platform for the Vietnamese IT community. Understanding the trends, writing styles, and topic relationships on this platform provides valuable insights into the technological landscape in Vietnam.

1.2 Data Source

The dataset used in this analysis is `viblo_posts_with_tags.csv`, initially containing **6,500 rows**. The key attributes include:

- **Title:** The headline of the article.
- **Content:** The full body text, often containing Markdown formatting, code blocks, and Vietnamese natural language.
- **Tags:** A raw string of user-defined tags (e.g., "Machine Learning|Python|Tips").

2 Data Cleaning and Preprocessing

The raw data contained noise, inconsistent tagging, and non-textual elements (code snippets). A multi-stage cleaning pipeline was implemented.

2.1 Tag Normalization and Filtering

User-generated tags are often inconsistent (e.g., "ReactJS", "react.js", "React"). To standardize the dataset, we applied a mapping strategy to merge variations into canonical forms.

Step 1: Tag Mapping We defined a dictionary to map variations to a standard format (e.g., 'machine learning', 'deep learning' $\rightarrow$ 'ai').

Step 2: Filtering Top Topics To focus the analysis on significant trends, only articles containing at least top 20 most frequent tags were retained. This reduced the dataset from 6,500 to **4,093** relevant articles.

Since the tags on the forum were community-made, we need to map tags that have similar meanings. The mapping is shown in Table 1.

Table 1: Mapping các tag gốc sang 20 tag phổ biến nhất

Original Tag	**Mapped Tag**
reactjs	react
react.js	react
vuejs	vue
vue.js	vue
angularjs	angular
node.js	nodejs
artificial intelligence	ai
machine learning	ai
ml	ai
deep learning	ai
aitechnology	ai
agenticai	ai
golang	go
c++	cpp
c/cpp	cpp
amazon web services (aws)	aws
amazon web services	aws
aws resource explorer	aws
web development	webdev
webdevelopment	webdev
webapp	webdev
coding	programming
codingdev	programming
programing	programming
js	javascript
ts	typescript
dot net	dotnet
.net	dotnet
c#	csharp
android programming	programming
businessanalyst	ba
hỗ trợ phát triển ai	ai
bad programming	programming
dotnet	webdev
software developmen	software development
stocktradingsoftware	trading
paxos gold	trading
llm	ai

2.2 Text Processing (NLP)

The `content` column required extensive cleaning to separate natural language from technical artifacts.

1. **Lowercase:** Lowercase the text
2. **Removal of Non-Language Elements:** Using Regular Expressions (Regex) to remove Markdown code blocks (‘‘‘ . . .‘‘‘) and links, as these distort word frequency analysis.
3. **Tokenization:** utilized the `pyvi` library for Vietnamese word segmentation (e.g., "học máy" is treated as a single token "học_máy" rather than "học" and "máy").
4. **Stopwords Removal:** A standard Vietnamese stopwords and English list was augmented with custom "trash words" specific to blog writing (e.g., "mình", "bạn", "hello", "demo", "part").

3 Exploratory Data Analysis (EDA)

3.1 Descriptive Statistics

After cleaning, we calculated key metrics for the refined dataset:

- **Total Articles:** 4,093
- **Word Count Distribution:** The length of articles varies significantly, with a right-skewed distribution indicating most articles are concise, while a few are extremely detailed.

