Summary

**Improved Automatic Summarization of Subroutines via Attention to File Context**

**Arthur : Sakib Haque , Alexander LeClair , Lingfei Wu , Collin McMillan**

**Conference : In 17th International Conference on Mining Software Repositories (MSR ’20), October 5–6, 2020, Seoul, Republic of Korea**

**INTRODUCTION :** One of the most important aspects of software documentation is the generation of short, usually one-sentence, descriptions of the subroutines in the software source code. The task of generating these descriptions has become known as «summarization» of subroutines. Yet while currently a vast majority of summarization is handled manually by a programmer, automatic summarization has a long history of scientific interest and been described as a «holy grail» of SE research. A flurry of recent research has started to make automatic summarization a reality.

Nearly all of the literature on these AI-based approaches to subroutine summarization is inspired by Neural Machine Translation use an attention mechanism to learn associations between subroutines in the file context to words in the target subroutine’s summary during training. Our idea is novel because existing approaches generally only attend words in the output summary to words in the target subroutine. In our experimental section, we demonstrate that our approach improves existing baselines by providing orthogonal information to help the model provide better predictions for otherwise difficult to understand subroutines.

**Methodology :** we follow the methodology established by many papers on code summarization in both SE and machine translation in NLP. We obtain a standard dataset , then use several baselines plus our approach to create predictions on the dataset, then compute quantitative metrics for the output. The quantitative metrics we use are BLEU and ROUGE . These two metrics have various advantages and disadvantages, but one or another form the foundation of nearly all experiments on neural-based translation or summarization.In cases when the reference is only three words long we calculate onlyFirst, we provide an overview comparison of predictions by different models to determine whether the inputs give orthogonal results and estimate the proportion of predictions may be helped by file context. Attendgru This baseline represents a «no frills» attentional encoderdecoder model using unidirectional GRUs. We configure attendgru with identical hyperparameters to our approach whenever they overlap . Ast-attendgru This is the approach LeClair et al.

proposed an attention-only machine translation model in 2017. It was received in the NLP community with significant fanfare so, given the success of the model for NMT, we evaluate it as a baseline. proposed using a graph-NN to model source code, but applied it to a code generation task in their implementation. To reproduce the idea as a baseline for code summarization, we use a graph-NN-based text generation system proposed by Xu et al.

described in Section 3.1, is a recent code summarization approach based on AST paths that is reported to have good results on a C# dataset. We reimplemented the approach from Section 3.2 of their paper, with their online implementation as a guide. Note we did not use their implementation verbatim. Otherwise, it would not have been possible to know whether performance differences were due to file context or these other factors.

**Result :** This paper advances the state of the art by demonstrating how file context can be used to improve neural source code summarization. The idea that file context includes important clues for understanding subroutines is well-studied in software engineering and especially program comprehension – it has even been proposed for code summarization . The result is that the model is able to use more words from that context, as in the following examples for maximum reproducibility, number is method ID in the dataset followed by the reference summary, attention visualizations As with all papers, our work carries threats to validity and limitations. For one, we use a quantitative evaluation methodology, which while in line with almost all related work and which enables us to evaluate the approach over tens of thousands of subroutines, may miss nuances that a qualitative human evaluation would catch.

However, space limitations prevent us from including a thorough discussion of both in a single paper, so we defer a human evaluation for extended work. Another limitation is that we evaluate only against a Java dataset. This dataset is the largest available that follows good practice to avoid biases , but caution is advisable when generalizing the work to other languages.