

Final Report

|  |
| --- |
| School of Computing  Faculty of Engineering AND PHYSICAL SCIENCES |

<Distributed Training with k-async SGD>

<Wei Yangchi>

Submitted in accordance with the requirements for the degree of  
< BSc Computer Science >

**<2021/22>**

The candidate confirms that the following have been submitted*:*

*<As an example>*

|  |  |  |
| --- | --- | --- |
| **Items** | **Format** | **Recipient(s) and Date** |
| *Final Report* | *Hard copy and PDF file* | *Hard copy handed to SSO (DD/MM/YY); PDF uploaded to Minerva (DD/MM/YY)* |
| *Participant consent forms* | *PDF file / file archive* | *Uploaded to Minerva (DD/MM/YY)* |
| *Link to online code repository* | *URL* | *Sent to supervisor and assessor (DD/MM/YY)* |
| *User manuals* | *PDF and/or hard copy* | *Sent to client and supervisor (DD/MM/YY)* |

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

(Signature of student)

© <Year of Submission> The University of Leeds and <full name of candidate>

# Summary

The mission of this project is to empirically investigate the effect of parameter k values on the convergence of distributed k-asynchronous SGD training by conducting distributed model training tests on Pytorch. To deal with possible outliers and communication bottlenecks, researchers recently proposed an optimization of k-async SGDs, in which PS waits only for the previous k of n workers to complete each round of training first.

# Acknowledgements

*<This page should contain any acknowledgements to those who have assisted with your work. Where you have worked as part of a team, you should, where appropriate, reference to any contribution made by others to the project.>*

*Note that it is not acceptable to solicit assistance on ‘proof reading’ which is defined as “the systematic checking and identification of errors in spelling, punctuation, grammar and sentence construction, formatting and layout in the text”; see*

https:://www.leeds.ac.uk/secretariat/documents/proof\_reading\_policy.pdf

# Table of Contents (example of how to format)

Summary iii

Acknowledgements iv

Table of Contents v

Chapter 1 Level 1 Heading with ‘heading 1’ Style 1

1.1 Level 2 Heading with ‘heading 2’ Style 1

1.1.1 Level 3 Heading with ‘heading 3’ Style 1

1.1.1.1 Level 4 Heading with ‘heading 4’ Style 1

Chapter 2 Tables and Figures 2

2.1 Tables using the ‘table caption’ and ‘table description’ Styles 2

2.2 Figures using the ‘figure caption’ and ‘figure description’ Styles 2

List of References 3

Appendix A Self-appraisal 4

A.1 Level 2 Heading with ‘heading 2’ Style 4

A.1.1 Level 3 Heading with ‘heading 3’ Style 4

A.1.1.1 Level 4 Heading with ‘heading 4’ Style 4

Appendix B External materials 4

B.1 Level 2 Heading with ‘heading 2’ Style 4

# Chapter 1 Introduction and Background Research

<Recommend using ‘Heading 1’ for chapter titles, ‘Heading 2’ style for section headings, ‘Heading 3’ for subsection headings, and ‘Heading 4’ for sub-subsection headings, but whatever you choose you must be consistent. Don’t forget that text (other than headings) should be 11 point.>

## 1.1 Introduction

<A brief introduction suitable for a non-specialist, *i.e.* without using technical terms or jargon, as far as possible. This may be similar/the same as that in the 'Outline and Plan' document. The remainder of this chapter will normally cover everything to be assessed under the `Background Research` criterion in the mark scheme.>

Today, data-parallel distributed machine learning is widely used to enable efficient iterative model training on large data sets (e.g., using stochastic gradient descent, SGD). To guarantee convergence, Worders will periodically synchronize its local model updates using a parameter server or collective communication operations such as All-Reduce during training.

Suppose n worker threads are training a model and synchronizing their model updates with a centralized parameter server (PS). In the widely adopted synchronous SGD training, in order to move on to the next round of training, PS first waits for all workers to complete: then, by aggregating the results of its local training, PS obtains the latest global model and sends it to all workers. Similar to the case of All-Reduce, in order to deal with possible outliers and communication bottlenecks, researchers recently proposed an optimization of k-async SGD, in which PS waits only for the previous of n workers K, they are completed first, each round of training.

The mission of this project is to empirically investigate the effect of parameter k values on the convergence of distributed k-asynchronous SGD training by conducting distributed model training tests on Pytorch.

## 1.2 Literature Review

<This section heading is purely a suggestion -- you should subdivide this chapter in whatever manner you think makes most sense for your project. It may also make sense to spread the `Background Research' over more than one chapter, in which case they should be named sensibly.>

1.2.1 Relevant Literature

1.2.1.1 Neural Networks

In 1943, McCulloch and Pitts, first described how a neuron could be modeled mathematically.