3.2 Tags

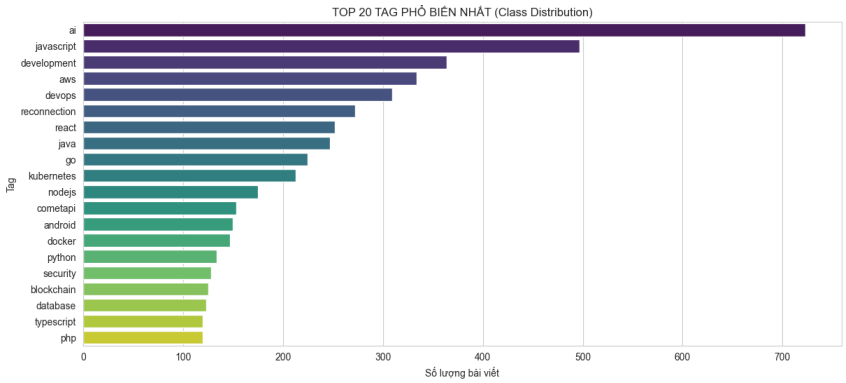

Figure 1: Tag distribution

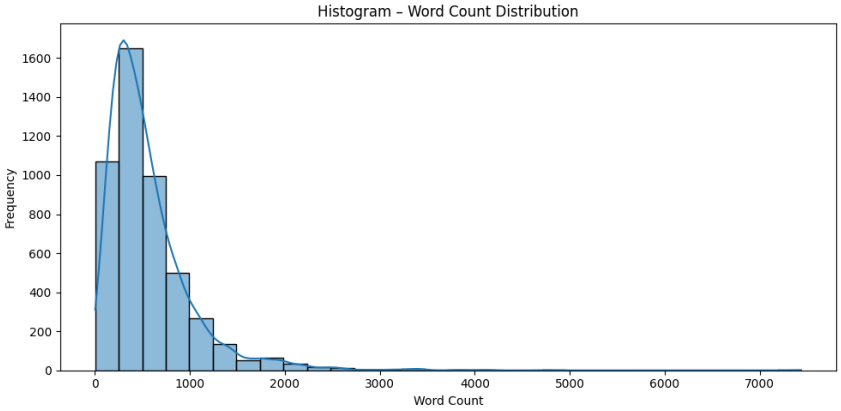

Figure 2: Number of tags each post

Most of posts only have 1 tag only, with a few exceptions of having 2 and 3 tags. In 20 tags, tag "ai" appears the most.

3.3 Lexical Diversity Analysis

We analyzed **Lexical Diversity** (Unique Words / Total Words).

- **Observation:** There is an inverse relationship between Word Count and Lexical Diversity. Shorter posts tend to have higher diversity, while longer tutorials repeat technical terms, lowering the ratio.

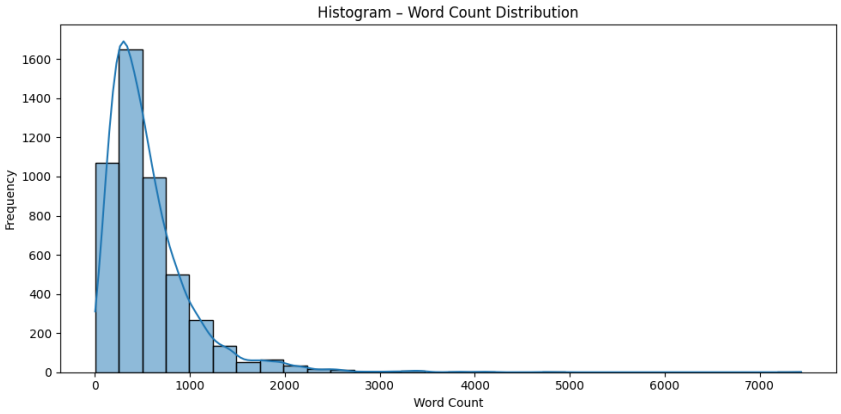

Figure 3: Length distribution

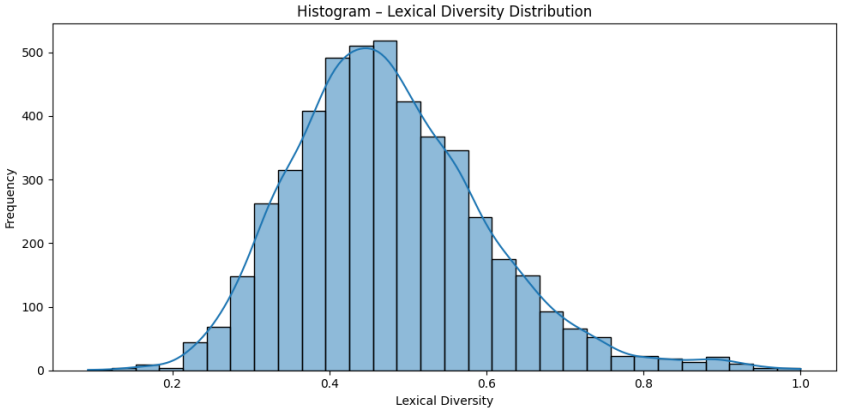

Figure 4: Lexical Diversity Distribution

3.4 N-gram Analysis

Table 2 presents the top 20 most frequent bigrams found in the dataset.

Rank	Bigram (Original)	Count
1	cơ_sở dữ_liệu (database)	1845
2	tác vụ (task)	1577
3	truy vấn (query)	1490
4	node js	1375
5	trải nghiệm (experience)	1115
6	kiểu dữ_liệu (data type)	1003
7	tự_động hóa (automation)	884
8	câu trả_lời (answer)	800
9	ngôn_ngữ lập_trình (programming language)	684
10	tập dữ_liệu (dataset)	672
11	such as	665
12	align center	645
13	div align	637
14	ứng_dụng web (web application)	591
15	đám mây (cloud)	591
16	ci cd	590
17	độ chính_xác (accuracy)	578
18	can be	563
19	xây_dựng ứng_dụng (build application)	556
20	mô_hình ngôn_ngữ (language model)	534

Table 2: Top 20 Most Frequent Bigrams in Viblo Articles

Analysis of Findings:

- **Dominance of Fundamental Concepts:** The highest-ranking bigram, *"cơ_sở*

dữ_ liệu" (database), appearing 1,845 times, alongside *"truy vấn"* (query) and *"kiểu dữ_ liệu"* (data type), indicates that data management and manipulation are central themes in the community's discourse.

