

本科毕业论文



|  |  |
| --- | --- |
| 论文题目： | 神经网络与大规模机器学习 |
| 院 系： | 计算机科学技术学院 |
| 专 业： | 计算机科学与技术 |
| 姓 名： | 戴涵俊 |
| 学 号： | 10300240030 |
| 指导教师： | 张军平 |

2014年 6月1日

神经网络与大规模机器学习

摘要：

关键词：

**Abstract:**

**Keyword:**

Table of Contents

大规模神经网络平台IECABrain 6

神经网络综述 7

神经网络相关工作 7

现有神经网络平台 7

平台概览 8

动机 8

现有问题 8

需求 8

平台架构 10

架构预览 10

核心设计 12

平台特点 12

IECAMatrix矩阵库 13

动机 13

现有矩阵库 13

IECAMatrix特点 13

数据结构 13

模板框架 13

表达式引擎 13

表达式解析引擎 13

表达式执行引擎 13

性能对比 13

训练算法 14

反向传播 14

梯度下降 14

批量训练 14

大规模训练方法 14

多模型并发 14

分布式训练 14

softmax近似 14

神经网络训练技巧 14

动量(momentum) 14

随机丢弃(dropout) 14

扩展功能 14

卷积神经网络 14

栈式自动编码器 14

用递归神经网络进行搜索广告点击预测 15

神经网络与多义词的向量嵌入 16

# 大规模神经网络平台IECABrain

## 神经网络综述

### 神经网络相关工作

### 现有神经网络平台

## 平台概览

这一节主要介绍大规模神经网络平台的高层次信息。首先将阐述我们构建IECABrain的动机，通过对现今学术界和工业界的神经网络平台进行回顾，我们会发现他们具有各种不同的问题，而无法满足我们的需求；接下来，将概括性介绍我们开发的神经网络平台的设计架构和实线理念，具体的实现方法和算法将留在后面的章节中具体阐述；最后，我们总结新构建的平台的特点。

### 动机

构建一个通用的神经网络平台是一件需要大量精力与时间的工作。为了能够支持我们后续的研究工作，我们有必要构建适合自己的神经网络平台——IECABrain。以下将通过横向对比现有的神经网络平台、以及结合自身研究课题需求，阐述IECABrain诞生的意义，以及其所要完成的任务。

#### 现有问题

在当今的学术界和工业界，大数据和机器学习已经成为了热门关键词。而大数据带来的不仅仅是机遇，还有对现有机器学习算法以及软件开发人员的巨大挑战。神经网络在最近兴起的深度学习中，又再次引起了一波研究热潮。不过，由于多层神经网络的训练并不是一个凸优化问题，因而要训练好一个神经网络，即防止得到较差的局部最优解，需要很多的技巧与经验。

通过上文中对现有的神经网络平台的介绍，我们对其特点有了初步的掌握。在开发我们自己的平台之前，我们对这些现有的平台进行了详细的调查。通过安装并运行一些标准数据集，我们发现这些平台有如下的一些问题：

1. **Matlab**

Matlab一个主要的问题是非免费。它是个收费软件，因而要部署在集群上，就需要有多个许可证，而其收费也是非常高昂。Matlab程序编写容易所带来的代价便是效率的降低，尤其是和优化过的C++代码的效率相比。另外，在Matlab的神经网络工具包中，其包装的完整性也使得我们并不方便进行深层次的定制，而用其进行神经网络的研究，必须要研究、改变其底层结构和实现。这些原因使得Matlab不满足我们的研究需求。

1. **Theano**

虽然Theano可以编译成C代码执行，也可以方便地利用GPU运算，不过效率上还是不如优化过的C代码。另外，Theano更多的是可以看做一个符号运算的执行器，相对于神经网络来说，负责的是更底层的运算。考虑到后续开发和维护的代价，用高效的C++更为合适。

