**可视化集成式深度学习开发环境**

目录

**1. 简介1**

**2.** **相关工作2**

**3. Node-Based图形化编程模型7**

**4. Untouchable扩展接口11**

**5. 应用案例11**

**5. 讨论与结论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,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.

[参考文献]

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.

本程序的设计目的是帮助进行深度学习研究的工作人员在构思和设计实验时进行快速原型设计和方法初期验证,通过图形化编程,使用者能够在工作初期从重复性的工作中解脱出来,将精力集中在自己的核心工作上,而不必被工具的使用和编程问题所困扰,同时,其他对深度学习和图像处理感兴趣的用户也能通过本程序进行一些简单的尝试.

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. 相关工作

Caffe,Tensorflow,的简介,并说明他们的使用特点.根据统计,目前Tensorflow等深度学框架的使用率非常高.深度学习模型带有模块化的特征,无论多么复杂的深度学习模型,都能分解为卷积层,全连接层,激活函数,损失函数,正则化方法等block,对2013年以来的比较有代表性的深度学习论文的统计和研究发现,在对深度学习方法本身的研究中,除了对已知方法的拆解组合和微调外,还有很多工作提出了新的结构和方法,例如残差网络首次应用跨层链接和残差学习,…首次应用动量方法解决局部最优解等.

数据预处理方面,由于深度学习需要使用大量标注数据,对数据进行预处理和简单分析是开展工作的前提.图像处理是数据预处理中最常用的手段.常用的针对图像进行预处理有分割,缩放,旋转,平移,形态学处理,边缘检测等方法或者这些方法的组合.对论文公开的源代码和深度学习相关的开源项目的不完全统计表明,OpenCV和python是最常用的图像预处理手段.

封装度和易用性的矛盾体现在深度学习工具和图像处理上,为了便于使用并降低学习成本,就要提高封装度,向用户提供高级API,但是这样系统的可扩展性就会下降;如果要向用户提供足够多的选择,那么用户将不得不面对更复杂的编程问题和更高的学习成本.

图形化编程方面,麻省理工学院开发的Scratch可以进行简单的面相过程编程,他的building block对应的是高级语言中的基本语句结构,如变量,语句块等,主要面向青少年教育领域; ArduBlock是开源硬件Arduino的第三方图形化编程工具,他的building block类似于Scratch,添加了一些与硬件对应的例如引脚查询等功能.和这两种类似的图形化编程方法类似于对编程语言的直接翻译,没有提供面向对象功能,且IO和扩展能力有限;BluePrint是虚幻4游戏引擎中的脚本系统,该系统能够实现游戏设计中几乎全部的脚本编程需求,蓝图的building block对应的是高级语言中的方法,变量和对象,其中内置block封装了引擎的API和常用功能,并提供了扩展接口,用户可以使用C++编写自己的building block.蓝图是设计比较完善的图形化编程工具,但是他依托于虚幻,无法应用到游戏以外的其他领域.

TensorEditor(<https://www.tensoreditor.com/>),是一个闭源的面向TensorFlow的第三方图形化工具, TensorEditor的building block完全是对深度学习模型中的各种子结构进行封装,不能进行其他计算,而且无法进行用户扩展,只能使用其中内置好的功能.

[统计数据有参考文献,是一个调查机构写的]

# 3. Node-based 图形化编程模型

由于本文主要面相深度学习和图像数据处理,基于这些任务是面相数据(data based)的这一事实,使用了Node-based”图形化编程模型.

3.1 Node是对方法,变量,对象的封装,

一个Simple Node由Name,Input Port,OutPut Port组成,对于封装只读变量的Node,将没有输入接口,对于封装了静态方法的Node,他的输入和输出接口对应的是方法的输入输出接口,而对于封装了非静态方法的Node,将会多一个接口用于连接调用方法所必须的对象(将会在4.2中详细说明),大多数Node都是属于某个类的方法,这些方法将会被编译成动态连接库,并由目录服务组件在程序启动时进行加载(将在3.5和4.1中详细说明).[配一张导出图即可]

3.2 Node抽象,有向图排列,绘图接口

为了便于存储和绘制Node,程序设计中对Node进行了多层抽象,[配类图],分别是图,层,边,节点,其中边和节点相连组成图,图分为多个层,分层的目的是为了进行有向图排列.

采用杉山算法进行有向图的排列.[在此简述有向图排列,并配效果图]

实现drawable接口的Node可以连接到watch Node上进行绘图,实现接口方法可以通过设置顶点位置和定点着色的方式,使用类似D3D的语法进行简单几何体和二维图形的绘制,同时,也可以通过图像处理组件直接进行面向矩阵的绘图和显示.

