Appendix

1 Evaluation Metrics

We provide the details of the evaluation metrics we used in the experiment.

1.1 BLEU

BLEU measures the average n-gram precision between the reference sentences and generated sentences, with a brevity penalty for short sentences. The formula to compute BLEU-1/2/3/4 is:

$$\text{BLEU-N} = BP \cdot \exp \sum_{n=1}^{N} \omega_n \log p_n, \quad (1)$$

where p_n (n-gram precision) is the fraction of n-grams in the generated sentences which are present in the reference sentences, and ω_n is the uniform weight 1/N. Since the generated summary is very short, high-order n-grams may not overlap. We use the +1 smoothing function (Lin and Och, 2004). BP is a brevity penalty given as:

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases} \quad (2)$$

Here, c is the length of the generated summary, and r is the length of the reference sentence.

1.2 Rouge-L

Based on the longest common subsequence (LCS), Rouge-L is widely used in text summarization. Instead of using only recall, it uses the F-score which is the harmonic mean of precision and recall values. Suppose A and B are generated and reference summaries of lengths c and r respectively, we have:

$$\begin{cases} P_{Rouge-L} = \frac{LCS(A,B)}{c} \\ R_{Rouge-L} = \frac{LCS(A,B)}{r} \end{cases} \quad (3)$$

$F_{Rouge-L}$, which indicates the value of Rouge-L, is calculated as the weighted harmonic mean of $P_{Rouge-L}$ and $R_{Rouge-L}$:

$$F_{Rouge-L} = \frac{\left(1+\beta^2\right) P_{Rouge-L} \cdot R_{Rouge-L}}{R_{Rouge-L} + \beta^2 P_{Rouge-L}} \quad (4)$$

β is set to 1.2 as in (Zhang et al., 2020; Wan et al., 2018).

1.3 Meteor

Meteor is a recall-oriented metric that measures how well the model captures the content from the references in the generated sentences and has a better correlation with human judgment. Suppose m is the number of mapped unigrams between the reference and generated sentence with lengths c and r respectively. Then, precision, recall, and F are given as:

$$P = \frac{m}{c}, \; R = \frac{m}{r}, \; F = \frac{PR}{\alpha P + (1-\alpha)R} \quad (5)$$

The sequence of mapping unigrams between the two sentences is divided into the fewest possible number of "chunks". This way, the matching unigrams in each "chunk" are adjacent (in two sentences) and the word order is the same. The penalty is then computed as:

$$\text{Pen} = \gamma \cdot \text{ frag }^{\beta} \quad (6)$$

where frag is a fragmentation fraction: frag $= ch/m$, where ch is the number of matching chunks and m is the total number of matches. The default values of α, β, γ are 0.9, 3.0 and 0.5 respectively.

1.4 Cider

Cider is a consensus-based evaluation metric used in image captioning tasks. The notions of importance and accuracy are inherently captured by computing the TF-IDF weight for each n-gram and using cosine similarity for sentence similarity. To compute Cider, we first calculate the TF-IDF weighting $g_k(s_i)$ for each n-gram ω_k in reference

sentence s_i. Here ω is the vocabulary of all n-grams. Then we use the cosine similarity between the generated sentence and the reference sentences to compute Cider_n score for n-grams of length n. The formula is given as:

$$\text{Cider}_n\left(c_i, s_i\right) == \frac{< \boldsymbol{g^n}\left(c_i\right), \boldsymbol{g^n}\left(s_i\right) >}{\|\boldsymbol{g}^n\left(c_i\right)\| \, \|\boldsymbol{g}^n\left(s_i\right)\|} \quad (7)$$

where $\boldsymbol{g^n}(s_i)$ is a vector formed by $g_k(s_i)$ corresponding to all the n-grams (n varying from 1 to 4). c_i is the i^{th} generated sentence. Finally, the scores of various n-grams can be combined to calculate Cider as follows:

$$\text{Cider}\left(c_i, s_i\right) = \sum_{n=1}^{N} w_n \, \text{Cider}_n\left(c_i, s_i\right) \quad (8)$$

2 Applying our Framework to Seq2Seq Neural Network Models

We adapt our framework to four neural-based models, NMTGen (M1), CommitBERT (M2), CodeT5-small (M3) and CodeT5-base (M4). The experimental results in different metrics are shown in Figure 1, 2, 3, and 4. We can see that our framework can improve the performance of all four neural models on all programming languages (except CommitBERT on JavaScript in terms of Meteor). These results demonstrate that our approach can be applied to these neural models and boost their performance.

