# ERNIE-Code: Beyond English-Centric Cross-lingual Pretraining for Programming Languages

## Summary

Software engineers who are currently working on the same programming language may speak in different languages for their communication purposes. Considering this scenario we have seen that some communication barriers arrises while defining the source code to some other contributors who are willing to work on the same project. The person who does not use English as his native language has to write their source code in English. Therefore, to minimize the gap between the Natural language and the programming language for different large language models the Authors of the paper have introduced ERNIE-Code a pre-trained language model for 6 different programming languages and 114 Natural Languages.

**Motivation & Purpose-** As mentioned earlier, the sole purpose of the authors of this paper is to minimize the gap between the multilingual programming language and the multilingual natural language. Also, the main purpose of this paper is to introduce code-to-text, text-to-code, and code-to-code generation. Basically, the authors have tried to make the summarization of multilingual code text-to-text translation.

**Methodology:** The authors of this paper have tried to pre-train their model by Span-corruption Language Modeling and then have used Pivot-based translation language Modelling. These two methods have been used in order to narrow the cross-lingual gap among the non-English natural and Programming languages. While talking about the pre-trained model the authors have also used pre-training corpora from CodeSearchNet as it consists of Python, Java, Ruby, and other 3 different programming languages then for Natural languages the authors have used CommonCrawl Corpus and parallel data from OPUS website consisting of 15 different languages. In this paper, the authors have used BLEU and ROUGE to evaluate the semantic and syntactic aspects of the test to code. One of the interesting things the authors of the paper have introduced is that Zero-Shot Prompting is used to verify that the translation of code-to-text matches the selected language by the user. From the paper, it is seen that the proposed model performs excellently on zero-shot capability for Japanese and Russian code-to-text generation.

**Conclusion:** To conclude I must say the authors of this paper have done a great job by introducing their own pre-trained model as this solves most of the code-to-text and text-to-code problems faced by programmers who do not have English as their first language. As this paper has included 116 natural languages and 6 programming languages in their model to pave a new way of research in code-to-text translation and vice versa it was worth giving a review even though of some limitations.

## Limitations

The very first Limitation which has been discussed by the authors is that due to a lack of available benchmarks the evaluation of the performance of the proposed model was not enough to perfectly evaluate since they only used Zero-shot prompting.

The second limitation that I have come up with considering this paper is that the authors have only used CodeSearch Net's resource to get all the Programming language corpus. Which I believe is not enough. By increasing the amount of resources the pair of natural and programming language could have been more rich.

## Synthesis

As mentioned earlier the authors have paved a new way for research regarding automated summarization of code written in English to any other natural language. Though more resources of collecting code to enrich could have been included to enrich the dataset for the model. Also, more evaluating tools should have been added other than just zero-shot prompting.For future work I would rather like to do the same sort of research on code to bangla text summarization.