3.3 Node的执行:AST和DesignScript

图(3.2中所述)将被描述成一个AST(Abstract Syntax Tree)进行后续分析,完成编译的前端工作,编译的后端工作由DesignScript执行.[此处为关键点,需要大量细节,配流程图]

3.4 序列化存储

由Node构成的图比较直观,但是也对拷贝和交流造成了一定限制,所以必须有一种有效的存储形式,程序使用json作为存储格式,通过图遍历,将图序列化,并可以用相同的方法解序列化为原图.[这一段有点太弱了,考虑不要写]

3.5 基于Chromium引擎的渲染

由于程序采用了用户自定义扩展的设计,为了实现对外接口的简洁高效,使用MVVC设计模式,将Node的实现和表示分割,加载封装在动态链接库中的Node时,根据基本UI框架实时生成对应于每个Node的UI,采用基于Chromium引擎的渲染方法,生成的UI通过本地静态服务管理,并依据方法和程序集签名生成URL,表示层在需要绘制UI时,构造出对应的URL向静态服务请求资源,并将返回的资源渲染到UI上,对于用户来讲,如果自定义了Simple Node,不需要指定他的UI是什么样的,UI渲染对于用户是不可见的,但是系统依旧留出了外挂UI的接口,为用户编写带有个性化UI的Node提供基础.同时,静态服务还提供了用于调试的访问接口.[需要配流程图]

# 4.Untouchable扩展接口

程序提供了扩展接口以便用户可以扩展自己设计的新Node.从加载机制和接口设计两方面确保用户可以在完全不了解程序内部细节的情况下设计自定义的Node并使用

4.1 基于动态链接的运行时加载

为了能加载用户自定义的Node,并为用户扩展提供足够的灵活性,本程序的所有Node都是以动态链接库的形式存在,在程序启动时,启动服务和路径管理服务会找到核心动态来接库和存放在插件目录下的全部用户自定义DLL,分析其中的命名空间,类名和方法名,并根据标注(将在4.2中说明)和可访问性决定是否将方法注册为Node.

[这里需要详细说明COM和怎么从DLL中找到这些信息]

4.2 Untouchable接口

程序采用的Node加载方式和前后端分离的渲染方式,使得Node类可以是一个普通的类,不需要继承自某些特定的类或者实现一些特定的方法(当然如果要使用绘图功能,必须使用drawable接口),甚至没有任何特殊要求,就可以将一个类中的方法和属性注册成Node,这种设计的出发点是可以尽量满足不同用户的不同扩展需求(例如有些用户可能想在他的自扩展Node中使用其他的库或者开源项目,由于我们不要求作为Node的类有什么特别之处,所以也就不会对使用其他库造成任何障碍,实际上,编写Node类和编写其他程序中的普通类没有任何区别),同时,使用这项功能并不需要用户阅读一个长达数十页的API文档.

尽管使用Untouchable接口看起来如此方便,但是这其中仍然存在一些问题,例如我们可能并不想把一个类中的全部方法和属性字段注册成Node(那些我们不想看到的东西可能是实现内部操作的方法或者中间变量等,这在编程中很常见),所以我们还是对用户扩展做出了一些约定:

① 程序集中只有public class是可见的,同理,一切public的方法和属性字段都是可见的.

② 显然,protected和private的,external的都不会被注册成Node

只使用上述约定还是不足以描述所有的可见性,例如,由于某种原因,我们需要一个方法是public的,但是却不想在程序中被注册成Node,所以除此之外,我们还约定了一些属性(注意这个属性是Attribute,C#中的属性有好几种含义)以供用户标注方法和字段的可见性,同时,还提供了一些属性用来声明带有多个返回值的函数和带有默认值的函数.[此处参考Expressior-Addin API]

需要注意的是,尽管我们定义了一些属性,但是用户在完全不使用这些属性的情况下也能编写出可正常导入系统的扩展动态链接库.另外,由于本程序是开源的,同时为也为能够获取并构建源代码的用户提供了功能更强大但是也更复杂的内置Node扩展接口,使用这些接口,用户能够扩展系统的内置Node,这些Node能够访问系统内部的更多信息,并能随着程序启动自动加载,还能编制自定义的UI.[这个API很麻烦,看情况写点]

# 5.应用案例

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,]

# 6.讨论与结论

本文介绍了一种Node Based的可视化编程方法,并基于Autodesk Dynamo实现了使用这种方法的开源系统,该系统能够以节点的形式封装方法,变量和对象,能够帮助进行深度学习研究的工作人员在构思和设计实验时进行快速原型设计和方法初期验证,通过图形化编程,使用者能够在工作初期从重复性的工作中解脱出来,将精力集中在自己的核心工作上,而不必被工具的使用和编程问题所困扰;同时,系统提供了Untouchable扩展接口,用户可以通过简单的API编写自定义的Node,扩展系统的功能.

尽管我们的系统提供了友好的用户界面和扩展接口,但是仍然需要一定的学习时间和成本,对于不熟悉强类型语言的用户,编写自定义功能可能存在一些障碍.

另外,对于非常复杂的问题,用Node表示的程序可能会“很大”以至于很难在一个屏幕上观察完整的思路,过于复杂的图可能非但不能起到直观的效果,反而会使得问题更加令人迷惑,这同时也是各种图形化编程方法的通病.

由于我们关注的是快速原型设计和方法初期验证,所以执行效率问题并不是我们的首要考虑,在执行一些计算量很大的操作时,有可能会耗费比直接编程更多的时间.

在此后的工作中,我们将就提高运行效率和将图转换成代码进行深入研究.

# 致谢

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

2018年6月11日10:43:55更新

【根据Apache 2.0开源协议,我可以修改源代码,但是必须注明我对哪些文件进行了修改,发布时必须附带Apache 2.0 Licence,必须保留原有的商标,专利和版权声明,我可以在任何需要的时候修改源代码并将其用于其他开源项目或者商品进行发布甚至销售】