An Empirical Analysis on the Performance of UniXcoder

Tim van Dam

Supervisor Maliheh Izadi
Professor Arie van Deursen
Advisor Georgios Gousios

01 Introduction

Machine Learning is increasingly common for code completion.
Great strides have been made - single tokens, but also entire lines and code blocks can be predicted with relatively high accuracy.

Providing more information to code completion models, such as type annotations and comments, might help these models to understand the code better, which can improve code completion.

This study aims to answer whether **UniXcoder** [1] is able to use type annotations and comments to improve code completion performance.

02 Methodology

UniXcoder is trained on three TypeScript datasets based on the most popular GitHub repositories: **TS1K-18, TS1K-18-E and TS1K-22.**

The datasets contain different amounts of type annotations: **48.66%, 87.74%,** and **45.79%** of the maximum amount of type annotations possible were present in the datasets.

UniXcoder is trained on TypeScript and JavaScript. JavaScript is generated from TypeScript using the TypeScript compiler.

Every dataset is trained on multiple times: once with all comments, once without any comments, once with only singleline comments, and once with only multiline comments.

Finally, the models are tasked to complete lines of code, and are evaluated by measuring a number of metrics on their predictions: **Exact Match, Edit Similarity, BLEU-4** [2], **ROUGE-L** [3] and **METEOR** [4].

03 Results

The TypeScript models outperformed the JavaScript models on **every metric,** but **margins were small.**

TypeScript models trained on **TS1K-18-E**, which contains the highest amount of type annotations, outperformed JavaScript models **by a larger margin** than TypeScript models trained on other datasets (figure 1).

Models trained **only singleline comments** or **only multiline comments** often did better than models trained on all comments (figure 2). However, the results are not consistent, and future work needs to confirm how well comments contribute to performance.

Figure 1

TypeScript Performance relative to JavaScript Performance 1

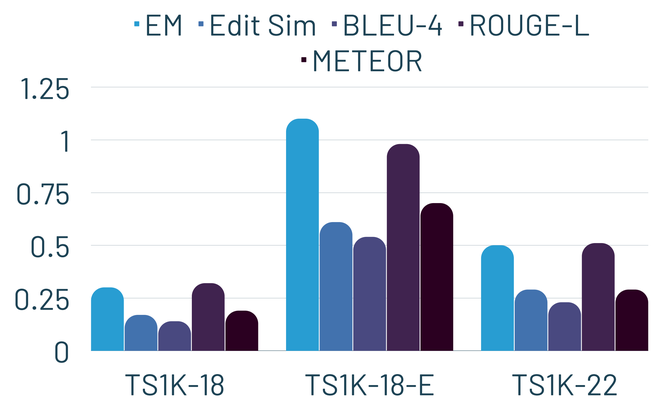
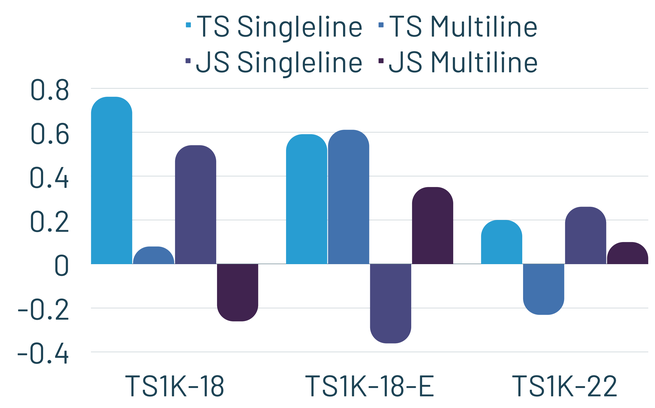


Figure 2

EM performance on Singleline Comments / Multiline Comments vs All Comments



04 Conclusion

The outperformance by TypeScript models on all metrics suggests a positive effect on code completion as a result of type annotations. The outperformance growing when more types are present strengthens this.

Restricting comments to only a certain type may help results. But results marginal and not completely consistent

Future research is required to confirm whether the inconsistent results regarding comments are caused by comment contents, or whether it shows that comments do not contribute much to language understanding.

05 References

[1] Guo et al. (2022). UniXcoder: Unified cross-modal pre-training for code representation
[2] Papineni et al. (2002). BLEU: A method for automatic evaluation of machine translation
[3] Lin. (2004). ROUGE: A package for automatic evaluation of summaries
[4] Banerjee et al. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgements

TU Delft Delft University of Technology