基于深度学习的代码摘要生成与优化方法研究

方　清

（阿荣旗职业中等专业学校，内蒙古 呼伦贝尔 162750）

摘　要： 代码摘要生成（Code Summarization）任务能够自动提取代码的核心功能和逻辑，为开发者提供快速理解和文档化支持。针对传统基于模板和规则的代码摘要方法效率低、语义理解能力有限的问题，提出一种基于深度学习的代码摘要生成与优化方法。该方法首先将代码序列化为抽象语法树（AST）表示，再利用改进的Transformer模型进行编码和生成，并结合强化学习提升生成摘要的准确性和可读性。

关键词： 深度学习；代码摘要；Transformer；AST表示；强化学习；自动文档生成

中图分类号： TP391　　**文献标识码：** A

Research on code summarization generation and optimization method based on deep learning

FANG Qing

（Arongqi Zhiye Zhongdeng Zhuanye Xuexiao，Hulunbuir，Inner Mongolia 162750，China）

Abstract：Code summarization can automatically extract the core functionality and logic of code，providing developers with support for rapid understanding and documentation. Aiming at the problems of low efficiency and limited semantic understanding capability in traditional template-based and rule-based code summarization methods，a code summarization generation and optimization method based on deep learning is proposed. The method first serializes code into an Abstract Syntax Tree（AST）representation，then employs an improved Transformer model for encoding and generation，and combines reinforcement learning to optimize the accuracy and readability of generated summaries.

Key words：deep learning，code summarization，Transformer，AST representation，reinforcement learning，automatic documentation generation

1　引言

随着开源软件与企业级系统代码量呈指数级增长，开发者在代码阅读、维护和复用过程中面临巨大挑战。手动编写代码文档和注释不仅耗时，而且易受开发者主观影响，导致文档质量参差不齐。自动生成代码摘要成为解决这一问题的重要手段。传统方法主要依赖规则模板或信息检索技术，对代码语法与语义的理解有限，难以生成高质量自然语言描述。近年来，深度学习（尤其是Transformer模型）在自然语言处理领域取得突破性进展，其强大的序列建模能力为代码摘要生成提供了新思路[1]。本文提出一种基于深度学习的代码摘要生成与优化方法，以提升生成摘要的语义准确性和可读性。

2　相关工作

2.1　传统代码摘要生成方法

传统代码摘要生成方法主要依赖静态分析、模板匹配及关键词提取技术。静态分析方法通过解析代码结构（如函数调用关系、控制流图和变量依赖关系），提取代码的基本功能信息，用于生成简要摘要。然而，这类方法通常对代码逻辑理解有限，只能捕捉局部结构特征，难以处理复杂业务逻辑。模板匹配方法则预先定义摘要模板，将函数名称、变量名或已有注释按照固定规则拼接生成摘要。该方法实现简单，但生成的摘要高度依赖模板设计，缺乏灵活性与通用性，难以适应多样化的代码风格和命名习惯。信息检索方法通过查找和目标代码相似的历史代码片段，利用已有文档生成摘要，这在一定程度上提高了摘要的可读性，但对新颖或复杂代码片段效果有限[2]。

2.2　基于深度学习的方法

随着深度学习的发展，研究者逐渐将神经网络应用于代码摘要生成任务。最早使用的循环神经网络（RNN）及其变体LSTM与GRU能够对代码序列的上下文信息进行建模，捕捉局部依赖关系，从而生成具有一定可读性的自然语言摘要。然而，RNN类模型在处理长序列时易出现梯度消失或梯度爆炸问题，导致远程依赖捕

捉效果差，生成摘要的质量受限。为解决长距离依赖问题，Transformer模型引入自注意力机制（Self-Attention），其能够在编码阶段捕获代码序列全局信息，并生成更准确、语义连贯的摘要。基于Transformer的预训练模型（如CodeBERT与GraphCodeBERT）通过大规模代码和自然语言联合训练，显著提升了模型对代码语义的理解能力[3]。