3 Human Evaluation

We conduct a human evaluation to evaluate the effectiveness of the commit message generated by our approach RACE and the other four approaches (NNGen, NMTGen, CommitBERT, and CoRec) in terms of three aspects, Informative, Conciseness and Expressiveness. We define detailed criteria on three aspects for manual labeling in Table 1.

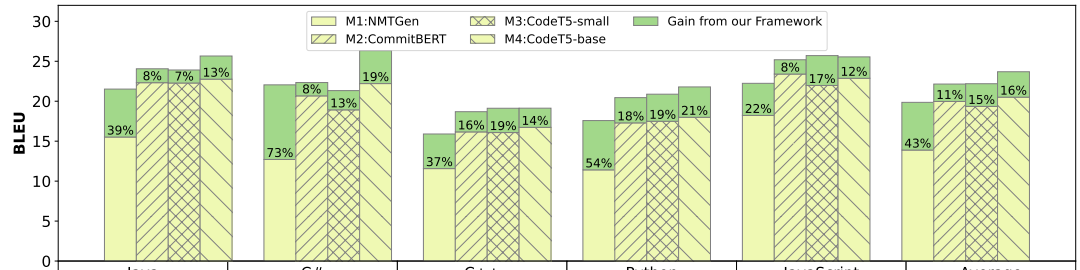
The experimental results in the main text show that RACE outperforms the other approaches in three aspects. We also confirmed the dominance of our approach using Wilcoxon signed-rank tests for human evaluation. The results shown in Table 2 reflect that the improvement of RACE over other approaches is statistically significant with all p-values smaller than 0.05 at 95% confidence level.

References

Chin-Yew Lin and Franz Josef Och. 2004. ORANGE: a method for evaluating automatic evaluation metrics for machine translation. In *COLING*.

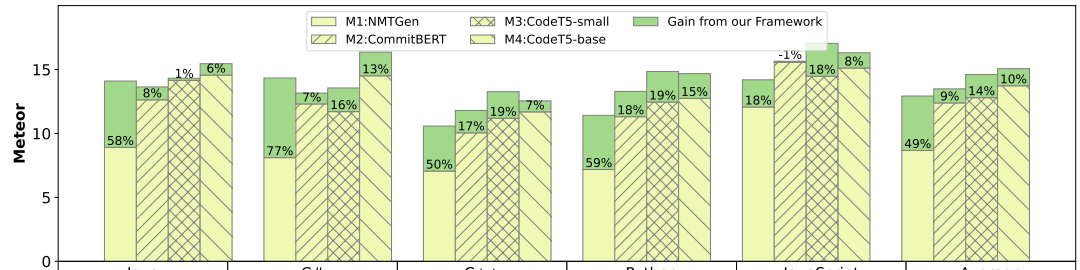
Yao Wan, Zhou Zhao, Min Yang, Guandong Xu, Haochao Ying, Jian Wu, and Philip S. Yu. 2018. Improving automatic source code summarization via deep reinforcement learning. In *ASE*.

Jian Zhang, Xu Wang, Hongyu Zhang, Hailong Sun, and Xudong Liu. 2020. Retrieval-based neural source code summarization. In *ICSE*.



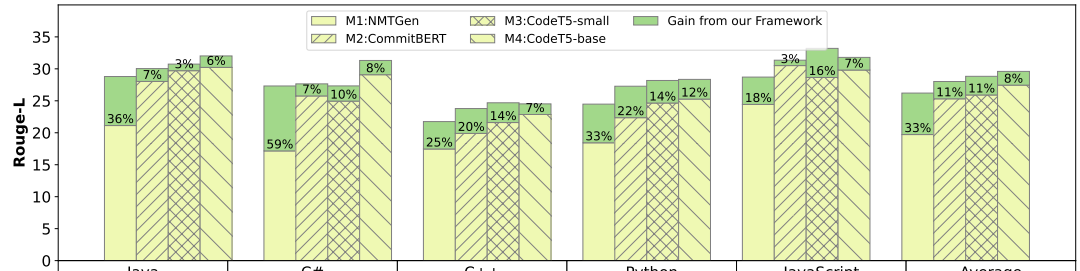
	Java				C#				C++				Python				JavaScript				Average			
	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
+ Gain	21.52	24.06	23.91	25.66	22.05	22.34	21.33	26.33	15.9	18.69	19.12	19.13	17.58	20.45	20.88	21.79	22.24	25.19	25.71	25.55	19.86	22.15	22.19	23.69
Original	15.52	22.32	22.28	22.76	12.71	20.67	18.92	22.21	11.57	16.16	16.08	16.73	11.41	17.29	17.49	17.99	18.22	23.4	21.97	22.87	13.89	19.97	19.35	20.51

Figure 1: Performance gains on four models. The results are evaluated under BLEU.



