Artificial Intelligence and Robotics Research 人工智能与机器人研究, 2019, 8(1), 1-6
Published Online February 2019 in Hans. http://www.hanspub.org/journal/airr
https://doi.org/10.12677/airr.2019.81001

Hans汉斯

Function Construction of Interface Intelligent Model

Yebin Lin

Hangzhou Yihu Intelligent Technology Co., Ltd., Hangzhou Zhejiang
Email: 358477162@qq.com

Received: Nov. 22nd, 2018; accepted: Dec. 10th, 2018; published: Dec. 17th, 2018

Abstract

This paper aims to build an overall program that adapts to the requirements through the interface intelligence model and the use of function methods. In the "Algorithm Representation and Use of Interface Models", the representation of the interface and the process of adjusting the unsuitable interface to achieve the desired whole are initially demonstrated. The function we usually use can also be seen as an interface. Various methods are written in a programming language, and then these methods generally take the input and output of in, out. These inputs and outputs can be seen as a forward interface and a negative interface. Each parameter of the method can be regarded as an interface, the input parameter is a negative interface, and the output parameter is a forward interface. Then the interface intelligence model can use these inputs and outputs as interfaces, thus assembling into a variety of implementations of human needs. That is to say, the interface intelligent model can realize the process of automatic construction of function methods.

Keywords

Artificial Intelligence, Interface Driver, Algorithm Model, Function Composition, Intelligent Programming

接口智能模型的函数构建

林烨斌

杭州蚁护科技有限公司，浙江 杭州
Email: 358477162@qq.com

收稿日期：2018年11月22日；录用日期：2018年12月10日；发布日期：2018年12月17日

摘　要

本文旨在通过接口智能模型，使用函数方法，构建成适应需求的整体程序。在《接口模型的算法表示

文章引用: 林烨斌. 接口智能模型的函数构建[J]. 人工智能与机器人研究, 2019, 8(1): 1-6.
DOI: 10.12677/airr.2019.81001

和使用例子》中初步展示了接口的表示和通过调整不适合的接口最终达到想要整体的过程。我们平时使用的函数也可以看成是一种接口。通过编程语言编写各种方法，然后这些方法一般都会带着in,out的输入输出。而这些输入输出，就可以看作正向接口和负向接口。方法的每个参数都可以看成是一个接口，输入参数为负向接口，输出参数为正向接口。那么接口智能模型就可以把这些输入、输出当作接口，从而拼装成为各种实现人为需求的整体。也就是接口智能模型可以实现函数方法的自动构建的过程。

关键词

人工智能，接口驱动，算法模型，函数构成，智能编程

Copyright © 2019 by author and Hans Publishers Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY).
http://creativecommons.org/licenses/by/4.0/
Open Access

1. 引言

在讨论例子之前，我们先理清下接口的各种模式。第一种是比较直观地，直接相互连接在一起。类似于凹凸的积木块相互插在一起，形状互补的瓦块连接在一起等。这一类可以说是最简单的接口。两个连接物体之间，拥有着各种互补的接口，利用这个接口可以让两个个体连接。第二种就是 2 种个体之间并没有相互互补的接口，通过一些连接物体相互连接着。比如我们用手抓取物体，用夹子夹小东西，这些都可以看成是一种连接。抛开手和夹子。人和东西之间并没有存在着凹凸的接口连接，但通过一种相互的连接模式比如手和夹子，可以让原本不拥有相互连接接口的个体相互连接。那么从上面可以看出，几乎所有物体都是可以连接的。如果 2 个个体有相互互补的接口，就直接连接，如果没有就形成连接他们的中间个体来实现连接[1]。那么回到函数构建这里，如果两个函数之间有自己直接的对应关系的接口，那么就直接连接，如果没有那么就形成可以连接这两函数的整体，来连接这两个函数。下面通过一个简单的例子讲解它实现的过程。

