Project 2: Code Comment Classification

Using Machine Learning and Deep Learning for Multi-Label Classification of Code Comments

Presented by: Luca Francesco Macera and Calogero Carlino
Dataset: NLBSE’23 Tool Competition Dataset (6,738 samples)
Link: https://nlbse2023.github.io/tools/

What is Code Comment Classification?

Code comments are natural language annotations in software code.

They describe the purpose, functionality, or behaviour of code.

Automatically classifying comments helps in:

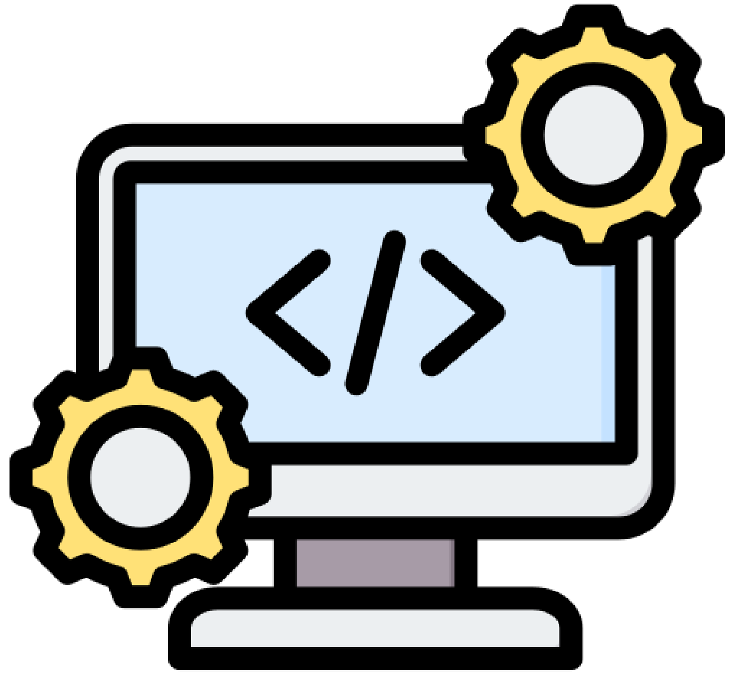
- Improving software documentation.
- Enhancing code readability and maintenance.
- Enabling better developer tools and automated summarization.