1. **Pybrain**

该项目已经多年没有更新了。虽然是一个完整的神经网络库，并且在设计方面有其独到之处，但是效率问题（并不支持MKL以及CUDA优化）以及扩展能力的不足（不能进行分布式运算），使得其离我们的需求还很远。

1. **FANN**

该平台虽然完全用C代码编写，但是由于没有用并行优化，也难以扩展到GPU平台和分布式框架，所以更多的适合于在单节点上小数据集的调试。另外，神经网络训练算法也固定在其设计中，导致批量梯度下降、随机梯度下降等方法不适用。

1. OpenCV
2. **Caffe**

该平台更多的偏向于使用，而并不是二次开发。一方面，它并不支持分布式的神经网络训练；另外，虽然在Caffe平台下配置一个神经网络，只需要写少量的标记语言，但是，这也限制了用户能够修改的部分。对比我们对平台的要求：适于应对研究工作中各种修改、有分布式计算的能力，Caffe显然也不是一个合适我们当前任务的选择。

#### 需求

为了支持神经网络相关的研究工作，一个强大而高效的神经网络平台是必须的。具体而言，我们当前主要有两方面的任务：1）计算广告学中的搜索广告点击预测问题；2）自然语言处理中的词向量空间嵌入问题。以下分别针对这两个任务阐述其难点以及对神经网络平台的需求。

**任务一、计算广告学中的搜索广告点击预测问题**

作为新兴的分支学科，计算广告学涉及到了大量研究内容，如机器学习、信息检索等。而作为一个典型的计算广告学的应用，搜索广告占有了重要的地位。通过在搜索引擎中投放广告，进而通过用户的点击进行收费，这种模式已经成为了大量的互联网公司的重要盈利手段。而广告平台对广告主收费的重要依据就是广告的点击，因此，投放用户最可能点击的广告，是广告投放平台和广告主都关心的问题。而广告点击预测最为显著的特点，就是——数据量巨大。平均而言，主流的搜索引擎（如Google，百度，Bing等）每日的搜索量都在10亿级别。每次用户的搜索，系统都会投放一定数量的广告。如果我们将过去一个月的用户搜索广告点击行为数据收集起来，用机器学习的方法去训练，将是对所用的算法以及系统一个巨大的挑战。

另外，用户的行为也在随着时间变化。而且广告点击的行为不仅仅与用户相关，还与当前的热点事件、节假日等有密切关系。因此，训练出来的模型也需要经常更新。一般可以用在线学习的方式，也可以每两到三天更新一次。

我们不难从上面总结出，在这个问题中，如果用神经网络模型作为机器学习的工具，那么就需要训练网络的平台具有如下的特点：

1. 平台具有很好的扩展能力。这种扩展能力主要体现在当计算资源变多了之后，我们能够有办法利用大量的服务器去提升计算能力，从而在短时间内训练模型，以满足计算广告的问题中关于时效性的需求。因而，该平台能够有分布式计算的能力。
2. 在计算广告问题中，常常会需要用bag-of-words形式的向量，即所谓的ID特征。考虑到作为特征的对象的规模常常非常大，例如广告的数量、用户的数量。这些将导致一个问题——由于单个节点的内存大小有限，我们的模型甚至无法在单个运算节点（服务器）上面存下来！所以，平台需要有能力将模型的参数也分布式存储。

**任务二、自然语言处理中的词向量空间嵌入问题**

在传统的文本处理方法中，常常采用向量空间模型（Vector Space Model, VSM）。这种模型将文档表示为一个向量，而向量的维度就是词表的大小。向量中的每个元素对应了一个词，可以是这个词的词频，也可以是TF-IDF等特征。这种表达方式没有对单词进行细致的描述。

所谓的词向量空间嵌入，就是将单词也用向量来表示，进而，可以通过在向量空间中的操作，来寻找词与词之间的关系。譬如，我们可以用词向量的距离，来表征两个词的相近程度。在后文中，我们将详细阐述词向量空间嵌入的问题。