- **Software Development Lifecycle (SDLC):** Terms such as *"tự_ động hóa"* (automation) and *"ci cd"* (Continuous Integration/Continuous Deployment) highlight a strong focus on modern DevOps practices and workflow optimization.
- **Web and Backend Technologies:** Specific technologies like *"node js"* and general concepts like *"ứng_ dụng web"* (web application) and *"đám mây"* (cloud) reflect the platform's strong bias towards web development and cloud computing.
- **AI and Data Science Trends:** The presence of *"tập dữ_ liệu"* (dataset), *"độ chính_ xác"* (accuracy), and *"mô_ hình ngôn_ ngữ"* (language model) confirms the growing interest in Artificial Intelligence and Machine Learning topics.
- **Formatting Artifacts:** Interestingly, terms like *"div align"* and *"align center"* appear in the top list. These are likely remnants of HTML code snippets embedded within tutorials that were not fully removed during the cleaning process, suggesting that many articles contain direct code examples.

3.5 Frequently appeared words in each tag

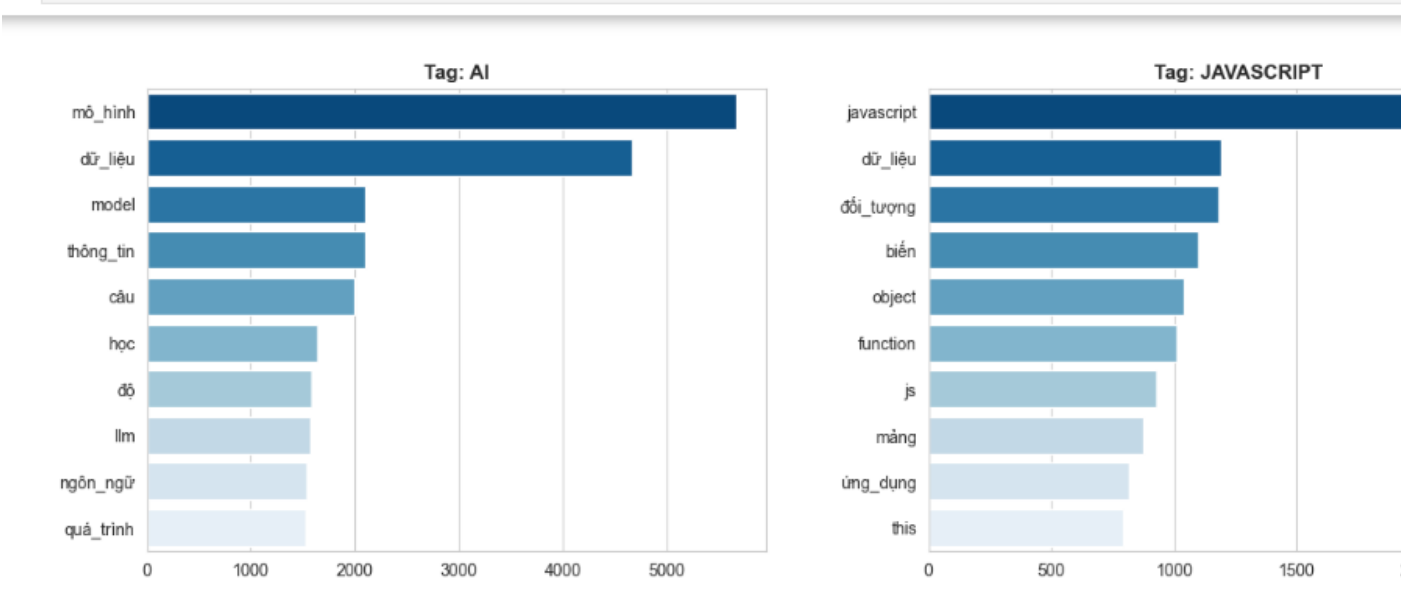

Figure 5

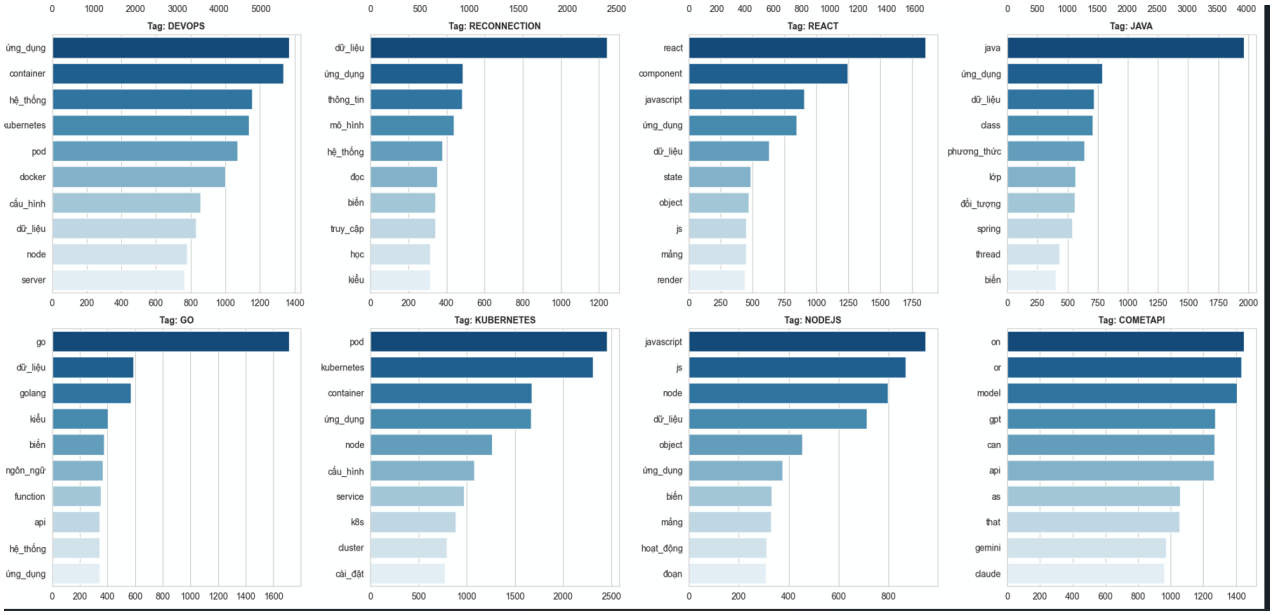

Figure 6

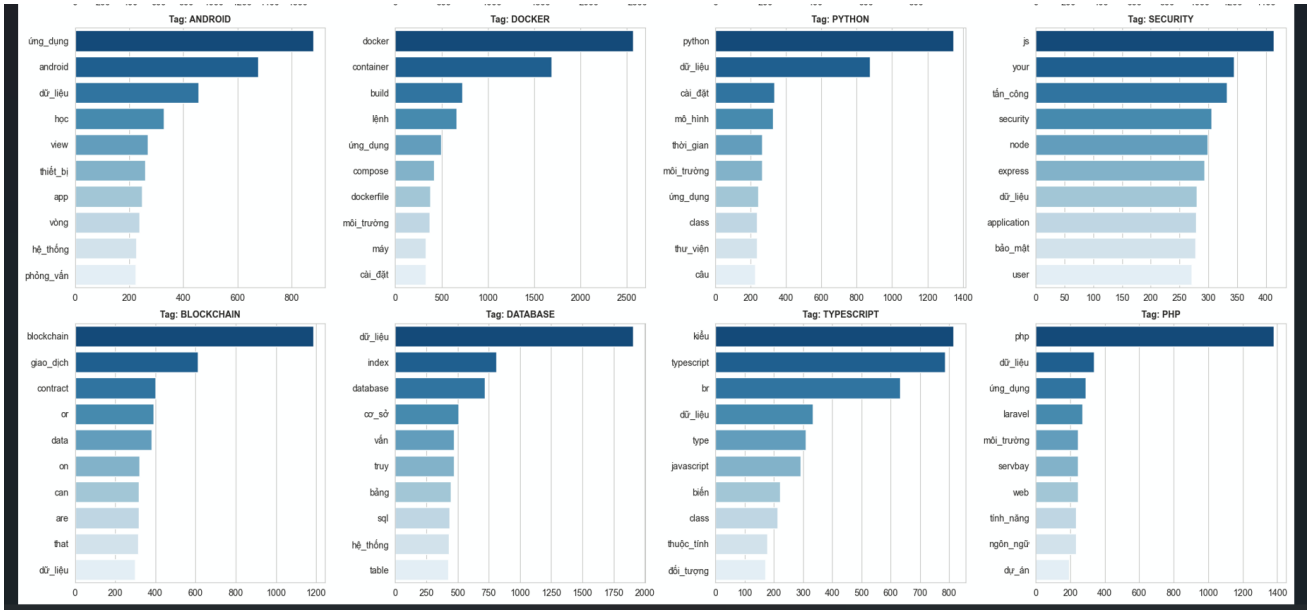

Figure 7

Key Findings:

- **Domain Consistency and Vocabulary Specialization:** The analysis reveals a high degree of semantic separation between technical domains, confirming that the tagging system on Viblo is accurate and reflective of the content. We observed distinct vocabulary patterns across three main clusters:

    1. **Frontend and Web Development (React, Vue, JavaScript):** The vocabulary in this cluster is highly focused on user interface construction and state management.