2.3 优化策略

尽管基于深度学习的方法在代码摘要生成上取得了较大进展，但生成质量仍受限于训练数据、模型结构及评估指标等因素。因此，研究者提出了多种优化策略以进一步提升摘要效果。首先，强化学习（Reinforcement Learning，RL）通过自定义奖励函数提升生成文本与真实摘要的一致性（如使用ROUGE或BLEU分数作为奖励信号），引导模型生成更高质量的摘要。其次，将代码表示为树结构（如AST）或图结构（如GNN），以捕获代码的语法和数据依赖关系，使模型理解能力更强，尤其在函数调用复杂或逻辑嵌套较深的代码片段中表现优异。此外，混合编码器策略将序列编码与图编码结合，通过并行捕获代码序列信息和图结构信息，提升模型对语义和结构的综合理解能力[4]。

3 方法设计

本文方法主要由AST表示、改进Transformer模型与优化策略3个部分组成。

3.1 AST表示

AST表示是代码摘要生成的基础，直接决定模型对代码语义和结构的理解能力。为充分捕获代码的语法与语义信息，本文采用AST表示方法将源代码结构化地表达为树状节点，其中每个节点对应代码元素（如函数、变量、操作符等），从而保留完整的语法信息。首先，通过语法解析器将原始代码解析为AST，确保语法层级关系清晰。其次，将树结构节点序列化，即按深度优先或广度优先顺序将每个节点及其子节点转换为节点序列，以便输入深度学习模型。节点序列中每个节点不仅包含类型信息，还包括标识符、数据类型等属性，以丰富语义信息[5]。最后，将节点及其属性映射为高维嵌入向量，形成可训练的输入表示。设节点序列为$n_1, n_2, \cdots, n_T$，其嵌入向量表示为：

$$\boldsymbol{h}_i=\boldsymbol{W}_e\cdot\text{one_hot}(n_i)+\boldsymbol{b}_e \tag{1}$$

其中，$\boldsymbol{W}_e$为嵌入矩阵；$\boldsymbol{b}_e$为偏置项；$\boldsymbol{h}_i$为节点n_i的向量表示。同时，AST序列化方法可支持不同编程语言的统一表示，便于跨语言训练和迁移学习。

3.2 改进Transformer模型

改进Transformer模型是代码理解与自然语言描述之间的桥梁。本文采用改进的Transformer模型，充分利用其自注意力机制处理长序列和捕获全局依赖的能力。编码器部分对AST序列进行多层自注意力编码，每层通过计算注意力权重α_{ij}衡量节点间的语义相关性：

$$\alpha_{ij}=\frac{\exp((\boldsymbol{h}_i\boldsymbol{W}_Q)(\boldsymbol{h}_j\boldsymbol{W}_K)^{\mathrm{T}}/\sqrt{d_k})}{\sum_{k=1}^{T}\exp((\boldsymbol{h}_i\boldsymbol{W}_Q)(\boldsymbol{h}_k\boldsymbol{W}_K)^{\mathrm{T}}/\sqrt{d_k})} \tag{2}$$

其中，$\boldsymbol{W}_Q$和$\boldsymbol{W}_K$为查询和键矩阵；d_k为维度缩放因子，确保注意力分布稳定。解码器部分根据编码器输出生成自然语言摘要，通过多头注意力机制同时关注不同节点的信息，并结合位置编码捕捉AST的非线性结构。为更好地反映代码树结构，本文引入结构感知位置编码（Structure-Aware Positional Encoding，SAPE），将节点深度与兄弟关系融入位置向量，实现对代码层次的敏感表示。

3.3 优化策略

为提升生成摘要的可读性和准确性，本文在模型训练和生成阶段引入多种优化策略。首先，强化学习优化通过自定义奖励函数$R(y, \hat{y})$指导模型生成与真实摘要更接近的文本，其中包括ROUGE-L得分奖励和语法一致性奖励：

$$R(y,\hat{y})=\lambda_1\cdot\text{ROUGE-}L(\hat{y},y)+\lambda_2\cdot\text{SyntaxConsistency}(\hat{y}) \tag{3}$$

其中，λ_1和λ_2为权重系数，旨在平衡语义和语法质量。其次，在解码阶段结合温度采样和束搜索（Beam Search），温度采样控制生成文本的多样性，束搜索保证生成序列的全局最优性。再次，本文采用多任务学习策略，将函数类型预测、关键变量识别等辅助任务加入训练，利用共享编码器提升模型的语义理解能力。最终，依托这些优化策略，模型不仅生成的摘要与原始代码高度匹配，还保证语法合理、可读性强，显著提高了代码摘要生成的实用性和可靠性。