词向量嵌入有很多种方法。用神经网络训练的自然语言模型（Neural Network Language Model, NNLM）是最近较为流行的方法。当然，为了训练出较好的自然语言模型，大量的训练语料是必不可少的。一个大规模的语料库，除了篇幅巨大（例如数百GB的文学资料）之外，包含的词的种类也巨大（数以亿计的单词、词组、特殊意义的字符数字组合，等等）。所以，这种任务同样需要平台具有以下的能力：

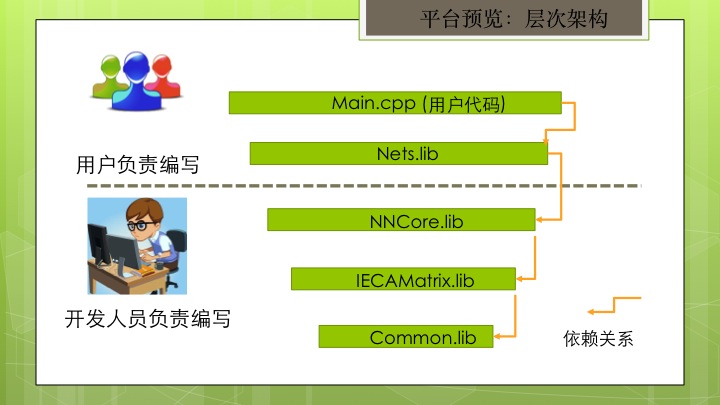
1. 平台能应对规模巨大的神经网络。该需求和任务一中的需求b类似，即需要能够分布式存储巨大的模型。
2. 平台需要高效。高效不仅仅需要计算上速度快，还要求其带来的额外冗余开销小。在神经网络自然语言模型中，单个词样本的训练计算代价非常小，但是由于训练样本数量巨大，因此总的代价也会很大。如果在单个词样本的训练上有多余的额外开销，那么整个训练过程的开销就非常可观了。另外，训练语言模型常常需要多次迭代训练一个语料库，这些问题都导致了对平台性能的高要求。

### 平台架构

在这一小节，我们将首先从大规模神经网络平台IECABrain的层次架构开始，介绍平台中各个部分的依赖关系；接下来会按照层次，从底层向高层介绍IECABrain，首先我们用一个小节介绍我们自主开发的矩阵运算库，这个矩阵库支撑了整个平台的核心运算，接着，便是核心部分的设计介绍，这包括了网络核心的各个部件，以及其组织逻辑。具体的大规模学习算法，以及神经网络的训练优化方法等，将留到第四章节详细阐述。

#### 架构预览

像众多软件的设计方法一样，IECABrain也采用多层次架构进行设计。具体来说，我们将架构分为五层，如下图：



可以看到，为了最大限度地让我们的平台易于扩展，需要用户编写少量的代码——也就是说，我们的平台不是一个傻瓜式的黑盒，只需要输入参数和数据即可，这不是我们的初衷。我们的目的是构建一个适合进行关于神经网络和深度学习的深层次研究的平台。而这就需要平台易于扩展，同时也要易于维护。

除了最顶层的用户代码，其余几层都被编译成了静态链接库。相较于将所有层次构建在一个项目编译输出，这种设计能够减少开发过程中编译的代价。同时，也易于之后的维护。

下面，我们由底层到顶层，分别介绍每个层次所负责的功能。

1. **Common.lib**

Common是通用的库，顾名思义，这个是最底层的，与神经网络几乎无关的库。该库主要包含许多的工具类，后文中将会直接使用到其中的一些类，由于其实现方法不是本文讨论的重点，所以统一在这里进行列举说明。主要用到的类有：

1. **Dictionary**

一种按照键-值对组织的数据结构。通过可以计算哈希编码的键来索引对应的值。核心部分是一个哈希表。这种实现方式平均情况下会比平衡树来的更高效，当然，是以一定空间代价作为交换的。

