Jens Knoop, Uwe Zdun (Hrsg.): Software Engineering 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 25

Intelligent Code Completion with Bayesian Networks

Sebastian Proksch[1], Johannes Lerch[1], and Mira Mezini[1]

Abstract: Code completion is an integral part of modern Integrated Development Environments (IDEs). Intelligent code completion systems can reduce long lists of type-correct proposals to relevant items. In this work, we replace an existing code completion engine named Best-Matching Neighbor (BMN) by an approach using Bayesian Networks named Pattern-based Bayesian Network (PBN).We use additional context information for more precise recommendations and apply clustering techniques to improve model sizes and to increase speed.

We compare the new approach with the existing algorithm and, in addition to prediction quality, we also evaluate model size and inference speed. Our results show that the additional context information we collect improves prediction quality, and that PBN can obtain comparable prediction quality to BMN, while model size and inference speed scale better with large input sizes.

Keywords: Recommender System, Static Analysis, Machine Learning, Evaluation

1 Motivation

Code completion systems are an integral part of modern Integrated Development Environments (IDEs). They reduce the amount of typing required, thus accelerating coding, and are often used by developers as a quick reference for the Application Programming Interface (API), because they show which fields and methods can be accessed in a certain context.

Traditional code completion systems determine the static type of the variable on which the developer triggers the completion and propose all type-correct methods to the developer. Such a list is often very long with many irrelevant items. More intelligent code completion systems reduce this list to relevant items. They extract a feature vector that describes the current edit location, match this feature vector to released source code found in repositories, and propose relevant methods based on the identified example code.

The prediction quality of intelligent code completion systems mainly depends on the available features, the number of analyzed repositories, and the underlying model that calculates the proposals. To further enhance the results of existing code completion approaches, more repositories need to be analyzed in order to see more examples and more features need to be extracted to have a more precise description of the context. Both results in a rapidly growing size of the available input and not all existing approaches scale well.

Existing completion engines are typically evaluated based on their prediction quality. As the code completion engine is supposed to be used by end users on machines with limited

[1] Technische Universität Darmstadt, Fachbereich Informatik, Fachgebiet Softwaretechnik, Hochschulstr. 10, 64289 Darmstadt, Deutschland, <lastname>@st.informatik.tu-darmstadt.de

resources, it is also necessary to consider memory consumption of the underlying models and computation speed. The three quality dimensions - prediction quality, prediction speed, and model sizes - are not orthogonal and the mutual effect they have on each other must be considered. The hypothesis is that prediction quality is increased by considering more features of the structural context. However, this will presumably increase the model size and negatively affect prediction speed. We need code completion engines that provide a good tradeoff between these quality dimensions or are even configurable along them.

2 Contributions

This paper contributes towards tackling these problems and presents advances to the state of the art in intelligent code completion systems in three ways:

(1) We extended the static analysis of the *best-matching neighbor* approach (BMN) and extracted more context information. We show that this indeed improves prediction quality by up to 3% at the cost of significantly increased model sizes by factor 2 and more.

(2) We introduced a new approach for intelligent code completion called *pattern-based bayesian network* (PBN), a new technique to infer intelligent code completions that enables to reduce model sizes via clustering. We introduced a clustering approach for PBN that enables to trade-off model size for prediction quality.

(3) We extended the state-of-the-art methodology for evaluating code completion systems. We perform comprehensive experiments to investigate the correlation between prediction quality and different model sizes. We show that clustering can decrease the model size by as much as 90% with only minor decrease of prediction quality. We also perform a comprehensive analysis of the effect of input data size on prediction quality, speed and model size. Our experiments show that prediction quality increases with increased input data and that both the model size and prediction speed scales better with the input data size for PBN compared to BMN.

We have released downloadable artifacts to allow replication of our results.[2] The released artifacts includes the dataset used in the paper and the complete source code (i.e., all intelligent code completion engines and the evaluations). We encourage other researches to compare their results based on the same data set.

3 Summary

Our results show that the additional context information we collect improves prediction quality, especially for queries that do not contain method calls. We also show that PBN can obtain comparable prediction quality to BMN, while model size and inference speed scale better with large input sizes.

[2] http://www.st.informatik.tu-darmstadt.de/artifacts/pbn/