[25] This neurons can be linked together to form a network of interconnected nodes. Neural

networks differ in their complexity, but deep networks that consisted of multiple layers can be

trained to learn to produce complex and intricate behaviour such as identifying handwritten

digits. [7] Similar to how neurons work in neurology, a neuron will be considered active if some pre-established weights multiplied by the neurons in the previous layer are activity crosses a certain threshold. As neural networks are a form of supervised machine learning, the weights are learned through a process of training where the weights are constantly readjusted to more accurately reflect the desired output for a given input.

1.2.1.2 Gradient Descent

Gradient descent algorithms are frequently used to train neural networks. There are many

different types of gradient descent algorithms, but ultimately all gradient descent algorithms

aim to reduce the errors produced by the output of the neural network by reaching a global

minimum in multi-dimensional space. A learning rate is used to dictate how much the weights should be update upon each iteration of training. If the learning rate is too high the algorithm may oscillate around the minimum, however if the learning rate is too low the algorithm may never find the global minimum since it would likely get trapped in a local minimum.

1.2.2.3 PyTorch

PyTorch is a popular open-source machine learning framework that can be used for computer vision tasks. Pix2Pix has a version implemented with Pytorch so we have opted to use it for the sake of consistency [29]. Alternatively, TensorFlow[1] could be used, however within the machine learning community Pytorch is generally used for research projects while TensorFlow is used for industry scale projects.

1.2.2.4 Anaconda

Anaconda is a package and environment manager for Python. It specialises in providing tools to facilitate data science and machine learning applications. Anaconda can be used to create virtual environments, which ensure that libraries and versions remain consistent between different work spaces.

1.2.2.5 CUDA

CUDA is a toolkit and parallel computing platform. CUDA facilitates parallel computation on

Nvidia GPUs and therefore allows computationally heavy applications intensive applications to run faster. It is a prerequisite for running the Pix2Pix and BicycleGAN training and testing

code.

1.2.2.6 NumPy

NumPy is a Python-based library used for performing numerical computation with

multi-dimensional arrays. It is a useful library to use with machine learning projects and will

simplify the project [9].

Today, data-parallel distributed machine learning is widely used to enable efficient iterative model training on large data sets (e.g., using stochastic gradient descent, SGD). To guarantee convergence, Worders will periodically synchronize its local model updates using a parameter server or collective communication operations such as All-Reduce during training.

Suppose n worker threads are training a model and synchronizing their model updates with a centralized parameter server (PS). In the widely adopted synchronous SGD training, in or der to move on to the next round of training, PS first waits for all workers to complete: then, by aggregating the results of its local training, PS obtains the latest global model and sends it to all workers. Similar to the case of All-Reduce, in order to deal with possible outliers and communication bottlenecks, researchers recently proposed an optimization of k-async SGD, in which PS only waits for the previous k of n workers, who complete first, per round of training.

The mission of this project is to empirically investigate the effect of parameter k values on the convergence of distributed k-asynchronous SGD training by conducting distributed model training tests on Pytorch.

1.2 background research

Deep learning

深度学习(DL, Deep Learning)是[机器学习](https://baike.baidu.com/item/%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0/217599)(ML, Machine Learning)领域中一个新的研究方向，它被引入机器学习使其更接近于最初的目标——[人工智能](https://baike.baidu.com/item/%E4%BA%BA%E5%B7%A5%E6%99%BA%E8%83%BD/9180" \t "_blank)(AI, Artificial Intelligence)。[1]

深度学习是学习[样本数据](https://baike.baidu.com/item/%E6%A0%B7%E6%9C%AC%E6%95%B0%E6%8D%AE/12726279)的内在规律和表示层次，这些学习过程中获得的信息对诸如文字，[图像](https://baike.baidu.com/item/%E5%9B%BE%E5%83%8F/773234)和声音等数据的解释有很大的帮助。它的最终目标是让机器能够像人一样具有分析学习能力，能够识别文字、图像和声音等数据。 深度学习是一个复杂的机器学习算法，在语音和图像识别方面取得的效果，远远超过先前相关技术。[1]