1. **HuffmanEncoder**

一个自动进行哈弗曼编码的工具。通过给定每个编码对象的字符串，以及该对象的权重（譬如单词的词频），该工具能够自动构建哈弗曼树，同时能够知道每个编码对象的编码、其在哈弗曼树上的路径等重要信息。

1. **CMultinomialSampler**

一个随机采样工具。该工具能够进行多项分布的采样。通过给定初始的项数和每项对应的概率，该工具能够很高效地对给定的多项分布进行采样，同时，其基于c++的伪随机数，方便调试。

1. **Log**

一个记录日志的工具。相比较工业界流行的glog之类的工具，这个工具更加轻量级，并且易于部署、管理。

更多的类和每个类各自的实现方式，请参考源代码。在后文中，这一个库就不会再进行更多的介绍。

1. **IECAMatrix.lib**

神经网络中，绝大多数操作都可以转化成矩阵运算。例如，两层网络之间的前向传播，就可以用一个矩阵乘法来表达。将这层运算用矩阵封装后，在更高层次，我们就只需要关注神经网络算法的逻辑，而不需要花费更多精力在具体运算实现上面了。

因此，我们有必要建立一个矩阵库，专门负责这些操作。IECAMatrix就是这样的一个通用矩阵库。之所以称之为通用，是因为它不仅可以服务于IECABrain这个神经网络平台，还可以用于其他的项目中。

然而，这个矩阵库有两个最主要的特点：统一性和优雅性。我们将在下一章节，**IECAMatrix矩阵库**中，进一步介绍。

1. **NNCore.lib**

该库包含了我们神经网络平台的核心部件。在设计的逻辑中，两个最主要的类——层（Layer）和连接（Connection）将在这一个层面实现。

除了构成神经网络的层和连接这些部件之外，还有配套的许多其他部件，例如流式数据读取、分布式计算、多线程并行，以及大规模神经网络训练的优化方法等，都会在这一层面实现。因而，这个库是整个神经网络平台的核心。

当然，这里不仅仅有一系列单独的部件，还涉及到这些部件如何协同工作的逻辑。这些内容将在下面的章节**核心设计**中进一步介绍。

1. **Nets.lib**

从这里开始，就要涉及到用户编写的代码了。有了NNCore这个核心部件，构建各种神经网络，就只需要组装这些部件了。而不同的神经网络的差别，大多也是在结构上、层和连接表现行为上的差异，因此，留给用户写的代码并不是很多。在**核心设计**这一节中，也会展示一个通用的抽象神经网络，即，通过配置不同的部件，同一个神经网络类可以展现完全不同的功能。

1. **Main.cpp**

这里完全是用户自己编写的代码。主要涉及到用户的个性化配置，例如矩阵运算用CPU还是GPU实现，数据文件存放在哪里，训练神经网络的时候，各个可以调控的参数该如何设置，等等。