English technical terms such as *component*, *function*, *props*, and *state* are dominant. These are frequently combined with Vietnamese action verbs like *xử_ lý* (handle/process) and *hiển_ thị* (display), or nouns like *giao_ diện* (interface). This indicates a strong focus on the implementation details of web applications.

2. **Infrastructure and DevOps (Docker, Kubernetes, DevOps):** The linguistic footprint here shifts entirely towards deployment and system architecture. Keywords such as *image*, *container*, *command*, and *server* appear with high frequency. Vietnamese terms in this cluster are notably system-oriented, such as *hệ_ thống* (system), *cài_ đặt* (install), and *môi_ trường* (environment). The absence of "UI/UX" vocabulary in this cluster further validates the clear separation from the Frontend group.
3. **Data Science and AI (Machine Learning, AI, Deep Learning):** This domain exhibits the most distinct vocabulary, characterized by mathematical and analytical terminology. Top words include *model*, *training*, *predict*, and *layer*. Crucially, the Vietnamese vocabulary here is specific to data manipulation, featuring words like *dữ_ liệu* (data), *mô_ hình* (model), and *huấn_ luyện* (train). The term *bài_ toán* (problem/task) also appears frequently, reflecting the problem-solving nature of AI articles.

4 Visualizations and Insights

4.1 Tag Frequency and Co-occurrence

This figure (derived from the heatmap analysis) illustrates how technologies appear together.

- **Strong Correlations:**
  - `DevOps` strongly co-occurs with `Docker`, `Kubernetes`, and `AWS`.
  - `Frontend` technologies like `Vue` and `React` often appear with `JavaScript`.
- **Independence:** `Blockchain` and `AI` tend to form isolated clusters, appearing less frequently with general web development tags.

