**可视化集成式深度学习开发环境**

目录

**1. 简介1**

**2. 软件的总体结构和核心方法2**

软件的总体结构3

键入章标题(第 3 级)4

核心方法5

键入章标题(第 3 级)6

**3. 案例7**

案例18

案例29

案例310

案例410

案例510

**4. 结论11**

**致谢12**

**参考文献13**

# 1. 简介

[从深度学习切入,讲深度学习工具,然后其局限性,然后我们的工具,最后模仿Caffe论文中那一段写明关于开源社区和基础项目的问题,关于此务必要尽早与Dynamo开发组取得联系,目前这一部分写的太少了.]

如我们所知,机器学习,尤其是深度学习是近年来的研究热点,被认为是非常有前景和应用价值的人工智能方法.经过多年的研究,深度学习已经拥有了很多理论基础和算法,例如自动求导,梯度下降,ADAM等这样的方法成为了研究者们常用的方法.随着大量的高校研究人员和百度,谷歌,微软这样的大企业进入该领域,深度学习迅速积累起大量的基础算法和诸多行之有效的经验,并在对弈,自动驾驶,搜索,图像处理等领域取得了应用效果.随着Caffe和Tensorflow这样的深度学习框架的推出,深度学习的研究和部署也有了实用的解决方案,他们的出现将研究人员从诸如自动求导这样的基本操作的编程实现中解放出来,使他们能够将精力集中于解决更前沿的问题.根据某大赛的统计目前大量的论文中的方法采用了这些框架.

[其中有好几个地方需要引用参考文献,均标蓝色]

尽管我们已经有了这样的工具,但是他们都是控制台程序,虽然借助TensorBoard这样的工具和其他的一些可视化工具powered by python,可以实现一定程度的可视化,但是这些工具本身具有一定的使用门槛,另外,在进行与图像相关的深度学习研究中,对数据进行预处理是很常见的操作,这就需要operator同时必须掌握一定程度的图像处理方法的编程实现.在需要大量实验时,冗余重复的编程工作会拖慢工作进度.

深度学习和与之相关的数据预处理工作具有面向数据流的特点,在较高层次(接近用户)的层次上不适合使用面向对象的编程范式,而适合使用基于函数的过程化编程,这就为使用类似蓝图脚本(in Unreal Engine 4)的Visual Scripting奠定了基础.

本程序是一个完全开源的面向对象的交互式图形软件,设计实现了支持数据读取,预处理,深度学习模型的搭建和可视化等功能的图形化编程环境,可以使用”拖拽”方式进行操作,并支持用户扩展.程序使用C#和python开发,在构思,研究数据预处理,调整网络模型参数和观察网络行为等方面应用本程序,只需要编写少量代码甚至完全不需要编程,可以降低对编程技能的需求,提高工作效率.

程序是在AutoDesk公司Dynamo项目(which is design for参数化建模)的基础上编写的,without its Community,本程序will be nothing,在此感谢Dynamo项目的开发组和所有在(Dynamo连接)上贡献过代码的Contributors.

[文章结构]

本文的其余部分组织如下:第二部分介绍了软件的整体结构,以及特色.第三部分介绍了几个深度学习的例子,以说明该软件对深度学习模型的表现能力并展示可视化表示,第四部分是结论.

(

重点词汇辨析:

1.”图形化编程”,是否会和Visual C++这种所谓的’’可视化编程’’混淆?

2.参考词汇: Visual Scripting/ node-based interface

3.汉语的版本中使用”图形化编程”

)

# 2. 软件的总体结构以及核心方法

”程序” 是一个图形界面软件,能以图形化的方式进行深度学习模型的设计和调试.

## 2.1软件的总体结构

[基于两张图来介绍总体结构,一张数据流图,一张模块示意图]

[数据流图主要体现蓝图->中间代码->程序代码的流程和蓝图直接运行的流程]

[模块示意图主要体现模块之间的依赖关系,这块还比较乱,现在画不出来]

[介绍几个重要模块的作用]

[说一下MVVM模式,一两句话概括优点,然后分别介绍程序里面的M/V/VM都是由哪些类扮演的,以及为什么要这样做]

[要提及友好的用户界面,并配图]