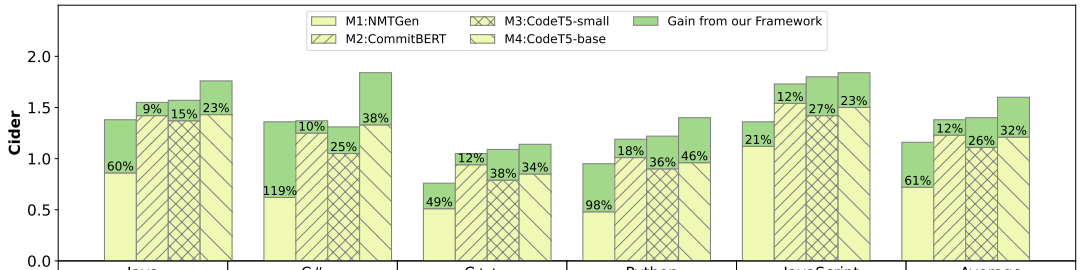
	Java				C#				C++				Python				JavaScript				Average			
	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
+ Gain	14.1	13.64	14.32	15.46	14.34	13.16	13.56	16.37	10.58	11.8	13.27	12.55	11.42	13.29	14.85	14.68	14.2	15.56	17.05	16.31	12.93	13.49	14.61	15.07
Original	8.91	12.63	14.16	14.57	8.11	12.31	11.71	14.51	7.06	10.05	11.19	11.69	7.18	11.31	12.46	12.74	12.07	15.64	14.48	15.12	8.67	12.39	12.8	13.73

Figure 2: Performance gains on four models. The results are evaluated under Meteor



	Java				C#				C++				Python				JavaScript				Average			
	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
+ Gain	28.8	30.04	30.74	32.02	27.32	27.65	27.33	31.31	21.75	23.79	24.69	24.52	24.48	27.28	28.18	28.35	28.71	31.35	33.2	31.79	26.21	28.02	28.83	29.6
Original	21.13	28.03	29.71	30.23	17.16	25.76	24.95	29.08	17.46	19.9	21.6	22.86	18.43	22.36	24.65	25.27	24.43	30.51	28.65	29.81	19.72	25.31	25.91	27.45

Figure 3: Performance gains on four models. The results are evaluated under Rouge-L.



	Java				C#				C++				Python				JavaScript				Average			
	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
+ Gain	1.38	1.55	1.57	1.76	1.36	1.37	1.31	1.84	0.76	1.05	1.09	1.14	0.95	1.19	1.22	1.4	1.36	1.73	1.8	1.84	1.16	1.38	1.4	1.6
Original	0.86	1.42	1.37	1.43	0.62	1.25	1.05	1.33	0.51	0.94	0.79	0.85	0.48	1.01	0.9	0.96	1.12	1.54	1.42	1.5	0.72	1.23	1.11	1.21

Figure 4: Performance gains on four models. The results are evaluated under Cider.

Score	Meaning
Informativeness Is the important information about the code diff reflected in the commit message?	
0	Missing all of the information about the code diff.
1	Is missing some very important information that can hinder the understanding of the code diff.
2	Is missing some information but some of the missing information is not necessary to understand the code diff.
3	Is missing some information but all of the missing information is not necessary to understand the code diff.
4	Is not missing any information.
Conciseness Is there extraneous information included in the commit message?	
0	All of the information is unnecessary.
1	Has a lot of unnecessary information.
2	Has some unnecessary information.
3	Has a little unnecessary information.
4	Has no unnecessary information.
Expressiveness How readable and understandable is the commit message?	
0	Cannot read and understand.
1	Is hard to read and understand.
2	Is somewhat readable and understandable.
3	Is mostly readable and understandable.
4	Is easy to read and understand.

Table 1: The meaning of scores in human evaluation on three metrics.

Model	Informativeness	Conciseness	Expressiveness
CommitBERT	$1.21e^{-24}$	$3.12e^{-22}$	$7.90e^{-05}$
NNGen	$2.04e^{-27}$	$2.28e^{-25}$	$4.28e^{-08}$
NMTGen	$8.64e^{-30}$	$5.87e^{-28}$	$2.55e^{-13}$
CoRec	$6.90e^{-26}$	$2.57e^{-26}$	$1.27e^{-06}$

Table 2: Statistics significance p-value of RACE over other approaches in human evaluation.