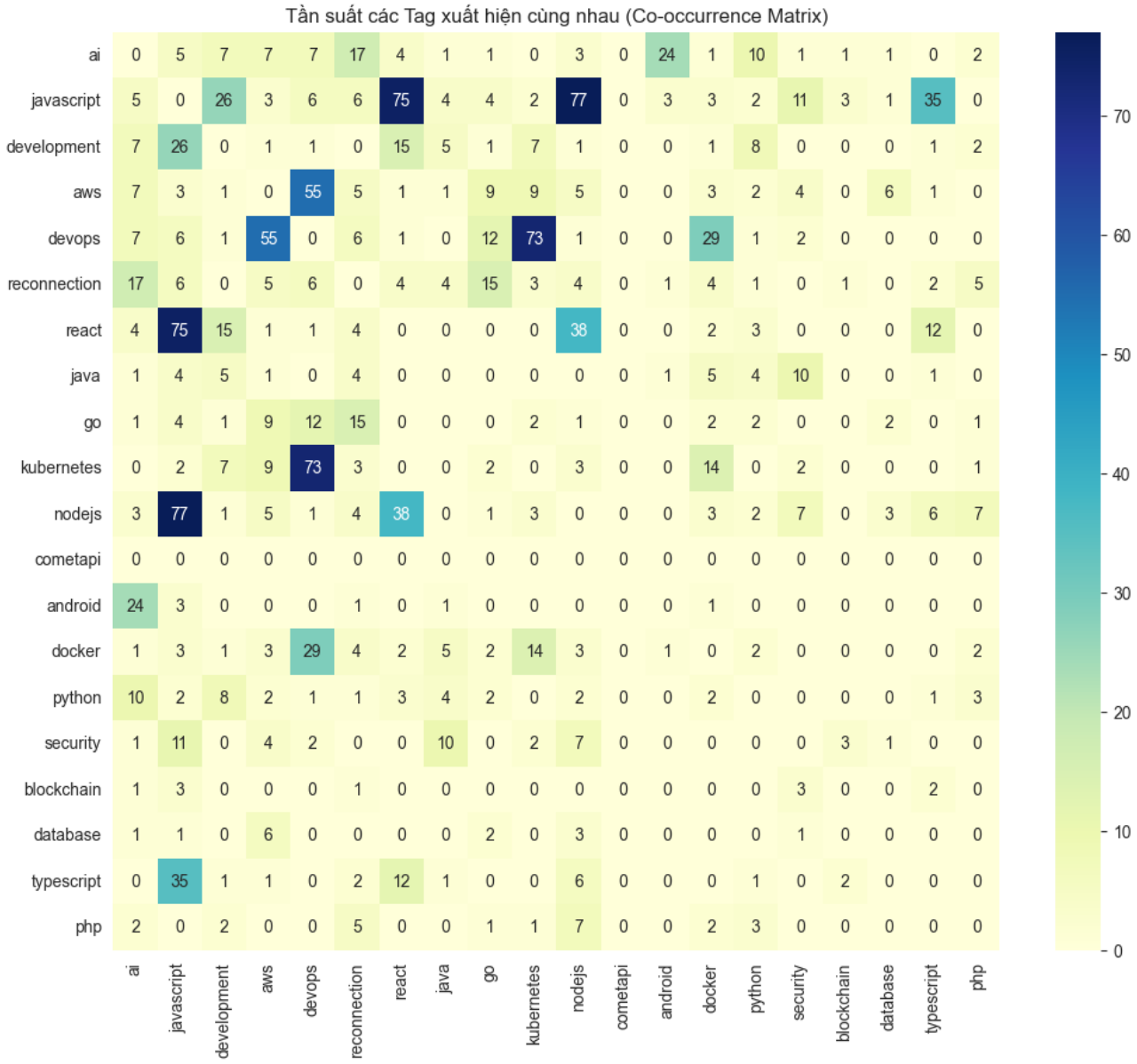

Figure 8: Heat Map

4.2 Semantic Clustering with t-SNE

We employed t-SNE (t-Distributed Stochastic Neighbor Embedding) to project the high-dimensional TF-IDF vectors (1,000 features) into 2D space.

Interpretation of the t-SNE Plot:

1. **Distinct Clusters:** Articles tagged with "Blockchain", "AI", and "DevOps" form clear, separated islands. This indicates their vocabulary is highly specialized and distinct from general programming.
2. **Overlap Area:** "JavaScript", "React", "Vue", and "WebDev" show significant overlap, reflecting the shared vocabulary of the Frontend ecosystem.

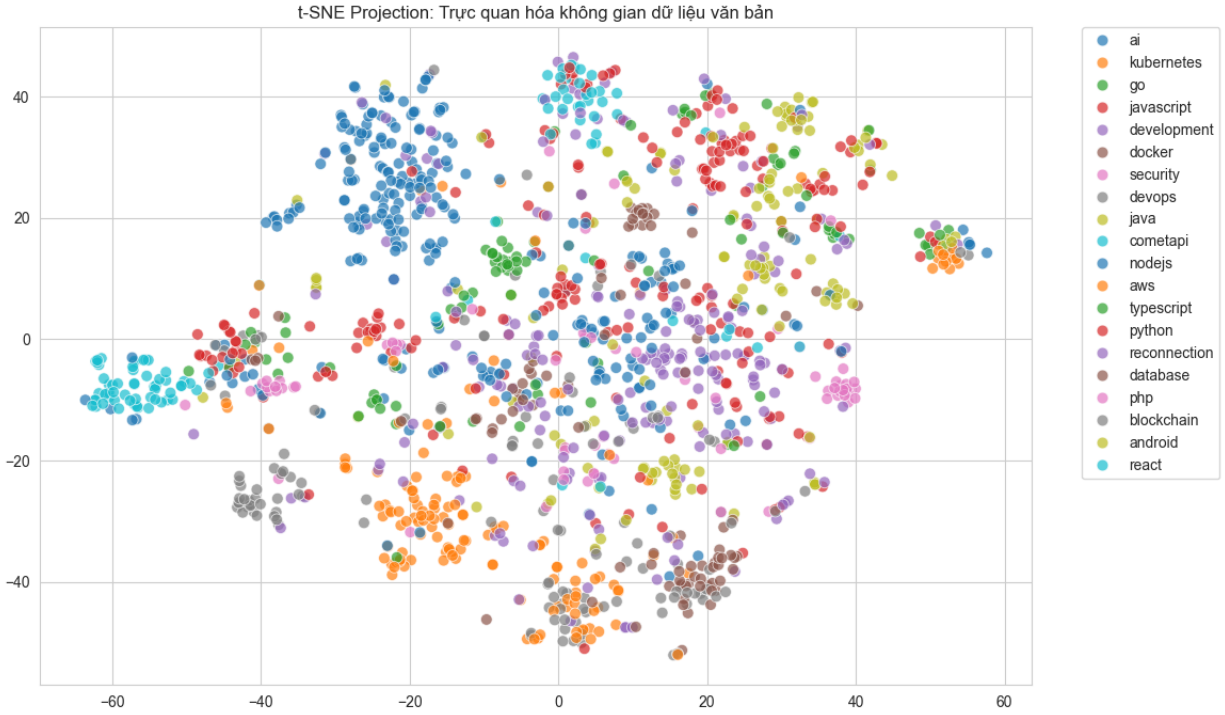

Figure 9: Heat Map

5 Model Methodology

In this section, we present our approach to the multi-label text classification task. We first define the challenges associated with the dataset, specifically the high cardinality and semantic overlap of tags.