2. 例子说明

这里要实现的例子是：一开始在一个起始文件夹的 get.txt 里面写入文字，比如 weather, day.然后在结果文件夹的 result.txt 里面输出 nice_weather, nice_day。

2.1. 接口的表示

首先第一步是把这些条件转换为相应的接口信息。那么先来看下接口模型的接口表示方法。

F1 = [[["sourceDir","word1"]["word2"],["write_result(sourceDir,word1)"]]]

F2 = [[0, 0, [0,2]]]

F3 = [["-Dir", "-word", "word"]]

F4 = [[["value",["result.txt"],["weather","day"],["weather","day"]],["coordinate",2]]]

其中 F1 表示着编程方法，第 3 个小[]里面的表示编程方法，第 1 个小[]里面表示着这个方法的输入参数，第 2 个小[]里面表示着这个方法的返回参数。

F2 表示着 3 个接口的连接情况。0 表示着接口没有连接，连接了的话就用连接接口的做坐标代替，也就是第 3 个的[0,2]。

F3 表示着接口的类型，正负号代表着接口的输出和输入。F3 重新申明下类型，是因为接口的输入参数和输出参数可能是随意定的，所以统一一下他们的类型作为真正连接的接口。

F4 表示着接口的值，比如 value 后面的第 1 个小[]表示着第一个接口的值。然后 F4 的后面有“**coordinate**”这个参数，这个参数表示着接口的坐标，也就是我们给每个参与连接的接口都排下序，这样就可以调用这个坐标来表示这个接口。比如 F2 中的[0,2]表示着，跟第 0 个参数的第 2 个接口进行连接。

那么上面的例子的起始条件和结束条件就可以表示为

S1 = [[['Dir', 'txt', 'word'],[],[]]]

S2 = [[0,0]]

S3 = [["Dir","word"]]

S4 = [[["value",["get.txt"],["weather"]],["coordinate",0]]]

F1 = [['Dir', 'txt'], ['word'],[]]

F2 = [[0, 0]]

F3 = [["Dir", "-word"]]

F4 = [[["value", ["result.txt"], ["nice_weather"]],["coordinate",1]]]

起始条件，相当于文件路径和里面写的文字都是提供的，因此用正向接口来表示。结束条件的文件路径是提供的，用正向接口表示。但文字是获取的，因此用负向接口来表示。然后 S4 和 F4 里面给他们编号为 0，1。当然这个编号可以系统生成，这里为了比较直观就写出来了。

2.2. 构建连接函数

在表示完成起始条件和结束条件的接口后，就开始调用函数方法库来形成连接的整体。函数方法库，是相应编程方法的库，比如把文字写到 txt 的方法，读取路径下 txt 的名字，获取 txt 里的文字啦等方法。可以说是文字，文件夹，txt 的相关操作读写的方法库。这个可以是写一个公共的方法库供调用，不需要每次都写，就相当于提供基础的积木一样。调用这个方法库后，就按照起始和结束的接口，进行接口的连接。比如 Dir 的接口，就会去寻找-Dir 的接口连接。那么在这样重复多次的寻找和连接，之后就可以形成一个覆盖全部起始接口和结果接口的编程方法整体。当然这样是很随机性的，很有可能并不是我们想要的整体，因此需要对这个整体进一步选择和调节。

2.3. 连接的选择

对上面形成的整体进行调节和选择。那么我们就需要知道选择的方法。只看起始条件和结束条件的话。至少我们可以确定一点是他们之间是有联系的，至于什么样的联系这个其实不用去关心。因为随着编程方法的复杂，他们之间的具体联系也会变得复杂，就相当于什么都不知道。那么首先就要确保他们之间是存在联系的[2]。对于一个编程方法来说，一个输入和它的输出是有联系。那么起始条件和结束条件之间联系，可以通过函数个体之间传递来表示。定义起始接口的连接参数为 **interfac_SF**，结束条件的连接参数为**-interfac_SF**。然后他们的连接参数通过与他们接口连接的编程方法传递下去。比如起始参数跟一个编程方法的输入参数连接，那么这个编程方法的连接接口和它的输出接口全部标注 **interfac_SF** 的连接参数。依次类推，只要跟拥有连接参数接口连接的函数，会把这个连接参数传递下去。像电流一样，从起始接口和结束接口开始向对面传递，如图 1。