至此，我们按照层次结构从底层到高层，完整介绍了我们的平台概况。

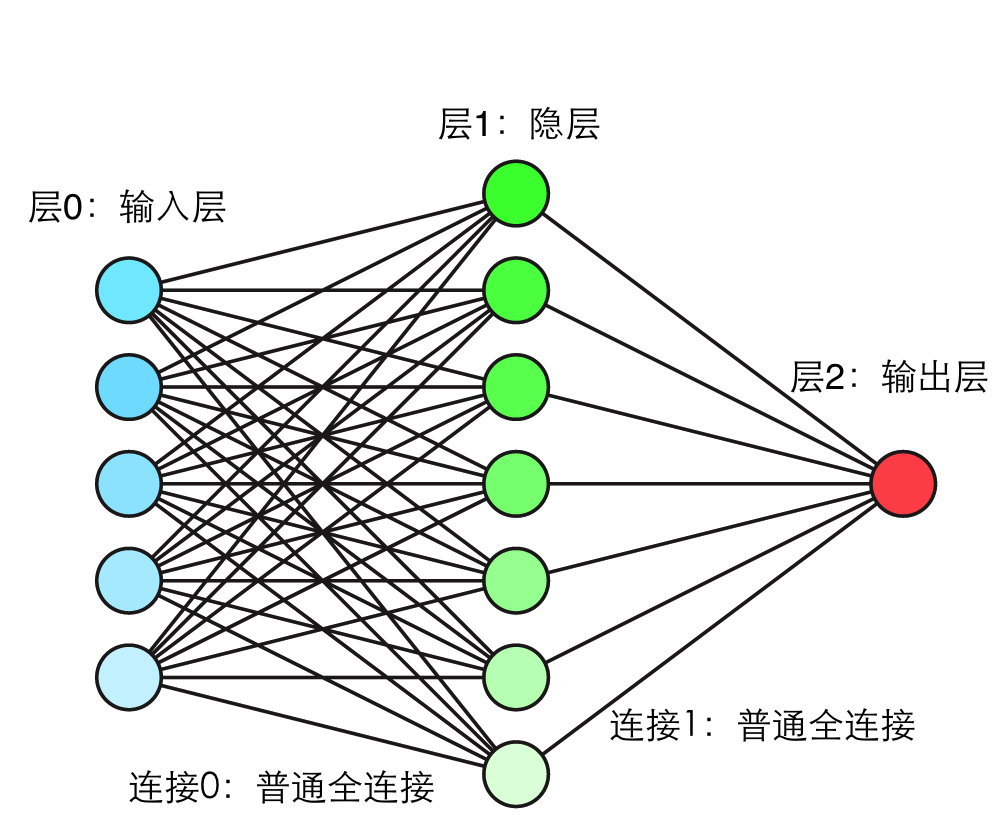
#### 核心设计

在这一节，我们主要介绍上文中提到的NNCore.lib，也就是神经网络平台的第三层。通过对各个部件的功能以及对其组织方式进行介绍，我们将从整体上把握整个平台的设计理念。在第三章**训练算法**中，我们会详细说明各个部件的实现方法，以及在各种标准数据集上，与现有平台的性能对比。

我们首先分门别类介绍各个部件。接下来，我们按照各个部分负责的逻辑上的功能，用示意图结合文字的形式，介绍它们组合运作的方式。

##### 层与连接

神经网络结构中，层和连接是最关键的两个组成元素。下图展示了一个普通的三层全连接神经网络，在这图上，我们定义了层和连接这两个概念。



上图中，一共有三层（层0~层2），两个连接（连接0和连接1）。因此，在我们的定义中，一层对应了一个神经元的集合，而一个连接，对应了连接两层之间的边。以上是数据结构层面的定义，而神经元和边连接又可以有多种表现行为，比如神经元的激活方式、边的前向传播方式以及反向更新方式等，都可以有变化。因此，我们将这两个元素——层和连接，用以下的数据结构来表示。

* **层**

如上所述，层是一个神经元集合的组织方式。而层的标号（0，1，2，等等）并不一定需要有顺序关系（即层1不一定代表着在层2之前），而仅仅是用来作为层的一个名字，用来唯一标识层。这种定义方式给我们平台带来了很大的灵活性——我们可以自己定义层的拓扑顺序。

* **连接**

### 平台特点

## IECAMatrix矩阵库

### 动机

#### 现有矩阵库

#### IECAMatrix特点

### 数据结构

### 模板框架

#### 表达式引擎

#### 表达式解析引擎

#### 表达式执行引擎

##### CPU引擎

##### GPU引擎

### 性能对比

## 训练算法

### 反向传播

#### 梯度下降

#### 批量训练

### 大规模训练方法

#### 多模型并发

#### 分布式训练

#### softmax近似

### 神经网络训练技巧

#### 动量(momentum)

#### 随机丢弃(dropout)

### 扩展功能

#### 卷积神经网络

#### 栈式自动编码器

# 用递归神经网络进行搜索广告点击预测

# 神经网络与多义词的向量嵌入