深度学习在[搜索技术](https://baike.baidu.com/item/%E6%90%9C%E7%B4%A2%E6%8A%80%E6%9C%AF/1447197)，[数据挖掘](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E6%8C%96%E6%8E%98/216477)，机器学习，[机器翻译](https://baike.baidu.com/item/%E6%9C%BA%E5%99%A8%E7%BF%BB%E8%AF%91/411793)，[自然语言处理](https://baike.baidu.com/item/%E8%87%AA%E7%84%B6%E8%AF%AD%E8%A8%80%E5%A4%84%E7%90%86/365730)，[多媒体学习](https://baike.baidu.com/item/%E5%A4%9A%E5%AA%92%E4%BD%93%E5%AD%A6%E4%B9%A0/10528812)，语音，推荐和个性化技术，以及其他相关领域都取得了很多成果。深度学习使机器模仿视听和思考等人类的活动，解决了很多复杂的模式识别难题，使得人工智能相关技术取得了很大进步。[1]

深度学习是一类[模式分析](https://baike.baidu.com/item/%E6%A8%A1%E5%BC%8F%E5%88%86%E6%9E%90/12598452)方法的统称，就具体研究内容而言，主要涉及三类方法：[2] 

(1)基于卷积运算的神经网络系统，即卷积神经网络(CNN)。[2]

(2)基于多层神经元的自编码[神经网络](https://baike.baidu.com/item/%E7%A5%9E%E7%BB%8F%E7%BD%91%E7%BB%9C/16600562)，包括自编码( Auto encoder)以及近年来受到广泛关注的[稀疏编码](https://baike.baidu.com/item/%E7%A8%80%E7%96%8F%E7%BC%96%E7%A0%81/10289670)两类( Sparse Coding)。[2]

(3)以多层自编码神经网络的方式进行预训练，进而结合鉴别信息进一步优化神经网络权值的深度置信网络(DBN)。[2]

通过多层处理，逐渐将初始的“低层”特征表示转化为“高层”特征表示后，用“简单模型”即可完成复杂的分类等学习任务。由此可将深度学习理解为进行“特征学习”（feature learning）或“表示学习”（representation learning）。[3] 

以往在机器学习用于现实任务时，描述样本的特征通常需由人类专家来设计，这成为“特征工程”（feature engineering）。众所周知，特征的好坏对泛化性能有至关重要的影响，人类专家设计出好特征也并非易事；特征学习（[表征学习](https://baike.baidu.com/item/%E8%A1%A8%E5%BE%81%E5%AD%A6%E4%B9%A0/2140515)）则通过机器学习技术自身来产生好特征，这使机器学习向“全自动数据分析”又前进了一步。[3]

近年来，研究人员也逐渐将这几类方法结合起来，如对原本是以有监督学习为基础的卷积神经网络结合自编码神经网络进行无监督的预训练，进而利用鉴别信息微调网络参数形成的[卷积](https://baike.baidu.com/item/%E5%8D%B7%E7%A7%AF/9411006)深度置信网络。与传统的学习方法相比，深度学习方法预设了更多的模型参数，因此模型训练难度更大，根据统计学习的一般规律知道，模型参数越多，需要参与训练的数据量也越大。[2]

20世纪八九十年代由于计算机计算能力有限和相关技术的限制，可用于分析的数据量太小，深度学习在模式分析中并没有表现出优异的识别性能。自从2006年， Hinton等提出快速计算受限玻耳兹曼机(RBM)网络权值及偏差的CD-K算法以后，RBM就成了增加神经网络深度的有力工具，导致后面使用广泛的DBN(由 Hinton等开发并已被微软等公司用于语音识别中)等深度网络的出现。与此同时，稀疏编码等由于能自动从数据中提取特征也被应用于深度学习中。基于局部数据区域的卷积神经网络方法今年来也被大量研究。[2]

Distributed Machine Learning

近年来，人工智能取得了飞速的发展，实现了一个又一个技术突破。这些成功的幕后英雄是大量的训练数据、超大规模的机器学习模型以及分布式的训练系统。一系列有关分布式机器学习的研究工作，从并行模式、跨机通信到聚合机制，从算法设计、理论推导到系统构建，都在如火如荼地展开。人们不仅发表了大量的学术论文，也开发出一批实用性很强的分布式机器学习系统。本书的目的是向读者全面展示分布式机器学习的现状，深入分析其中的核心技术问题，并且讨论该领域未来发展的方向。本书既可以作为研究生从事分布式机器学习方向研究的参考文献，也可以作为人工智能从业者进行算法选择和系统设计的工具书。

Distributed Data Parallel

DistributedDataParallel（DDP）是依靠多进程来实现数据并行的分布式训练方法（简单说，能够扩大batch\_size，每个进程负责一部分数据)。在使用DDP分布式训练前，有几个概念或者变量，需要弄清楚，这样后面出了bug大概知道从哪里入手，包括：

group: 进程组，一般就需要一个默认的

world size: 所有的进程数量

rank: 全局的进程id

local rank：某个节点上的进程id，进行

local\_word\_size: 某个节点上的进程数 (相对比较少见)

