

**第二十九届“冯如杯”学生学术科技作品竞赛项目论文**

**Code\_Searcher:**

**基于深度学习的编程助手**

摘要

随着计算机的不断普及发展，编程能力已经成为了人们日常生活中，学生老师科研学习工作中一项不可或缺的能力与工具。编写程序能够帮助我们有效的，随心所欲的利用计算机，极大的提高我们工作学习的效率。然而，也不得不承认编写代码对很多人来说是一项相对困难而且耗时的工作，对于那些并非从事计算机相关行业的人来说，往往希望快速得到解决方案，编写自己需要的程序而不耗费额外而不必要的精力。

以JAVA语言为例，JAVA语言中有丰富的函数库API，可以帮我们解决很多复杂的问题，当有问题需要解决时，找到合适的API可以让我们对问题的解决事半功倍甚至直接解决我们当前的问题。因此，当我们有编程困难时，第一件事便是到网上查询API和code片段，帮助我们打开解决问题的思路。

传统的搜索方法通过自然语言来搜索自然语言，不适合检索代码。因此，我们针对Java编程语言，设计了编程助手，通过自然语言找到与之关联性最高的GitHub代码片段，并检索出需要使用的方法和API, 采用Seq2Seq神经网络模型，分别对代码和自然语言进行encoder,并训练自然语言与代码的对应关系，使我们可以通过自然语言，找到与之相关性最高的5组代码及代码使用的API反馈给用户。

Code\_Searcher采用神经网络训练模型,搜索结果相关性高，且虽然使用神经网络，但将后端代码数据训练与前端输入自然语言的训练分离，实现在搜索准确度高的同时搜索速度快。并为其搭建了Web客户端，方便客户使用。

**关键词**：编程助手，深度神经网络，GitHub，API，Web应用

**Abstract**

As the continuous development and wide spread of computers, the ability of programming has become an essential part of study and research among people’s daily life. Programming can help people use computers more efficiently, do whatever they want and greatly improve the efficiency of learning and working. However, we have to agree that programming can be a really difficult and time-consuming task for many people. Those people who do not work on careers about computers usually hope to solve problems quickly and write programs without requiring extra energy.

Java has rich API (function library) which can help people solve many complicated problems. When there are problems to handle with, finding proper API can help solve the problems much faster or directly solve them. As a result, the first thing when facing programming difficulties is to search API or codes on the Internet, which may broaden our thoughts.

Traditional searching methods use natural language to search natural language. This cannot be applied to searching codes. So, we design a programming assistant to find the most relevant codes in GitHub through natural language and find out the necessary methods and API, using neural network model of Seq2Seq. We will encode the codes and natural language and train the relationship between natural language and codes to find five most relevant set of codes and the API they use to feedback users.

Code\_Searcher use neutral network model to improve the correlation between search keywords and search results. In order to improve the searching speed with high searching accuracy, we separate the data training in the back from the natural language input in the front. Also, we bulid Web client to make the use of searcher more convenient.

**Keyword**s: Programming assistant, Deep neutral network, GitHub, API, Web application

目录

[一、绪论 1](#_Toc4397857)

[（一）问题定义与背景 1](#_Toc4397858)

[（二） 相关应用 1](#_Toc4397859)

[二． 整体设计 2](#_Toc4397860)

[（一）获取数据 2](#_Toc4397861)

[（二）训练code模型 2](#_Toc4397862)

[（三）训练Sentence模型 3](#_Toc4397863)

[（四）神经网络映射 4](#_Toc4397864)

[三． 实际应用 5](#_Toc4397865)

[四．不足与未来发展 6](#_Toc4397866)

[（一）发掘语义信息 6](#_Toc4397867)

[（二）拼接创造代码 6](#_Toc4397868)

[（三）改进前端 6](#_Toc4397869)

[结论 6](#_Toc4397870)

[参考文献 7](#_Toc4397871)

# 一、绪论

## （一）问题定义与背景

我们在编程时，常常有遇到问题不知道该如何去解决的困境，比如在科研任务中要通过不太熟悉的工具实现某一功能，有类似困难时，我们做的第一件事往往是到网上寻找答案，或者是CSDN，又或者是Stack Overflow，在找到类似的代码使用范例时，我们再把它转换成我们想要的格式。在这个过程中，我们实际面对的是一个调用API的问题：选择什么样的API（Application Programming Interface）来实现功能？又怎样用这些API来实现这个功能？显而易见，如果能有一个根据自然语言的请求来自动生成源代码或API的软件的话，我们的开发将会轻松许多。

