基于Torch平台的神经网络压缩

研究与应用

软件项目计划书

**编写人：17TeamG**

**2016年3月15日星期三**

目 录

[1. 引言 1](#_Toc477511699)

[1.1 背景 1](#_Toc477511700)

[1.2 定义 1](#_Toc477511701)

[1.3 参考资料 2](#_Toc477511702)

[1.4 标准、条约和约定 2](#_Toc477511703)

[2. 项目概述 2](#_Toc477511704)

[2.1 项目目标 2](#_Toc477511705)

[2.2 工作内容 2](#_Toc477511706)

[2.3 应交付成果 3](#_Toc477511707)

[2.4 项目验收方式和依据 3](#_Toc477511708)

[3. 项目团队组织 4](#_Toc477511709)

[3.1 组织结构 4](#_Toc477511710)

[3.2 人员组成 4](#_Toc477511711)

[3.3 协作与沟通 4](#_Toc477511712)

[4. 实施计划 4](#_Toc477511713)

[4.1 计划分析 4](#_Toc477511714)

[1) 安装Torch 4](#_Toc477511715)

[2) 需求分析 5](#_Toc477511716)

[3) 测试分析 5](#_Toc477511717)

[4) 功能实现 5](#_Toc477511718)

[5) 其他软件管理工具学习 5](#_Toc477511719)

[4.2 总体进度计划 5](#_Toc477511720)

[4.3 项目控制计划 7](#_Toc477511721)

[5. 关键问题 7](#_Toc477511722)

[6. 专题计划要点 7](#_Toc477511723)

# 引言

## 背景

近年来，人工神经网络因其远超过传统机器学习算法的表现，成为国内外研究的主流。人工神经网络（Artificial Neural Networks，简写为ANNs）也简称为神经网络（NNs）或称作连接模型（Connection Model），它是一种模仿动物神经网络行为特征，进行分布式并行信息处理的算法数学模型。这种网络依靠系统的复杂程度，通过调整内部大量节点之间相互连接的关系，从而达到处理信息的目的。

神经网络功能强大。但是，其巨大的存储和计算代价也使得其实用性特别是在移动设备上的应用受到了很大限制。因此，降低大型神经网络其存储和计算消耗，使得其可以在移动设备上得以运行，即要实现“深度压缩”，具有十分重要的现实意义。

进行神经网络的研究平台有OpenCV、Matlab等全能工具，但深度学习的未来主要是来自像Caffe或Torch那样有自给自足的软件包。

Torch的目标是在建立科学算法的同时，要有最大的灵活性和速度，而这一过程非常简单。Torch拥有一个大社区驱动包的生态系统，涉及机器学习、计算机视觉、信号处理、并行处理、图像、视频、音频和网络等，并建立在Lua社区基础之上。

Torch的核心是流行的神经网络和优化库，它们易于使用，同时在实现复杂的神经网络拓扑结构时具有最大的灵活性。可以建立任意的神经网络图，并在CPUs和GPUs上有效地并行化。

## 定义

关于Torch中的一些专业术语、缩略语如下表1所示。

|  |  |
| --- | --- |
| 术语/缩略语 | 描述 |
| LuaJIT | LuaJIT即采用C语言写的Lua代码的解释器。 |
| Lua | Lua是一个小巧的脚本语言。由标准C编写而成，几乎在所有操作系统和平台上都可以编译，运行。 |

表 1 专业术语及缩略语

## 参考资料

[1] Torch官方文档[http://torch.ch/](http://spark.apache.org)