这里需要注意的是，目前为止所有的概念的基本单元都是进程，与GPU没有关系，一个进程可以对应若干个GPU。 所以world\_size 并不是等于所有的GPU数量，而人为设定的，这一点网上的很多描述并不准确。只不过平时用的最多的情况是一个进程使用一块GPU，这种情况下 world\_size 可以等于所有节点的GPU数量。

假设所有进程数即 world\_size为W，每个节点上的进程数即local\_world\_size为L，则每个进程上的两个ID：

rank的取值范围：[0, W-1]，rank=0的进程为主进程，会负责一些同步分发的工作

local\_rank的取值：[0, L-1]

官方的示意图的非常形象，如下，

图形用户界面, 文本, 应用程序

描述已自动生成

假定有2个机器或者节点，每个机器上有4块GPU。图中一共有4个进程，即world\_size=4，那这样每个进程占用两块GPU，其中rank就是[0,1,2,3]，每个节点的local\_rank就是[0,1]了，其中local\_world\_size 也就是2。 这里需要注意的是，local\_rank是隐式参数，即torch自动分配的。比如local\_rank 可以通过自动注入命令行参数或者环境变量来获得) 。

从torch1.10开始，官方建议使用环境变量的方式来获取local\_rank, 在后期版本中，会移除命令行的方式。

一些简单的测试：

In this work, provide a novel convergence analysis of K-async SGD for fixed , relaxing the following assumptions in existing literature.

It is also assumed that the staleness process is independent of w. While this assumption simplifies the analysis greatly, it is not true in practice. For instance, for a two

### 

# Chapter 2 Methods

<Everything that comes under the `Methods' criterion in the mark scheme should be described in one, or possibly more than one, chapter(s). Note that it is not normally relevant to include extensive code, but short snippets for key aspects may be suitable.>

## 2.1 Table example

|  |  |  |
| --- | --- | --- |
| **Heading One** | **Heading Two** | **Heading Three** |
| 1.1 | 1.2 | 1.3 |
| 1.21 | 1.22 | 12.3 |
| 12.31 | 12.32 | 12.33 |

Text before table. Text before table. Text before table. Text before table. Text before table. Text before table. Text before table. Text before table. Text before table. Text before table.

**Table 2.1** This is the table description in the ‘table description’ style.

Nowadays, data-parallel distributed machine learning is widely employed to achieve efficient iterative model training(e.g., using stochastic gradient descent, SGD) over huge amounts of datasets. To guarantee convergence, during the training, worders will periodically synchronize their local model updates using parameter servers or collective communication operations like All-Reduce.

Suppose that n workers are training a model and synchronize their model updates with a centralized parameter server(PS). In the widely employed synchronous SGD training, to move to the next round of training, the PS first waits for all the workers to finish: then, by aggregating their local training results, the PS obtains the newest global model and sends it to all workers. Similar to the case of All-Reduce, to deal with possible stragglers and communication bottlenecks, researchers recently propose the optimization of k-async SGD, in which, the PS only waits for the first k out of n workers that finish first, for each round of training.

The task of this project is to empirically study the impact of the value of parameter k on the convergence of the distributed k-async SGD training in detail, by conducting distributed model training tests upon Pytorch.

## 2.2 Figure example

Figures can be added using the Illustrations section of the Insert tab.



**Figure 2.1** This is the figure description in the ‘figure description’ style.

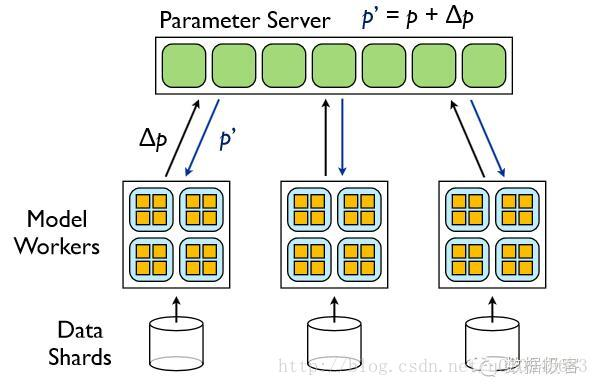
Nowadays, data-parallel distributed machine learning is widely employed to achieve efficient iterative model training(e.g., using stochastic gradient descent, SGD) over huge amounts of datasets. To guarantee convergence, during the training, worders will periodically synchronize their local model updates using parameter servers or collective communication operations like All-Reduce.

