ChatGPT

OpenTauによるTypeScript型注釈生成

OpenTauはTypeScriptの型推論問題に対し、「プログラム分解」「探索と検証」「型検査統合」に基づく検索ベース手法を提案する。大規模なプログラムを対象とし、LLM（大規模言語モデル）による型予測とTypeScriptコンパイラによる検証を組み合わせる点が特徴である [1] [2]。

プログラム分解

OpenTauでは大規模なソースコードを再帰的に**木構造**に分解し、各サブプログラム（部分ノード）ごとに型推論を行う枠組みを導入している [1]。具体的には、プログラム全体を宣言単位でツリー構造化し、葉ノードから順にLLMを用いて型注釈候補を生成する。こうして得られた部分ごとの候補を組み合わせて上位ノードの型推論に利用することで、モデルのコンテキストウィンドウ制限を回避しながら文脈を局所化して推論を実行する。

探索と検証ループ

各ノードでは、モデル予測とコンパイラによる**局所型推論**を組み合わせて複数の型注釈候補を生成する。葉ノードで候補を生成後、親ノードでは子ノードの全候補の組み合わせ（組み合わせ数上限付き）から入力プロンプトを作成し、新たに型注釈候補を生成する。最後に全候補解が得られた段階で、TypeScriptの型チェッカ（tsc）を用いて各候補を順次検証する。型エラーの数が最も少ない（理想的にはゼロの）候補を最終出力とする探索・検証ループを通じて、出力プログラムの型検査整合性を確保する [3]。

評価指標：型検査成功率

OpenTauでは型注釈の品質指標として、生成プログラムが型エラーなく通るか否かを評価する**型検査成功率（file-level type-check success）**を提案している。従来の精度評価と異なり、実際にtscでエラーが生じないことを最終ゴールとする手法である [2]。評価実験では、OpenTauはTypeScriptファイル単位で47.4%の成功率を達成し、最良の従来手法に対して絶対値で14.5%の改善を報告している [4] [5]。なお、成功率指標は従来の個別型注釈の正解率よりも実践的な評価指標であるとしている [2]。

他手法との比較

先行研究には、Holtzenらの**TypeWriter**（2020年）など型チェッカを使った探索手法や、大規模言語モデル（CodexやSantaCoderなど）を用いる生成手法がある [6]。TypeWriterも型チェッカによる探索を行うが、OpenTauは最新のLLM予測にプログラム分解と検証ループを組み合わせることで性能を向上させている。また、一般的なLLM生成モデルでは生成精度（個々の型注釈の正答率）が評価されることが多いが、OpenTauは「型検査を通過するか」を最終評価基準とする点で差別化されている [2] [6]。このように、OpenTauは生成と検証を統合するアプローチにより、従来法を上回る高い型検査成功率を実現している。

参考文献: OpenTau (Cassano et al. 2023) [1] [3] [5] [6] など。

---

[1] [2] [3] [4] [5] [6] [2305.17145] Type Prediction With Program Decomposition and Fill-in-the-Type Training
https://ar5iv.labs.arxiv.org/html/2305.17145