GROUP 3

Intelligent Tagging and Recommendation System for StackOverflow Posts

APAN 5430: Applied Text & Natural Language Analytics Term Project

Group Members: Sixuan Li, Wenyang Cao, Haoran Yang, Wenling Zhou, Jake Xiao

Github Repo:

https://github.com/educated-fool/stack-overflow-intelligent-tagging

Background & Problem Definition

1

Background:

- ***Stack Overflow*** is a major ***Q&A platform*** for programmers.
- Over ***18 million questions*** covering diverse topics.
- Efficient ***tagging*** is crucial for content organization and discovery.
- ***Manual tagging*** is inconsistent and time-consuming.

2

Problem Definition:

- ***Automate*** the ***tagging process*** for Stack Overflow posts.
- Enhance post ***discoverability*** with accurate ***tag prediction***.
- Improve ***user experience*** with relevant ***tag suggestions*** and ***similar posts.***

3

Challenges:

- Handling ***diverse*** and ***large volumes*** of posts.
- Addressing ***imbalanced tag datasets***.
- Ensuring ***high accuracy*** in tag prediction.
- Efficiently ***processing and vectorizing*** vast text data.

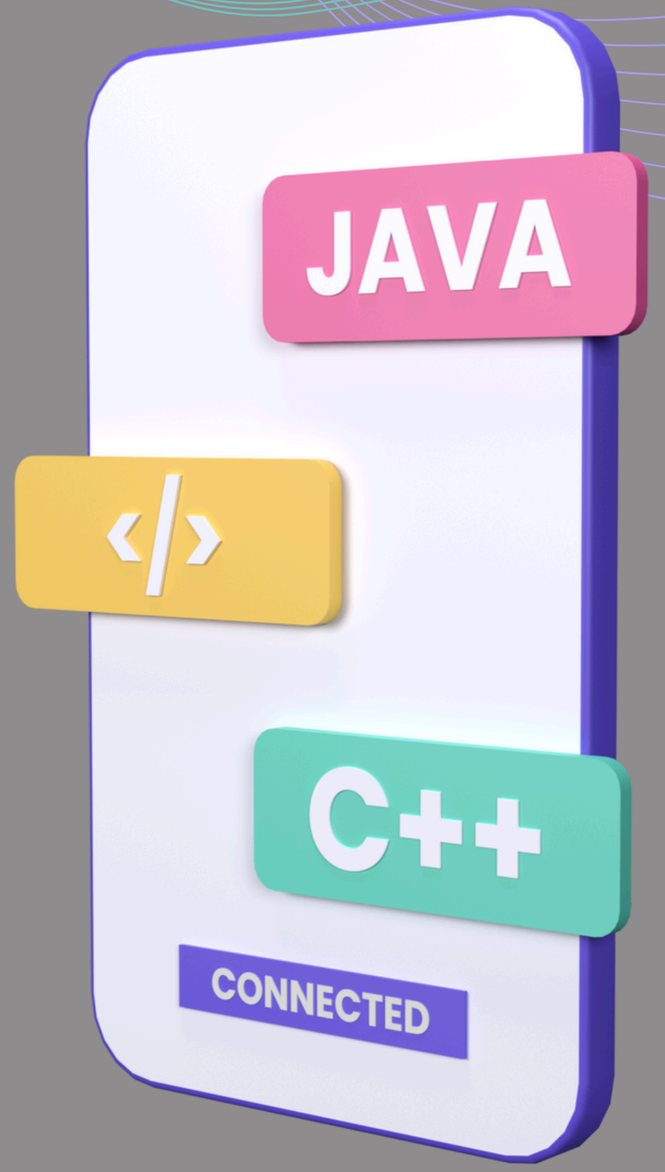
4

Project Value:

- ***Automate*** and ***improve*** the accuracy of tagging.
- Enhance ***content discoverability*** and ***user experience***.
- Provide a ***scalable solution*** for other platforms.
- ***Reduce*** user and moderator workload.

Data Source Details

- ***StackSample: 10% of Stack Overflow Q&A***
- Dataset with the text of ***10%*** of questions and answers from ***Stack Overflow***.
- Organized into ***three*** tables:
  - ***Questions:*** Includes title, body, creation date, closed date, score, and owner ID for questions.
  - ***Answers:*** Includes body, creation date, score, and owner ID for answers, linked to questions via ParentId.
  - ***Tags:*** Includes tags for each question.
- Contains ***1.26*** million questions ***from August 2008 to October 2016***.
- Over ***37,000*** unique tags.
- Total size: ***3.6 GB***.



Design Structure

- ***Data Ingestion and Preprocessing:***
  - ***Collect*** and ***clean*** text data from Kaggle's Stack Overflow dataset.
  - ***Tokenize***, ***lemmatize***, and ***vectorize*** text using ***NLTK***, ***Spacy***, and ***TF-IDF***.
- ***Feature Engineering:***
  - Use advanced embeddings like ***BERT*** for semantic representation.
  - Implement ***TF-IDF vectorization*** for baseline models.
- ***Model Development:***
  - Train classifiers for ***tag prediction***.
  - Compute ***cosine similarity*** to recommend similar posts.
  - Apply ***LDA*** for topic modeling to suggest additional tags.
- ***Deployment and User Interface:***
  - Develop ***Streamlit*** and build a ***web interface*** for user interaction.
  - ***Visualize*** topic clusters and suggested tags.