Problem

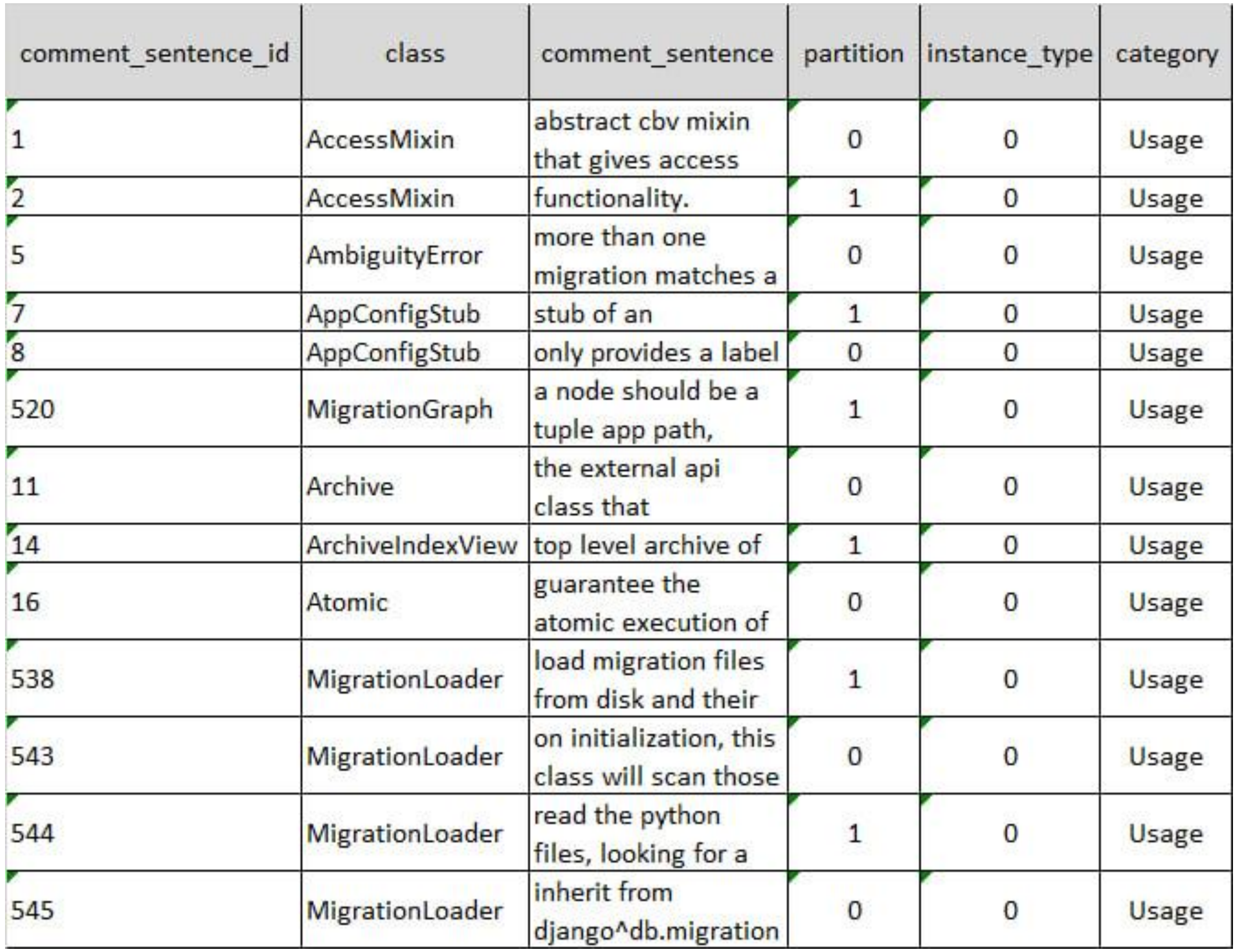
Goal: Develop a model to automatically **classify code comments** into **multiple relevant categories**.

Challenges:

- Comments can belong to more than one class (multi-label classification).
- Comments are short and context-dependent.
- High variability in wording and style between developers.