而当一个函数同时拥有着 **interfac_SF**，**-interfac_SF** 链接参数的时候，我们就认为这个函数是参与起始条件和结束条件连接的有效编程方法，而其它只有一个连接参数的则认为是无效的编程方法。

通过这些我们可以进一步选择出能够连接起始条件和结束条件的编程方法。如下所示：

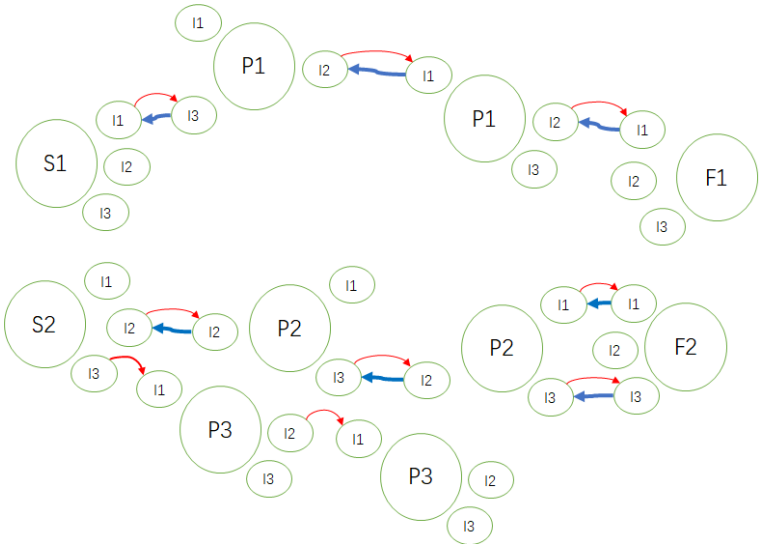

Figure 1. Diagram of connection parameters
图 1. 连接参数示意图

total_array0=[[['Dir', 'txt', 'word'],[],[]], [['Dir', 'txt'], ['word'],[]], [['sourceDir', 'word'], ['word'],['write_result1(Dir,word)']]]

total_array1=[[[2, 0], 0, [2, 1]], [0, 0, [2, 2]], [[0, 0], [0, 2], [1, 2]]]

total_array2=[['Dir', 'txt', 'word'], ['Dir', 'txt', '-word'], ['-Dir', '-word', 'word']]

其中 **write_result1(Dir,word)**这个函数方法的作用是读取 result.txt 和 word，并输出 nice_加 word。也就是我们例子所需要的编程方法。它是符合 **interfac_SF** 的连接参数方法的，它的 sourcDir 接口和 word 接口跟起始接口连接，第 2 个 word 跟结束接口连接。因此它是同时具备两个相反的连接参数的。但是显然它并不是我们想要的，因为它的 sourceDir 不应该跟起始条件的 Dir 连接，它应该跟结束条件的 Dir 连接。它还需要进一步地优化。

2.4. 连接参数的生成

从起始条件和结束条件来说，他们之间的连接，不仅仅是他们之间存在连接就行了。是需要明确的值对应关系。而这个值的对应关系，在没有连接函数的时候，是找不到他们的对应关系的。但有了连接整体后，运行这个整体函数，就有可能找的到他们的对应关系。比如上面的连接，虽然函数连接的 Dir 是错误的，但是它输出的 word 的值是“nice_weather”。这个和结束条件的 word 值是相同的，那么就会生成新的连接参数，如下所示。

["interface","nice_weather",[2,2]],["interface","-nice_weather"]