4 实验与结果

4.1 数据集

本文实验选取了CodeSearchNet和FunCom两个开源代码数据集。CodeSearchNet数据集覆盖多种编程语言（包括Java，Python，JavaScript等），每个函数均配有对应的自然语言注释，可用于训练和评估代码摘要生成模型。FunCom数据集专注于大规模Java函数及其注释，提供更丰富的函数实现和复杂逻辑场景，有利于测试模型在长函数和复杂结构代码上的表现。为保证训练、验证和测试的科学性，本文将数据集按8∶1∶1的比例划分为训练集、验证集和测试集。训练集用于模型学习，验证集用于超参数调整和模型选择，测试集用于最终性能评估。该划分方法保证了模型在不同数据上的泛化能力，同时避免数据泄露对实验结果的影响。通过使用多语言、多类型函数的数据集，本文方法在多场景、多语言的代码摘要生成中均能得到有效验证，以确保实验结果

的可信性和可推广性。

4.2　实验设置

实验采用基于改进 Transformer 架构的模型配置。编码器和解码器均设置为 6 层，每层包含多头注意力与前馈网络模块，隐藏维度设为 512，以保证模型具有足够的表达能力，同时兼顾训练效率。优化器选用 Adam，学习率设为 5e-4，通过逐步衰减策略优化模型收敛。训练过程中使用批量大小为 64，采用早停策略避免过拟合。为评估生成摘要的质量，本文选取 ROUGE-L 和 BLEU 两个指标。其中，ROUGE-L 指标用于衡量生成文本和真实摘要在长序列匹配上的相似度，反映摘要的完整性与语义覆盖率；BLEU 指标用于评价生成文本和参考文本在 n-gram 层面的精确匹配程度，反映摘要的精确性与可读性。

4.3　对比方法

为全面评估本文方法的有效性，实验选取 3 种对比模型：RNN/LSTM、原版 Transformer 和 GraphCodeBERT。RNN/LSTM 模型作为传统 RNN 的代表，能够捕捉代码序列的上下文信息，但在长序列依赖和复杂结构处理上存在局限。原版 Transformer 模型引入自注意力机制，可处理长距离依赖，生成文本质量较高，但缺乏对代码语法结构的显式建模。GraphCodeBERT 则结合图结构编码与预训练模型，通过对代码的 AST 和依赖关系建模，提升了对代码语义的理解能力。

4.4　实验结果

实验结果显示，本文方法在代码摘要生成任务中取得了显著性能提升。具体而言，ROUGE-L 指标达到 0.64，BLEU 指标达到 0.55，相较于 RNN/LSTM 的 0.42 和 0.35 提升明显，亦优于原版 Transformer 和 GraphCodeBERT。该结果表明，本文方法在捕获代码语义、理解函数逻辑及生成自然语言摘要方面具有优势，通过 AST 表示结合改进 Transformer 模型和优化策略，使该模型能够关注关键代码节点，生成的摘要更加完整且可读性更强。此外，本文方法在处理复杂函数和长序列代码时稳定性更高，生成摘要的准确性和流畅度均优于对比模型。

5　结束语

本文提出了一种基于深度学习的代码摘要生成与优化方法，通过 AST 表示、改进 Transformer 模型和优化策略，有效提升了生成摘要的准确性和可读性。实验结果显示，该方法在多个开源代码数据集上优于现有方法，能够为自动文档生成、代码理解及智能维护提供有力支持。未来将进一步扩展模型应用场景，提升跨语言适应性与生成效率。

参考文献：

[1] 王泽治.基于深度学习的多特征代码摘要研究[D].长春：吉林大学，2023.

[2] 要博伦.基于深度学习的跨模态文本摘要生成及三维物体描述应用[D].南京：南京理工大学，2021.

[3] 高宁哼.基于深度学习的代码摘要生成方法研究[D].长春：吉林大学，2022.

[4] 李学文.基于深度学习的代码摘要生成算法研究[D].秦皇岛：燕山大学，2023.

[5] 王豪庆.基于深度学习的多编码器代码摘要自动生成研究[D].广州：广东工业大学，2024.

作者简介：

方清（1985—），硕士，高级讲师，研究方向：计算机科学与技术应用。