1

Tag Prediction Model

- Train classifiers (***Logistic Regression***, ***Random Forest***, ***SVM***) using ***TF-IDF*** and ***BERT*** embeddings.
- Evaluate models using ***accuracy***, ***precision***, ***recall***, and ***F1-score***.
- Optimize models with ***hyperparameter tuning*** (Grid Search, Random Search).

2

Cosine Similarity for Similar Posts

- Compute ***cosine similarity*** between posts using their ***vector representations***.
- Recommend ***top similar posts*** based on ***similarity scores***.

3

Topic Modeling for Tag Recommendations

- Use ***LDA*** to generate ***topic clusters*** with up to ***10 keywords*** per cluster.
- ***Visualize topic clusters*** using ***pyLDAvis*** for interpretability.
- Apply ***LDA model*** to identify ***topics*** in new posts and suggest ***additional relevant tags***.

Design Choices and Rationale

1. Use the Tag Prediction Model to classify Stack posts into predefined tags based on their content.

- Content-Based Classification
- Accurate tag prediction enhances the search functionality of the platform
- Reducing the variability and subjectivity of manual tagging

2. Apply Cosine Similarity to identify and recommend similar Stack posts based on their vector representations.

- Vector Representations Capture Content Similarity
- Effective Similarity Matching
- Accurate Similarity Measurement

3. Implement Topic Modeling to identify themes in Stack posts and suggest appropriate tags based on these themes.

- LDA Generates Clear Topic Clusters
- Visualization with pyLDAvis enhances interpretability by clarifying topic distribution and relevance
- Enabling better understanding and management of the types of discussions

Evaluation metrics

Quantitative Metric

1. Accuracy:

Measures the proportion of ***correctly predicted tags*** out of all predictions made.

2. Precision:

Calculates the ratio of ***correctly predicted tags*** to the total tags predicted as positive.

3. Recall:

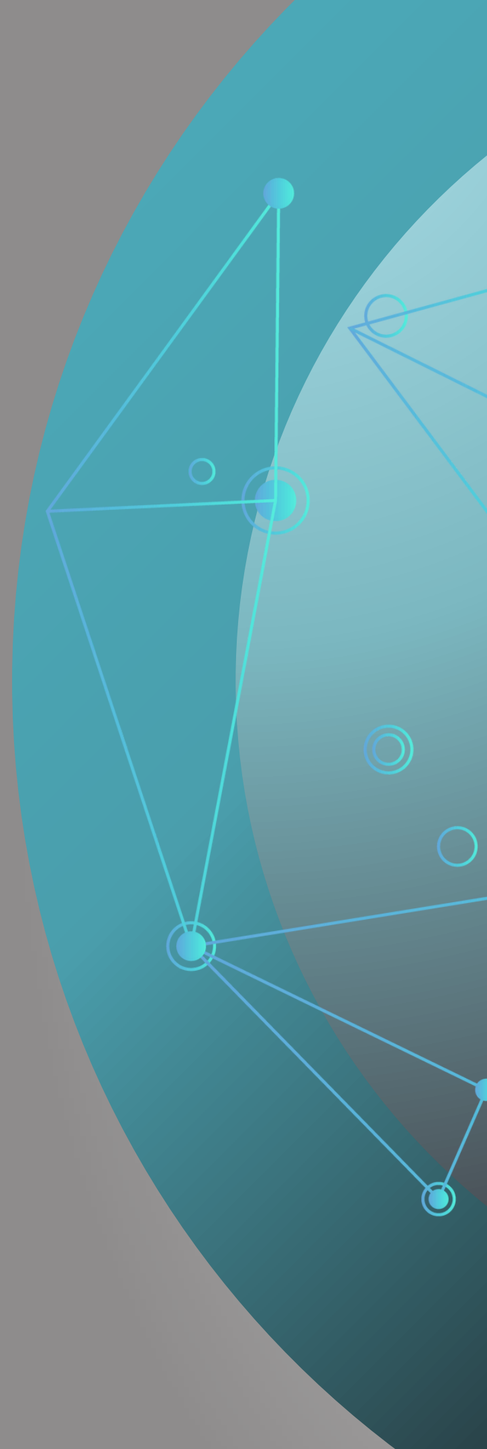
Computes the ratio of ***correctly predicted tags*** to all actual positive tags in the dataset.

4. F1-score:

Provides a ***harmonic mean*** of precision and recall, offering a single metric for model performance.

5. Hamming Loss:

Evaluates the fraction of ***incorrect predictions***, with lower values indicating better performance.



Team Responsibilities

Data Ingestion and Preprocessing • Collect text data from Kaggle's Stack Overflow dataset. • Clean the data by removing HTML tags, special characters, and stop words. • Tokenize, lemmatize, and vectorize the text using NLTK, Spacy, and TF-IDF.	Wenling Zhou
Feature Engineering and Embedding • Implement TF-IDF vectorization for baseline models. • Use advanced embeddings like BERT for semantic representation of text.	Haoran Yang
Model Development • Train classifiers (Logistic Regression, Random Forest, SVM) using TF-IDF and BERT embeddings. • Evaluate models using accuracy, precision, recall, and F1-score. • Optimize models with hyperparameter tuning (Grid Search, Random Search).	Sixuan Li
Similarity Calculation and Topic Modeling • Compute cosine similarity between posts using their vector representations. • Use LDA to generate topic clusters with up to 10 keywords per cluster. • Visualize topic clusters using pyLDAvis. • Apply LDA model to identify topics in new posts and suggest additional relevant tags.	Wenyang Cao
Deployment and User Interface • Build a web interface using Streamlit for user interaction. • Visualize topic clusters and suggested tags.	Jake Xiao

Thank You