也就是形成了新的连接关系，连接关系为结束的接口和这个 word 的接口。虽然生成了新的连接参数，但是这个函数的接口是已经连接的。如果把这个函数的接口打破，或者重新连接这个函数的接口，那么可能又会不符合之前的 **interfac_SF** 的连接模式。因此这里选择了更加保守的方式，在这个接口和相关的接口上加入一个 Y 枝函数，如图 2。

也就是加入一个方法，读取一个值，输出 2 个同样的值的方法。把这个接口变成 2 个接口，其中的一个接口还是跟之前的连接，而另外一个接口就参与新的连接参数的连接。那么对结束条件的接口也进行这样处理。然后重新进行编程方法的构成和连接参数的选择。这样就可以形成一个满足起始条件和输出条件的编程方法整体。

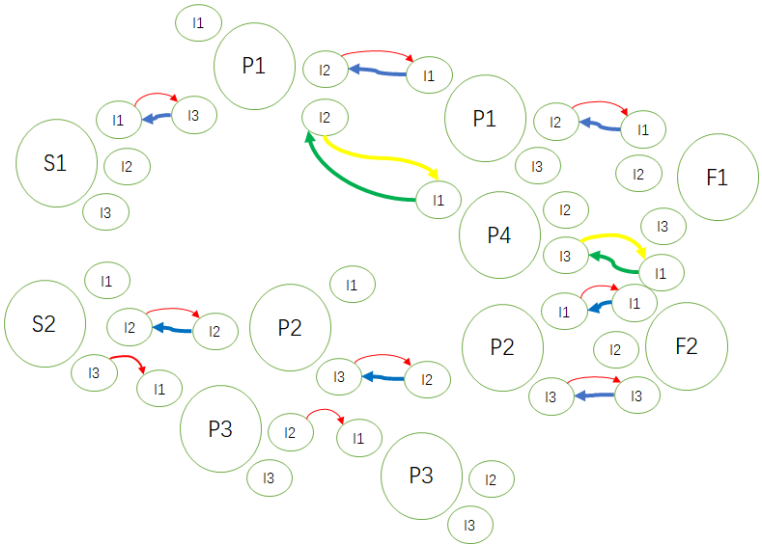

Figure 2. Diagram of Y-branch connection parameters
图 2. Y 枝连接参数示意图

2.5. 训练和调节

当然上面形成的满足条件的编程函数整体是比最简单的模式要臃肿的，也可能产生一些不必要的信息。那么就需要对上面的整体进行训练和进一步地调整。输入不同的符合条件，比如一开始是 weather 和 nice_weather，然后是 day 和 nice_day。来保证这个形成函数整体是有效的。最后进行结构的简化，那些没有值对应的接口，尝试着跟空接口进行连接。比如之前错误的 sourceDir，跟空的路径进行连接。或者把这个函数的接口断开，再重新连接来判断是否有最佳的模式。这样处理之后，就可以避免上面臃肿的函数产生不必要的信息和进一步的简化。当然这些前提也都是建立在整体可用的前提下，不然简化也没有意义了。最后如果这个编程方法整体构建好了之后，也可以用测试集来对他进行测试性能，从而进一步提升性能。

上面例子能够完成的一个原因是，这个函数库里已经包含了结束值的形成的方法。那么假如没有对应结束值的形成的方法话。需要对结束值进行进一步的分析，比如 nice_weather，分析成为 nice_加-weather 的文字接口。就可以根据接口模式，形成同时输出这两个值的函数。也就是如果无法组成输出值，就需要进一步地对值进行分析处理，直到在形成的范围之内[3]。

2.6. 小结

总的来说，通过把起始条件和结束条件的信息进行接口化处理。然后用编程函数方法形成他们连接的整体。并且通过连接参数和新生成的连接参数调整他们形成的整体是可用的。最后进行数据的多次训练和不必要的接口进行空值处理，保证整体的精简程度。过程相当于实现了输入起始的条件，结束的条件，编程方法的库形成一个适应条件的编程方法整体。