## 标准、条约和约定

参照如下标准：

GB/T 13702-1992 计算机软件分类与代码

GB/T 19003-2008 软件工程

GB/T 5538-1995 软件工程标准分类法

GB/T 9385-2008 计算机软件需求规格说明

GB/T 5532-2008 计算机软件测试规范

GB/T 18221-2000 信息技术程序设计语言

GB/T 8567-2006 计算机软件文档编制规范

# 项目概述

## 项目目标

从Torch平台出发，重新编写起需求规格说明书、测试需求规格说明书，并根据需求规格说明书与测试需求规格说明书对Torch进行模块化测试，并在此基础上拓展应用。

## 工作内容

1.学习使用Torch平台和Lua语言。

2.学习调研当前深度神经网络的发展现状，认真论证深度神经网络压缩技术的必要性，细致分析项目需求，完成需求文档。

3.在Torch平台下使用Lua语言搭建深度神经网络，实现深度神经网络压缩算法，同时根据评审和工程中出现的新需求、新问题，进行项目开发和文档编写的迭代。

4.使用主流数据集对算法的性能指标进行独立的测试，完成测试报告。

5.根据课程的整体安排和每个实验的具体要求，依照项目时间计划完成其他各类文档和报告。

## 应交付成果

项目应交付的产品如表2所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 名称 | 交付时间 | 要求 | 备注 |
| No.1 | 项目计划书 | 2016/3/15 | 完整、明确 | 暂无 |
| No.2 | 项目进度表 | 2016/3/15 | 完整、详细 | 暂无 |
| No.3 | 项目工作日志 | 待定 | 暂无 | 暂无 |
| No.4 | 软件需求规格说明书 | 待定 | 暂无 | 暂无 |
| No.5 | 需求检查单 | 待定 | 暂无 | 暂无 |
| No.6 | 测试需求规格说明书 | 待定 | 暂无 | 暂无 |
| No.7 | 软件测试分析报告 | 待定 | 暂无 | 暂无 |
| No.8 | 软件测试评审报告 | 待定 | 暂无 | 暂无 |
| No.9 | 项目工作量统计报告 | 待定 | 暂无 | 暂无 |
| No.10 | 项目方法总结报告 | 待定 | 暂无 | 暂无 |
| No.11 | 项目个人总结 | 待定 | 暂无 | 暂无 |

表 2 应交付成果列表

## 项目验收方式和依据

1. 验收方式

项目验收方式采用中期课堂评审和终期交付验收。

1. 验收依据

验收依据“应交付成果”清单进行验收。

# 项目团队组织

## 组织结构

此次项目有两个角色：组员和组长。组长与组员沟通协调，安排具体事宜、分配相应的任务。而组员则需要根据组长的安排要求，及时完成任务。

## 人员组成

人员组成如下表3所示。

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **学号** | **姓名** | **角色** |
| 1 | SY1606113 | 陈伟民 | 组员，组长 |
| 2 | SY1506116 | 付 强 | 组员 |
| 3 | SY1606323 | 李恬霖 | 组员 |
| 4 | SY1606411 | 曹 进 | 组员 |

表 3 人员组成

## 协作与沟通

小组成员通过微信群、电话、邮件等方式进行及时沟通，在项目有重大决定时，可举行临时面谈会议。而且小组通过文档、Git工具协作完成相应的具体任务。

沟通和协作都要记录具体内容。

# 实施计划

## 计划分析

## 安装Torch

在初步了解的基础上，尽可能的全面了解Torch，熟悉Torch的一些常用实例，分析的源代码。尽可能的从开发思想上去了解开发者的意图。

## 需求分析

确定对系统的综合要求。分析Torch平台的需求，修正系统的开发计划。主要难点包括需求分析和建模。需求分析时还需要了解一些评审指标和制作评审文档等。

## 测试分析

对其进行测试，检查软件有没有错误，决定Torch是否具有稳定性，写出相应的测试规范和测试用例。和分析需求时一样，我们还要了解测试评审指标及制定评审文档。

## 功能实现

首先完成神经网络压缩算法的编码，定义神经网络模型，再确定一个应用场景，定义损失函数，在训练集上训练网络，在测试集上测试网络；再对相同的应用场景测试压缩神经网络的性能。

## 其他软件管理工具学习

在研究过程中，我们会用到项目管理和项目配置工具，还有建模工具等，在整个的开发的过程中，我们都将伴随开发工具的学习，不再单独为研究工具而花费时间，可以通过其他阶段的时间剩余来合理安排。