Dataset Overview



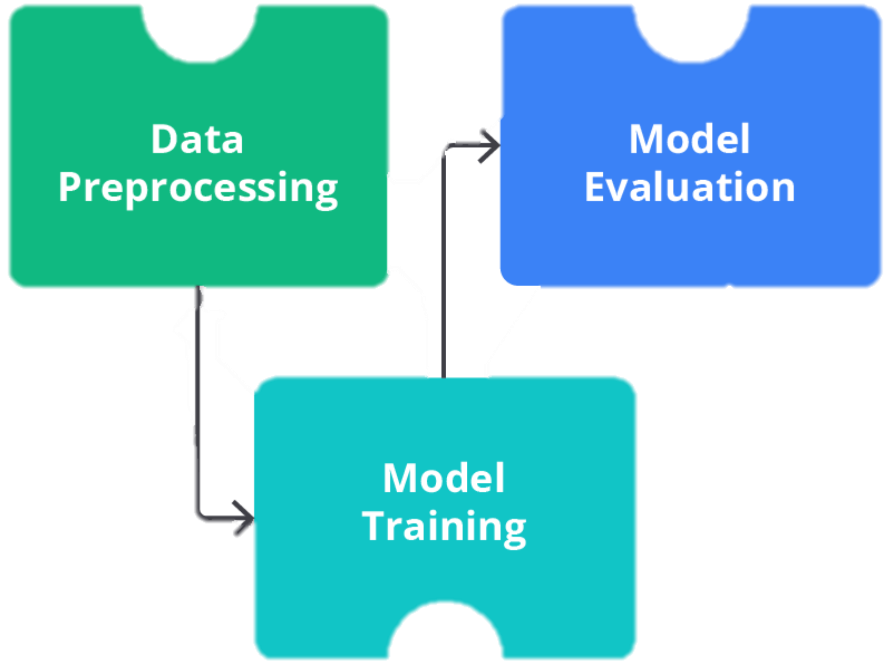
comment_sentence_id	class	comment_sentence	partition	instance_type	category
1	AccessMixin	abstract cbv mixin that gives access	0	0	Usage
2	AccessMixin	functionality.	1	0	Usage
5	AmbiguityError	more than one migration matches a	0	0	Usage
7	AppConfigStub	stub of an	1	0	Usage
8	AppConfigStub	only provides a label	0	0	Usage
520	MigrationGraph	a node should be a tuple app path,	1	0	Usage
11	Archive	the external api class that	0	0	Usage
14	ArchiveIndexView	top level archive of	1	0	Usage
16	Atomic	guarantee the atomic execution of	0	0	Usage
538	MigrationLoader	load migration files from disk and their	1	0	Usage
543	MigrationLoader	on initialization, this class will scan those	0	0	Usage
544	MigrationLoader	read the python files, looking for a	1	0	Usage
545	MigrationLoader	inherit from django^db.migration	0	0	Usage

Source: NLBSE’23 Tool Competition Dataset

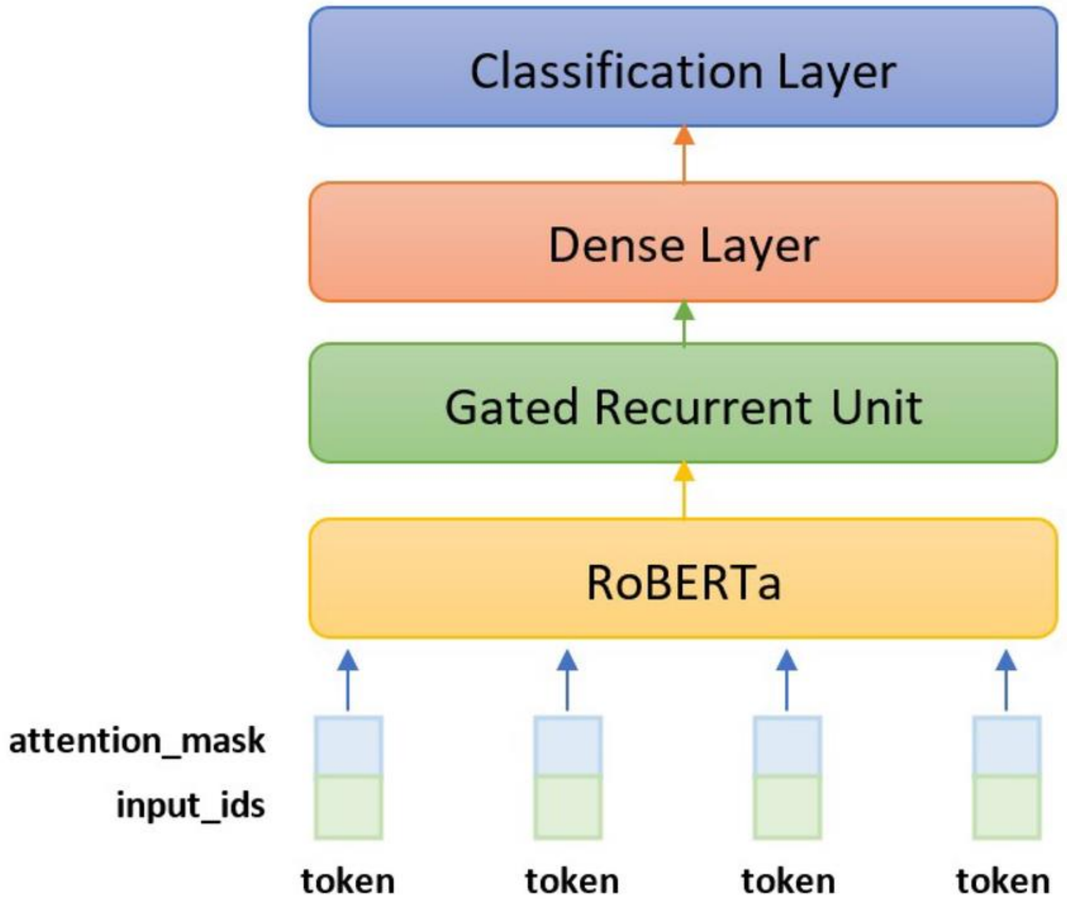
- **Total classes:** 6,738
- **Data type:** Actual sentence string
- **Format:** each row represent a sentence (aka an instance) and each sentence contains multiple labels
- **Task:** Predict comment class