## 相关应用

一方面，该应用将方便学习者对于github上的特定工程项目进行搜索，我们可以给搜索者提供所有含有其搜索内容的github上的已知项目的链接。另一方面，该应用将大大降低初学者乃至门外汉使用特定语言编程的门槛，他只需要使用自然语言对自己的需求进行描述，就可以得到该需求在特定语言中的实现方法。最后，该应用也将大大节省开发者的开发时间，提高开发效率，使得开发者可以在最短的时间内找到符合自己需求的代码，并为其提供实现样例。

# 整体设计

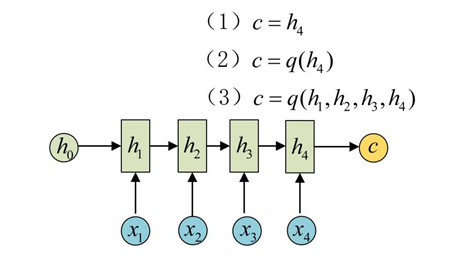
我们的整体设计主要针对代码数据和自然语言数据分别建立神经网络，并将二者分别映射到高维向量空间中去，以达到将两者均转化为数值向量的目的，最后，我们建立两者之间的联系，在抽去的encoder层，即代码层上增加线性映射层，将两个向量集合映射到同一向量空间，并通过对神经网络的训练。具体实现主要分四步：

## （一）获取数据

我们从GitHub上获取真实的项目代码，为确保我们代码的有效性，我们对GitHub中低Star的代码进行了筛除，最终得到我们的代码数据库。

## （二）训练code模型

得到代码数据库后，为了提取出代码特征，我们采取了seq2seq模型中encoder部分来将代码映射到高维向量空间，以达到获取代码特征的目的[2]。

****而在从代码到向量的过程中，我们采用的是深度神经网络中的seq2seq模型的encoder部分。seq2seq是RNN模型的一个变种，RNN（Recurrent Neural Network）是一类专门用于处理序列数据的神经网络，而seq2seq模型的特点则是输入与输出可以不等长，这也就方便了我们实现从代码到向量的转变。

**图 1**

如图所示，h0是我们的输入层，我们将代码从此层输入，经过h1h2h3h4等隐藏层的处理之后从c层输出，得到的就是我们需要的向量。

## （三）训练Sentence模型

根据前馈神经网络模型改进得到skip-gram的神经网络模型，给模型输入一个单词，模型的输出是的上下文wO,1,...,wO,C，上下文window的大小为C，对于输出的这些单词，采用独热码编码方式。下图为神经网络模型：其中向量x代表单词的独热码，对应输出变量{y1,…,yC}，对于在Input layer和Hidden layer之间的权重矩阵W的第i行代表词汇表中的第i个单词的权重需要学习的目标即该矩阵W，其中包含了词汇表内全部单词的权重信息，在模型中，输出的每个单词向量也有一个N×V维输出向量W′。模型中包含了N个节点的隐藏层，隐藏层结点的输入即是输入层所有输入加权后求和，而因为输入向量x采用独热码编码方式，则只有当向量中的元素为非零项时才可以对输入层进行输入，因此隐藏层的输出仅仅与权重矩阵的某一行有关，对于输入向量x，其中存在并且，，存在数学证明：

(1)

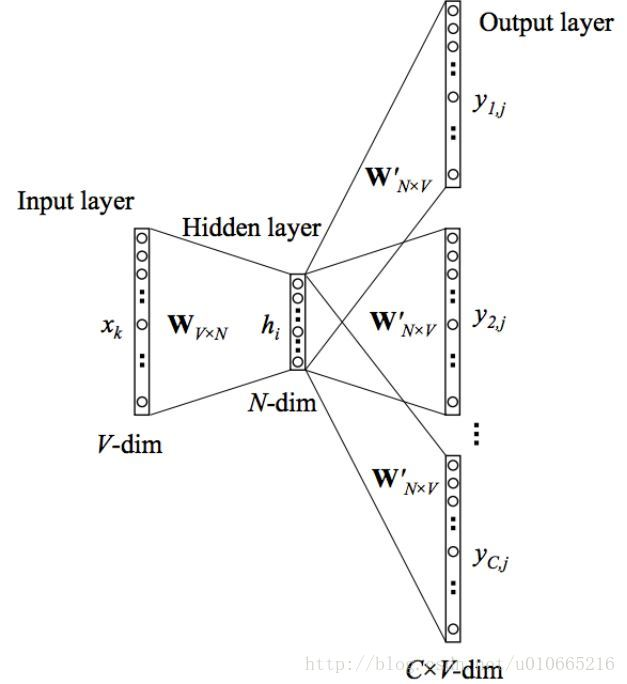
输入使用的独热码，所以此处无需使用函数，模型输出节点C×V的输入由对应输入节点的加权求和计算得出：