## 总体进度计划

|  |  |  |  |
| --- | --- | --- | --- |
| 任务名称 | 工期 | 开始时间 | 完成时间 |
| **项目确定以及介绍** | **2 个工作日** | **2017年3月12日** | **2017年3月13日** |
| 项目调研 | 5 工时 | 2017年3月12日 | 2017年3月13日 |
| 选题会议 | 8 工时 | 2017年3月13日 | 2017年3月13日 |
| **项目计划书** | **3 个工作日** | **2017年3月14日** | **2017年3月16日** |
| 撰写引言 | 12 工时 | 2017年3月14日 | 2017年3月15日 |
| 项目概述 | 12 工时 | 2017年3月14日 | 2017年3月15日 |
| 目的及成果 | 12 工时 | 2017年3月14日 | 2017年3月15日 |
| mpp学习与使用 | 12 工时 | 2017年3月15日 | 2017年3月16日 |
| **软件需求分析** | **5 个工作日** | **2017年3月19日** | **2017年3月24日** |
| 软件模块分析 | 3 个工作日 | 2017年3月19日 | 2017年3月22日 |
| 测试需求分析 | 3 个工作日 | 2017年3月19日 | 2017年3月22日 |
| 软工规范分析 | 3 个工作日 | 2017年3月19日 | 2017年3月22日 |
| 扩展功能分析 | 3 个工作日 | 2017年3月19日 | 2017年3月22日 |
| 需求分析会议 | 6 工时 | 2017年3月23日 | 2017年3月23日 |
| **软件需求评审** | **2 个工作日** | **2017年3月25日** | **2017年3月26日** |
| 修改软件模块需求 | 2 个工作日 | 2017年3月25日 | 2017年3月26日 |
| 修改测试需求 | 2 个工作日 | 2017年3月25日 | 2017年3月26日 |
| 修改软工规范需求 | 2 个工作日 | 2017年3月25日 | 2017年3月26日 |
| 修改扩展功能需求 | 2 个工作日 | 2017年3月25日 | 2017年3月26日 |
| 需求评审会议 | 6 工时 | 2017年3月26日 | 2017年3月26日 |
| **软件需求复评审** | **2 个工作日** | **2017年3月27日** | **2017年3月28日** |
| 确定软件模块需求 | 2 个工作日 | 2017年3月27日 | 2017年3月28日 |
| 确定测试需求 | 2 个工作日 | 2017年3月27日 | 2017年3月28日 |
| 确定软工规范需求 | 2 个工作日 | 2017年3月27日 | 2017年3月28日 |
| 确定扩展功能需求 | 2 个工作日 | 2017年3月27日 | 2017年3月28日 |
| **软件产品改进与展示** | **10 个工作日** | **2017年3月29日** | **2017年4月10日** |
| 软件改进 | 9 个工作日 | 2017年3月29日 | 2017年4月8日 |
| 软件展示 | 6 工时 | 2017年4月10日 | 2017年4月10日 |
| **软件测试分析** | **6 个工作日** | **2017年4月11日** | **2017年4月16日** |
| 测试需求分析 | 6 个工作日 | 2017年4月11日 | 2017年4月16日 |
| **软件测试评审** | **2 个工作日** | **2017年4月17日** | **2017年4月18日** |
| 修改测试需求 | 2 个工作日 | 2017年4月17日 | 2017年4月18日 |
| **软件测试复评审** | **2 个工作日** | **2017年4月19日** | **2017年4月20日** |
| 确定测试需求 | 2 个工作日 | 2017年4月19日 | 2017年4月20日 |
| 软件演示与测评 | 4 个工作日 | 2017年4月21日 | 2017年4月26日 |
| 综合实验分析 | 4 个工作日 | 2017年4月27日 | 2017年4月30日 |
| 综合实验总结 | 4 个工作日 | 2017年5月4日 | 2017年5月8日 |
| **软件进度计划与控制** | **76 个工作日** | **2017年3月12日** | **2017年6月24日** |
| 每周组会 | 76 个工作日 | 2017年3月12日 | 2017年6月24日 |
| 每日更新github | 76 个工作日 | 2017年3月12日 | 2017年6月24日 |
| 工作量估计与统计分析 | 76 个工作日 | 2017年3月12日 | 2017年6月24日 |
| 配置管理 | 76 个工作日 | 2017年3月12日 | 2017年6月24日 |

## 项目控制计划

暂无。

# 关键问题

暂无。

# 专题计划要点

暂无。