Methodology

1. **Data Preprocessing**
   - Clean and tokenize text
   - Handle class imbalance (if present)
2. **Model Implementation**
   - Apply Machine Learning (e.g., SVM, Random Forest) or Deep Learning (e.g., LSTM, BERT, RoBERTa) classifiers
3. **Evaluation Setup**
   - Use k-fold cross-validation
   - Evaluate using metrics such as F1-score, Precision, Recall



Evaluation and Comparison



- Compare model performance against baseline models (Transformer-based RoBERTa baseline)
- Use multi-label metrics for fair evaluation
- Analyze and interpret results to identify improvement areas

Expected Outcomes

- Most effective ML/DL techniques for comment classification
- Gain insights into linguistic patterns in code comments
- Provide recommendations for better automated documentation tools

Why Code Comment Classification Is Useful

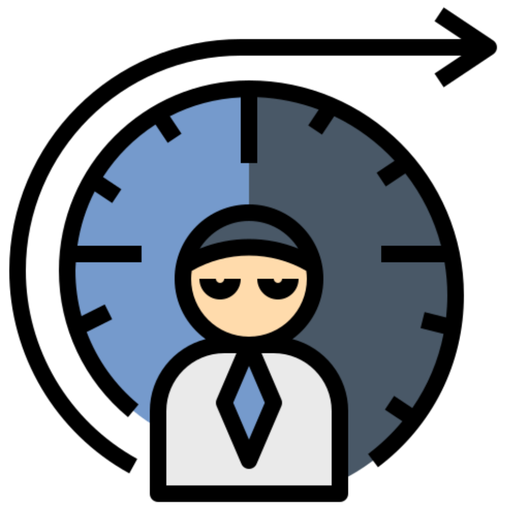
1. **Improves Software Maintenance**
   - Helps developers quickly understand the intent and functionality of code.
   - Facilitates bug fixing, feature updates, and onboarding of new team members.
2. **Enhances Documentation Quality**
   - Automatically organizes and categorizes comments.
   - Detects missing or outdated documentation.

Why Code Comment Classification Is Useful

3. **Enables Better Developer Tools**
   - Powers intelligent search and recommendation systems in IDEs.
   - Supports automated documentation generation and code summarization.
4. **Supports Software Analytics**
   - Provides insights into developer behavior, code quality, and project evolution.
5. **Saves Time and Effort**
   - Reduces manual labeling and review of large codebases.
   - Streamlines workflows in large software projects.

Future Work

- Extend dataset with more repositories
- Incorporate code context (not just comments)
- Fine-tune advanced transformer models for domain-specific optimization



Project 2: Code Comment Classification

Using Machine Learning and Deep Learning for Multi-Label Classification of Code Comments

Presented by: Luca Francesco Macera and Calogero Carlino
Dataset: NLBSE’23 Tool Competition Dataset (6,738 samples)
Link: https://nlbse2023.github.io/tools/