(2)

上图中我们可以得出输出层中的每个单词的权重都是共享的，则，最终使用softmax函数可得到第C个单词的分布，

(3)

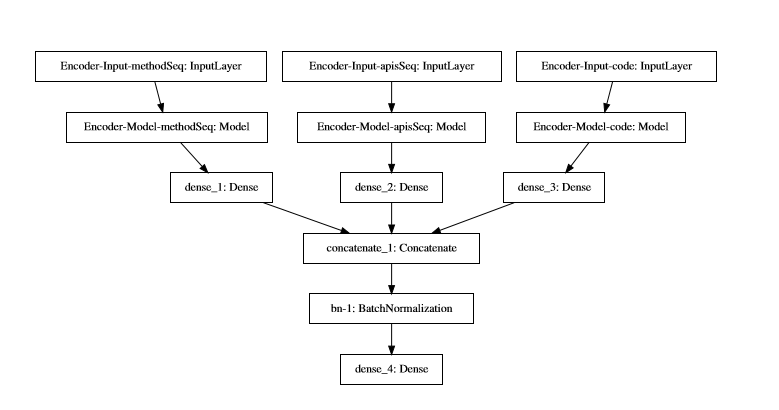
这就是第C个单词第j个节点的概率。是一堆浮点值，而不是独热码向量。



**图 2**

## （四）神经网络映射

在第二部抽取的encoder层之上增加线性映射层,通过训练这个网络，最终实现从code向量空间到自然语言向量空间的映射。得到该网络后，我们便将所有代码进行映射并将映射的结果储存到数据库中，最终，当用户进行检索之后，我们可以通过寻找在空间中距离用户搜索内容所对应向量距离最近的结果，并将结果返还给用户。从而实现我们的目的。