## 2.2 核心方法

(这个核心方法是什么现在还没确定,候选的有两个:图形化编程语言,代码转换技术,

其中图形编程语言这块如果作为核心方法,主要从模块化设计,用户扩展接口,内置代码段这三个方面来说,

如果是代码转换技术作为核心方法,就是从编译的角度来讲,词法分析,语法分析,语义分析,序列化,解序列化,比较零碎

目前的状况是

图形编程语言这块,基础的计算和函数都有,但是针对深度学习的算法大模块还没做,

用户扩展接口是基于自定义dll的编程接口,加载的稳定性不太好,如果用户自定义的模块想调用绘图方面的功能,还不行,做基础计算和复杂计算没问题,

代码转换现在是能做到序列化和解序列化,但是是基于json的,之前想过基于python做,但是.net自带的Ironpython不能扩展tensorflow模块,所以想法是做成离线转换,设计好的图直接转换为离线的python,然后外部调用tensorflow等模块,但是这样一来,实时调试就很不好做,因为如果是外部进程,只能在生成代码的时候插入一些额外的代码来实现实时进程通讯,这个没有尝试过,不知道什么效果,这个点是整个工程中最难的难点,如何实现实时调试,这是一件很重要的事情,我觉得重要的不是怎么写出来这个东西,而是想明白以一种什么样的形式去实时调试才能更好更有用)

而且我觉得必须要有一定的数学描述才好,那么什么地方能加上数学描述呢?

核心方法可以给一个流程图,例如代码转换的流程,但是图形化编程语言可以给渲染流程.

介绍核心方法的各个步骤的时候,应该说明这一步是总体流程中数据流图或者模块依赖图中的哪一个模块.

# 3.案例

案例1:使用蓝图进行简单计算,要设计到文件IO和标准输入输出IO

案例2:使用蓝图进行图像处理,实现某篇论文中的图像预处理方法.

案例:3使用蓝图进行神经网络设计,挑选具有代表性的例如AlexNet,残差网络,SSD,RNN等进行设计表示并导出代码进行运行测试,并与直接编写的对应的网络进行性能对比和代码量对比,如果可能,演示从代码转换成蓝图的能力.并说明限制条件

案例4:使用蓝图绘制曲线,曲面,查看卷积核等,此处可以是用一个简单的神经网络或者简单计算进行,

案例5:进行用户自定义扩展,目前有两种扩展:用节点组合成一个复杂节点/基于dll的编扩展.

[所有的案例配截图,简要说明操作,]

# 4.结论

本文介绍了图形化编程方法在深度学习模型的设计和调试中的应用,有效的使用了代码转换和MVVM设计模式,设计并实现了一种基于节点的图形化编程方法,使用了几个特定的示例来演示功能,所提出的的用户界面已被证明是直观且灵活的,可以通过节点的连接完成深度学习模型的设计,并使用调试工具进行模型的调试,为快速简洁的深度学习应用开发提供了一个有效的工具,他还提供了扩展接口,能够被用户简单的定制为更适合自身开发需要的灵活而强大的平台.

# 致谢

在文章结尾部分的单独部分中，在引用之前整理确认，因此，不要将它们包括在标题页上，作为标题或其他的脚注。这里列出那些在研究期间提供帮助的人（例如，提供语言帮助，写作帮助或者阅读文章的证明等）。  
以这种标准方式列出资金来源：

资助：这项工作得到了国立卫生研究院的资助[grant number xxxx，yyyy]; 西雅图比尔和梅林达·盖茨基金会[授权号码zzzz]; 和美国和平研究所[grant number aaaa]。

没有必要对计划或赠款和奖励的类型进行详细的说明。当拨款来自大学，学院或其他研究机构可获得的整笔拨款或其他资源时，请提供提供资助的机构或组织的名称。

如果没有为研究提供资金，请包括以下句子：

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

# 参考文献

2018年6月2日09:35:30更新:

1. 目前已出现了一款在线软件TensorEditor, <https://www.tensoreditor.com/>,但是他是闭源的,但是他真的能生成tensorflow代码.

2. 可以加一个例子,肺癌检测中形态学处理那8步,引用那篇论文:grt123

3. 要说明程序是基于AutoDesk Dynamo开源软件编写的,其中核心部件是DesignScript.

4. 可以说一下,在深度学习中,进行数据预处理经常是比较麻烦但是也比较重要的,程序的存在可以在构思期间让开发者专注于思路和工作流的搭建而不是具体的编程操作中.

6. Python的绘图功能能不能想办法集成进来

7.考虑加载已经训练好的模型,然后导出某些层的特征并可视化