1 Evaluation Metrics

1.1 Code Summarization

For code summarization, we follow previous work and use three metrics BLEU-4, ROUGE-L and METEOR for evaluation. They are commonly used metrics that measure the similarity between the generated sequence and the ground truth sequence in many fields such as machine translation and text summarization.

BLEU uses n-gram for matching and calculates the ratio of N groups of word similarity between generated comments and reference comments. The score is computed as:

$$BLEU - N = BP \times \exp(\sum_{n=1}^{N} \tau_n \log P_n), \tag{1}$$

where P_n is the ratio of the subsequences with length n in the candidate that are also in the reference. BP is the brevity penalty for short generated sequence and τ_n is the uniform weight $1/N$. We follow previous work and use sentence-level BLEU-4, i.e., $N = 4$, as our evaluation metric. It evaluates generated text by aligning them to reference text and calculating sentence-level similarity scores.

METEOR is a recall-oriented metric which measures how well our model captures content from the reference text in our generated text. It evaluates generated text by aligning them to reference text and calculating sentence-level similarity scores.

$$METEOR = (1 - \gamma \cdot \mathit{frag}^{\beta}) \cdot \frac{P \cdot R}{\alpha \cdot P + (1-\alpha) \cdot R}, \tag{2}$$

where P and R are the unigram precision and recall, *frag* is the fragmentation fraction. α, β and γ are three penalty parameters whose default values are 0.9, 3.0 and 0.5, respectively.

ROUGE-L is based on the Longest Common Subsequence (LCS) between two text and the F-measure is used as its value. Given a generated text X and the reference text Y whose lengths are m and n respectively, ROUGE-L is computed as:

$$P_{lcs} = \frac{LCS(X,Y)}{n}, \quad R_{lcs} = \frac{LCS(X,Y)}{m}, \tag{3}$$

$$F_{lcs} = \frac{(1+\beta^2) P_{lcs} R_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}, \tag{4}$$

where $\beta = P_{lcs}/R_{lcs}$ and F_{lcs} is the computed ROUGE-L value.

1.2 Vulnerability Detection and Clone Detection

For vulnerability detection and clone detection, we follow previous work and evaluate the results by Precision (P), Recall (R), and F1:

$$P = \frac{TP}{TP+FP}, R = \frac{TP}{TP+FN}, F1 = \frac{2 \cdot P \cdot R}{P+R} \tag{5}$$

where TP, FP, TN and FN denote the number of true positives, false positives, true negatives, and false negatives respectively. Since the datasets of vulnerability detection and clone detection are highly imbalanced, e,g., the ratio of vulnerable and non-vulnerable code in Big-Vul is about 1:16, the results of the F1 score is more preferable for these two tasks.

1.3 Evaluation for Continual Learning

To better evaluate the effectiveness of each method under the continual learning setting, following, we further evaluate them by their average performance on all test sets:

$$\Omega = \frac{1}{K} \sum_{i=1}^{K} \Omega_i, \quad \Omega_i = \frac{1}{i} \sum_{j=1}^{i} \Omega_{j,i} \tag{6}$$

where $\Omega_{j,i}$ denotes the performance on the j-th test set after the i-th dataset has been learned. Here Ω can represent any metric we introduced above. Ω evaluates the model's overall performance on all historical datasets. A method with higher value of Ω should perform well on both the current dataset and all previous datasets.