5.1 Motivation for Soft Clustering

A significant challenge in categorizing technical content is the *vocabulary mismatch problem* and the *long-tail distribution* of labels. In our dataset, distinct tags often represent semantically identical or highly correlated concepts (e.g., “AI” vs. “Machine Learning,” or “React” vs. “Frontend”). And we notice that since we only keep 20 most popular tags of the data we crawled, some of the tags that are removed have similar meaning to the tags we keep, for example: "deep learning" and "ai" or "web" and "JavaScript".

Standard multi-label classification models typically utilize a “Hard Metric” (Exact Match), where a prediction is penalized if it does not strictly match the ground truth string. This leads to two critical issues:

1. **Information Loss:** To make discriminative models feasible, rare tags are often truncated (e.g., keeping only the Top 20). This ignores fine-grained information.
2. **False Negatives in Evaluation:** A model predicting “programming” for a document

labeled “java” is penalized as incorrect, despite capturing the correct semantic domain.

To mitigate this, we introduce a **Soft Classification** mechanism. Instead of treating tags as independent orthogonal labels, we model them as vectors in a continuous semantic space. Correctness is thus defined not by exact string matching, but by *cluster membership*, allowing the model to be evaluated on its ability to identify the correct semantic topic rather than just the specific keyword.

5.1.1 Feature Extraction

The raw text input is transformed into numerical vectors using Term Frequency-Inverse Document Frequency (TF-IDF). To handle the sparsity of the TF-IDF matrix and reduce computational complexity, we apply Truncated Singular Value Decomposition (SVD), also known as Latent Semantic Analysis (LSA).

$$\mathbf{X}_{dense} = \text{SVD}_k(\text{TF-IDF}(\text{Text})) \tag{1}$$

where $k = 500$ represents the number of components retained.

5.2 Method I: XGBoost with Soft Semantic Evaluation

Our initial approach treats the problem as a series of independent binary classification tasks using the **One-vs-Rest** strategy. However, standard exact-match metrics often fail to capture semantic proximity (e.g., predicting ’python’ when the label is ’ai’). To address this, we implemented a **Soft Clustering Metric** based on vector similarity.

5.2.1 Soft Clustering Similarity Matrix Analysis

The core of our soft evaluation lies in constructing a "Semantic Map" of tags before training the classifier. The process involves three key steps as implemented in our code:

1. **Tag Centroid Calculation:** We first convert all documents into TF-IDF vectors. For every unique tag in the dataset, we calculate its *Centroid* — the mean vector of all documents containing that tag. This effectively places every tag into the same vector space as the documents.
2. **Cosine Similarity Matrix:** We compute the cosine similarity between the centroids of 20 tags . This results in a similarity map (e.g., Sim(’ai’, ’python’) = 0.64).
3. **Soft Precision Scoring:** During evaluation, a predicted tag p is considered a "match" for a ground truth tag g if:

$$p = g \quad \vee \quad \text{CosineSimilarity}(Centroid_p, Centroid_g) \geq \tau \tag{2}$$

where τ is a threshold (set to 0.7 in our experiments).

5.2.2 Experimental Results

The model was trained using `XGBClassifier` (OneVsRest) on TF-IDF vectors reduced by TruncatedSVD (200 components).

Table 3: XGBoost Performance: Hard vs. Soft Metrics

Metric	Score
Hard Accuracy (Train)	0.8
Hard Accuracy (Test)	0.5008
Soft Precision (Train)	0.82
Soft Precision (Test)	0.6206

Conclusion: The jump from 50.08% (Hard Accuracy) to 62.06% (Soft Precision) confirms that in roughly 12% of test cases, the XGBoost model predicted tags that were technically "incorrect" but semantically valid (e.g., related technologies within the same domain).

5.3 Soft Clustering via K-mean

5.3.1 Tag Embedding via Centroids

Instead of learning a decision boundary, we learn the "center of gravity" for each tag in the semantic space. Let $\mathbf{v}_d$ be the vector representation of document d (obtained via TF-IDF + SVD). The embedding for a tag t, denoted as $\mathbf{c}_t$, is calculated as the centroid of all documents labeled with t:

$$\mathbf{c}_t = \frac{1}{|D_t|} \sum_{d \in D_t} \mathbf{v}_d \tag{3}$$

where D_t is the set of documents containing tag t. This results in a vector representation where semantically similar tags (e.g., "CNN" and "RNN") are located near each other in the vector space.

5.3.2 Soft Clustering for Evaluation

To formalize the semantic relationships, we apply the **K-Means Clustering** algorithm on the derived tag centroids $\{\mathbf{c}_t\}$. The tags are partitioned into K clusters $C_1, C_2, \ldots, C_K$ such that tags within the same cluster share semantic context.

$$\text{Cluster}(t) = \arg\min_j \|\mathbf{c}_t - \mu_j\|^2 \tag{4}$$

where μ_j is the mean of cluster j.