3. 与神经网络模型的对比

接口模型智能模型也算是一种智能模型，只是它是以接口为基础的。它也拥有智能的一些特点，能够完成给定的目标，能够通过运行来提高准确度。下面就对他们做一个对比。

输入

神经网络：训练集，结果判断集，测试集

接口模型：起始接口集，结束接口集，测试接口集，接口模块库

这里接口模型多了接口模块库。接口模块相当于神经网络里面的神经元，只是接口模式是通过选择接口模块库里面的方法进行累加的。接口模块库改变不是很大的话，也可以设置成为默认的调用，从而不需要再一开始输入。起始的信息和结束的信息都是可以整理成接口，然后记录再数值里的。

运行

神经网络：通过权值的改变来对数值调整

接口模型：通过接口形成对条件接口进行连接

这里比较类似，只是一个是调整权值[4]，一个是接口的形成。虽然整体看上去不是那么一样，但是从数学上来说，都是相当于给 2 种因素增加可能性。

结果的判断

神经网络：判断是否跟设置的结果一样

接口模型：判断给定的起始接口和结束接口是否连接

接口的连接判断，包括起始接口与结束接口的连接和他们每个值的连接。这里接口模型判断完成后，会生成新的连接参数。

调整

神经网络：根据权值的与目标值的差距，进行梯度调节

接口模型：根据接口的连接参数，继续形成适应这个连接参数的整体

这里的形式有些不一样，神经网络的调节是权值的改变。而接口的模型的调整是接口的形成，因此可能接口模式叫调整生成比较好。当然原则上都是对目标更加接近做的调整。

粗略地对比了下，可以说他们之间还是有着比较类似的地方，虽然他们的基础并不一样。接口模型的作用，也可以类似于神经网络模型一样，给与起始数据和接口的数据，然后形成我们想要的关系。只是一个是完成适应数据之间的关系，一个是形成适应数据的整体[5]。然后他们提高准确度，也都是通过与结果对比，进而进一步地调节。

4. 总结

接口智能模型通过以函数的输入和输出作为接口，把一个函数当成一个接口块进行匹配连接。并且从函数库里，提取和形成能够实现这两个函数之间的连接。然后在运行的过程中，通过运算值的比较。保留正确运算结果的连接块，删除多余和错误的连接块，补上没有值对应的连接关系。从而一步步完善连接的过程，并最终让整体与期望的结果一致。并且说明它也是一种类似于神经网络的智能调节模式。当然上面的接口模型也有很多可以改善的地方。比如对起始值和结果值进一步分析，加大函数库，训练次数提升，寻找和编写核心函数等来对编程方法整体进行优化。在这里只是说明了，这个方法是一个可行性的方法，并且接口智能模型也可以做更多的事情。

参考文献

[1] 林烨斌. 接口模型的算法表示和使用例子[J]. 人工智能与机器人研究, 2018, 7(4): 193-199.

[2] 林烨斌. 以接口作为基础的智能模型[J]. 人工智能与机器人研究, 2018, 7(3): 129-134.

[3] 林烨斌. 如何让筛选一直持续下去[M]. 北京: 中国社会出版社, 2018.

[4] [德]海德格尔. 存在与时间[M]. 陈嘉映, 王庆节, 译. 北京: 三联书店, 1999.

[5] Oz, C. and Leu, M.C. (2011) American Sign Language Word Recognition with a Sensory Glove Using Artificial Neural Networks. *Engineering Applications of Artificial Intelligence*, **24**, 1204-1213. https://doi.org/10.1016/j.engappai.2011.06.015

知网检索的两种方式：

1. 打开知网页面 http://kns.cnki.net/kns/brief/result.aspx?dbPrefix=WWJD
   下拉列表框选择：[ISSN]，输入期刊 ISSN：2326-3415，即可查询
2. 打开知网首页 http://cnki.net/
   左侧“国际文献总库”进入，输入文章标题，即可查询

投稿请点击：http://www.hanspub.org/Submission.aspx
期刊邮箱：airr@hanspub.org