**图 3**

# 实际应用

我们搭建了Web应用来展示我们的搜索结果，如图4：



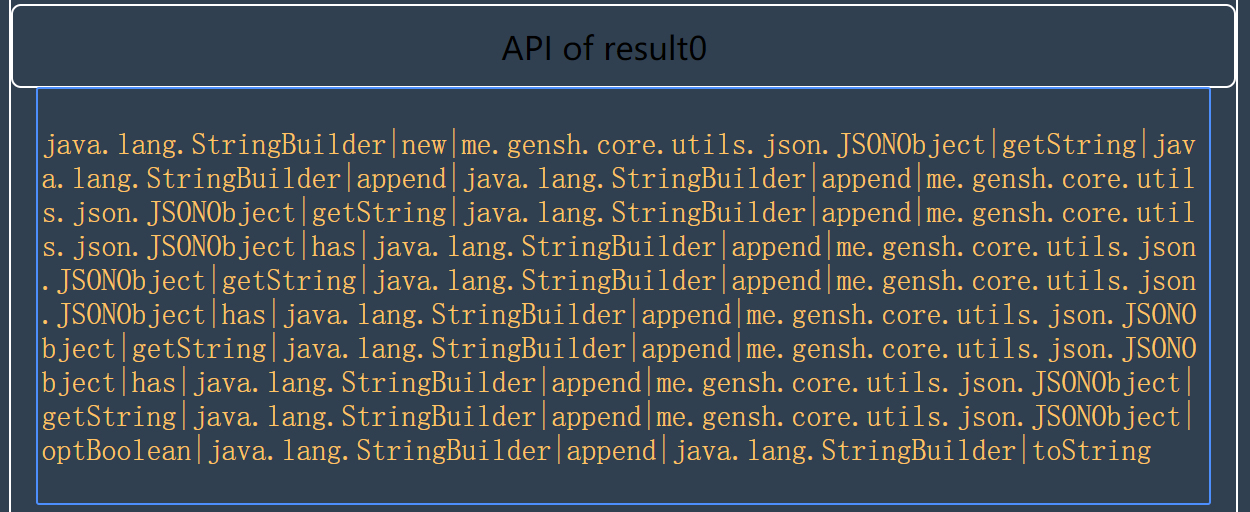
**图 4**

我们展示类似的五组代码，着重展示了响应的代码框架，如图5：



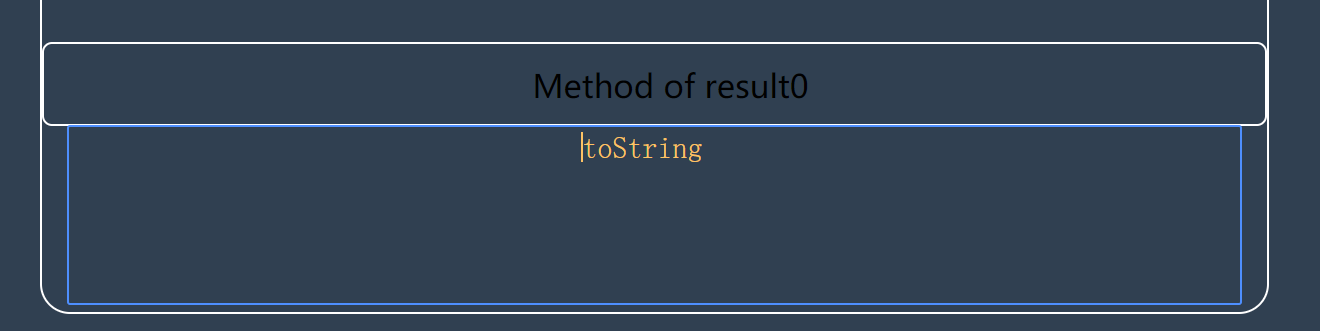
**图 5**

从代码中提取出的API



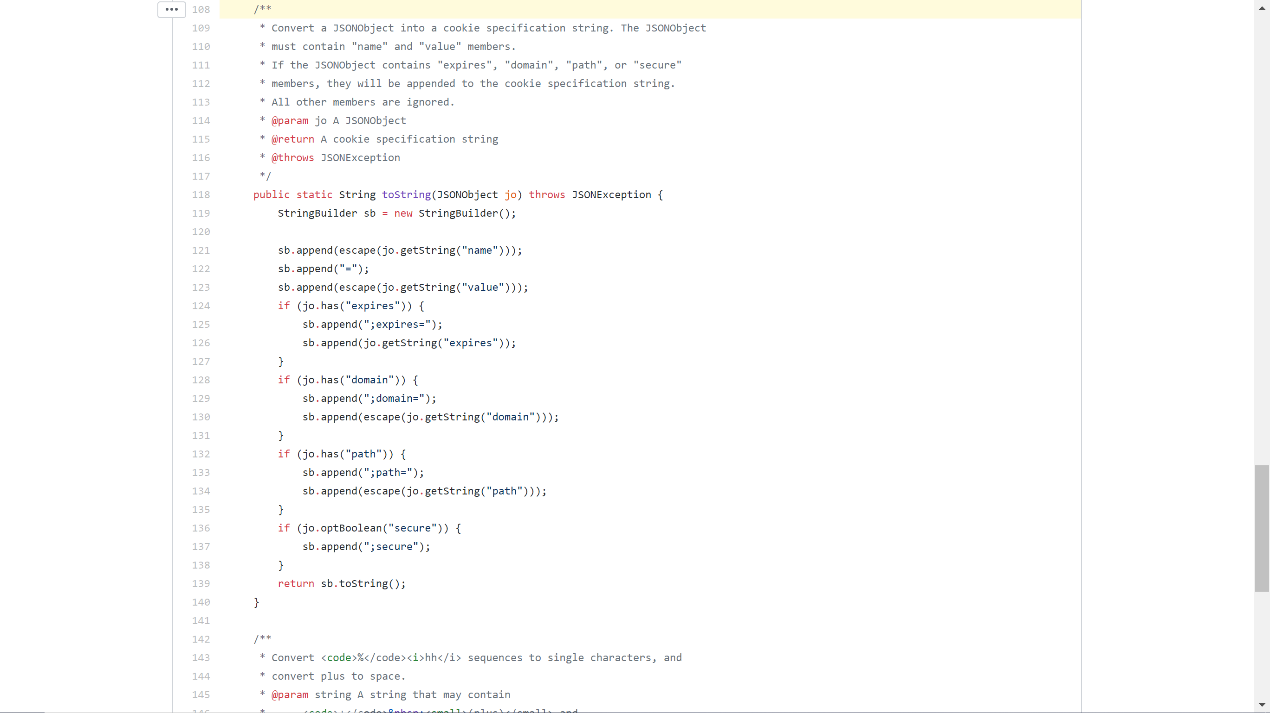
**图 6**

以及使用的方法名



**图 7**

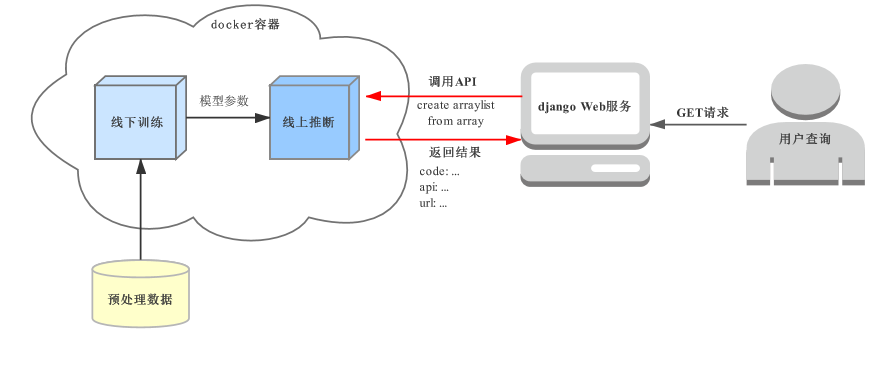
并附上了源代码的链接，若对代码片段不熟悉需要查看原码，或本身对GitHub项目感兴趣，希望找到响应的原码项目进行了解，便可通过链接找到原本的GitHub仓库。



**图 8**

通过对原码的了解，更有利于加深学习编程者对编程方法的理解。

最终形成如下所示的应用模型：



**图 9**

# 四．不足与未来发展

## （一）发掘语义信息

我们希望除了现阶段的代码框架，调用API种类数量，使用方法之外，挖掘更深层次的代码特征，以更好的挖掘代码的语义，使匹配过程更加精确。

## （二）拼接创造代码

考虑一些当前没有API而可能可以通过将几个方法共同使用而解决问题的情况，探究能否发掘几段代码组合后的语义，以解决更多问题，真正实现由现在的代码“创造”新的代码，解决新的问题。

## （三）改进前端

改进前端的展示机制，构建IDE应用程序，使用户可以根据输入的关键字直接在IDE中生成代码进行使用。

# 结论

我们由于传统IR方法的不足，提取代码和自然语言中的语义信息用以匹配代码和用于检索的自然语言，鉴于代码和自然语言提取语义方法不同，我们分别用神经网络模型提取代码和自然语言的语义信息，并通过对embedding构建线性层将两者均映射到768为空间，通过在空间中的距离远近检索语义的相似程度。最终完成匹配。得到用户需要的代码和API种类。同时帮助用户找到code出处，帮助用户熟悉GitHub中实际的项目工程，达到训练编程能力的效果。

# 参考文献

1. Y. Zhang and W. Xiao, *Keyphrase Generation Based on Deep Seq2seq Model*," in IEEE Access, vol. 6, pp. 46047-46057, 2018.
2. Y. Zhang, Y. Wang, J. Liao and W. Xiao, *A Hierarchical Attention Seq2seq Model with CopyNet for Text Summarization*, 2018 International Conference on Robots & Intelligent System (ICRIS), Changsha, 2018, pp. 316-320.
3. P. Rodeghero, C. McMillan and A. Shirey, *API Usage in Descriptions of Source Code Functionality*, 2017 IEEE/ACM 1st International Workshop on API Usage and Evolution (WAPI), Buenos Aires, 2017, pp. 3-6.
4. B. A. Myers, *Human-Centered Methods for Improving API Usability*, 2017 IEEE/ACM 1st International Workshop on API Usage and Evolution (WAPI), Buenos Aires, 2017, pp. 2-2.
5. T. D. Nguyen, A. T. Nguyen, H. D. Phan and T. N. Nguyen, *Exploring API Embedding for API Usages and Applications*, 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE), Buenos Aires, 2017, pp. 438-449.
6. Mikolov T, Sutskever I, Chen K, et al. *Distributed Representations of Words and Phrases and their Compositionality[J]*, 2013, 26:3111-3119.
7. Mikolov T, Chen K, Corrado G, et al. *Efficient Estimation of Word Representations in Vector Space[J]*. Computer Science, 2013.