5.3.3 Inference and Soft Metric

For a new test document vector $\mathbf{v}_{test}$, we compute the Cosine Similarity against all tag centroids and select the tags with the highest similarity scores.

$$\hat{y} = \arg\max_{t} \left(\frac{\mathbf{v}_{test} \cdot \mathbf{c}_t}{\|\mathbf{v}_{test}\| \|\mathbf{c}_t\|} \right) \tag{5}$$

Finally, the model is evaluated using a **Cluster-based Soft Accuracy**. A prediction $\hat{t}$ is considered correct if its cluster assignment matches the cluster of any ground truth tag t_{true}:

$$\text{Score} = \mathbb{I}(\text{Cluster}(\hat{t}) \in \{\text{Cluster}(t) \mid t \in Y_{true}\}) \tag{6}$$

This metric acknowledges that predicting a synonym or a sub-concept within the same semantic domain constitutes a successful classification.

5.3.4 Experimental Results

5.3.5 Experimental Results

We evaluate our non-parametric, centroid-based soft clustering method on the test set. The metric, **Cluster-based Soft Accuracy**, considers a prediction correct if the top predicted tag belongs to the same semantic cluster as any ground truth tag. This allows semantically related tags (e.g., synonyms or subtopics) to be counted as correct.

- Total test documents: 655
- Cluster-based Soft Accuracy: 0.8595

This indicates that **85.95%** of test documents have at least one predicted tag falling in the same semantic group as one of the ground truth tags.

Example Predictions:

Document	True Tags (Clusters) / Predicted Tags (Top 1 Cluster) / Result
0	True: ['development', 'javascript'] (Clusters: [1,1]) Predicted: ['java', 'development', 'reconnection'] (Top 1 Cluster: 1) Result: Correct Cluster
1	True: ['ai'] (Clusters: [1]) Predicted: ['ai', 'reconnection', 'development'] (Top 1 Cluster: 1) Result: Correct Cluster
2	True: ['security'] (Clusters: [1]) Predicted: ['security', 'nodejs', 'javascript'] (Top 1 Cluster: 1) Result: Correct Cluster
3	True: ['aws'] (Clusters: [6]) Predicted: ['typescript', 'aws', 'reconnection'] (Top 1 Cluster: 7) Result: Incorrect Cluster
4	True: ['javascript', 'nodejs', 'typescript'] (Clusters: [1,1,7]) Predicted: ['nodejs', 'javascript', 'security'] (Top 1 Cluster: 1) Result: Correct Cluster

6 Conclusion and Future Work

6.1 Conclusion

In this study, we proposed a centroid-based soft clustering approach for multi-label tag classification. Unlike traditional discriminative models such as XGBoost, which rely on exact label matches, our method leverages the semantic proximity of tags in a reduced-dimensional vector space. Each tag is represented by the centroid of all documents containing it, and K-Means clustering groups tags with similar semantics.

Experimental results on the dataset demonstrate that the soft cluster-based metric can achieve high semantic accuracy highlighting that even when the exact tag is not predicted, the model often identifies a semantically related tag within the same cluster. This approach provides a more flexible and realistic evaluation of tag prediction pe

6.2 Future Work

While the centroid-based soft clustering approach demonstrates promising semantic capture, there remain several directions for improvement:

1. **Dynamic Clustering:** Currently, the number of clusters is fixed. Future work could explore adaptive clustering methods (e.g., DBSCAN, hierarchical clustering) to automatically determine the optimal number of semantic groups.

2. **Contextual Embeddings:** Instead of TF-IDF + SVD, leveraging pre-trained contextual embeddings (such as BERT, RoBERTa, or ViBERTa for Vietnamese) could capture richer semantic relationships among tags.

3. **Soft Prediction Thresholds:** Extend the soft metric to consider multiple top-k predictions with weighted contributions, allowing a more graded measure of semantic correctness.

4. **Cross-Domain Generalization:** Test the approach on other technical domains or multilingual datasets to evaluate the robustness of tag centroid and cluster representations.

5. **Hybrid Models:** Combine centroid-based soft clustering with discriminative models (e.g., XGBoost, neural networks) to leverage both geometry-based and supervised learning advantages.

6.3 References

References

[1] G. Salton and C. Buckley, *Term-weighting approaches in automatic text retrieval*, Information Processing & Management, 1988.

[2] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, *Indexing by Latent Semantic Analysis*, Journal of the American Society for Information Science, 1990.

[3] T. Chen and C. Guestrin, *XGBoost: A Scalable Tree Boosting System*, Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.

[4] J. MacQueen, *Some Methods for Classification and Analysis of Multivariate Observations*, Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1967.

[5] J. Devlin, M. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, arXiv preprint arXiv:1810.04805, 2018.