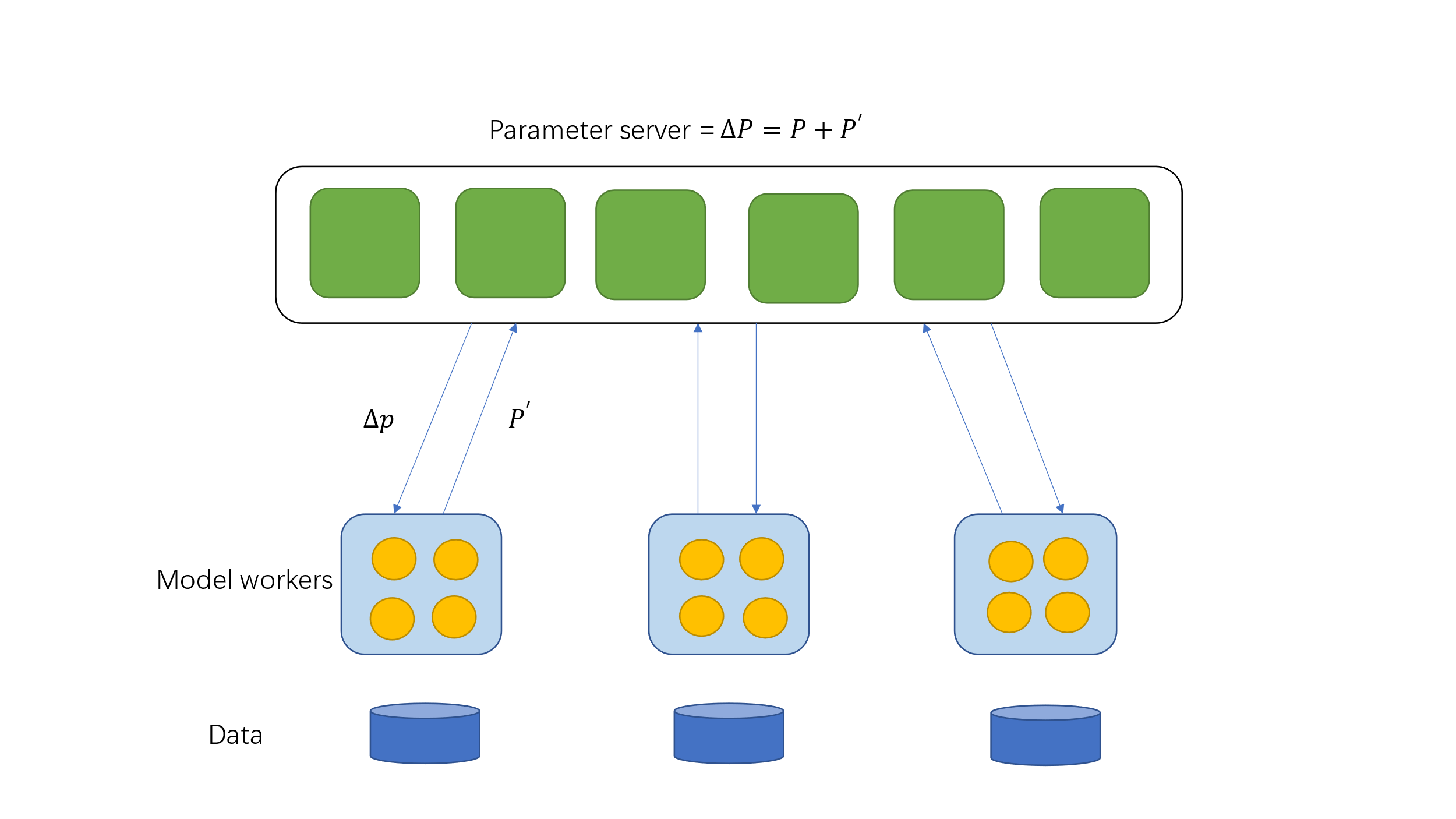
Suppose that n workers are training a model and synchronize their model updates with a centralized parameter server(PS). In the widely employed synchronous SGD training, to move to the next round of training, the PS first waits for all the workers to finish: then, by aggregating their local training results, the PS obtains the newest global model and sends it to all workers. Similar to the case of All-Reduce, to deal with possible stragglers and communication bottlenecks, researchers recently propose the optimization of k-async SGD, in which, the PS only waits for the first k out of n workers that finish first, for each round of training.

The task of this project is to empirically study the impact of the value of parameter k on the convergence of the distributed k-async SGD training in detail, by conducting distributed model training tests upon Pytorch.

2.1 Design and implementation of the solution, supported by justification

of choices made



tg

\

Figure 1: The working principle of PS

Description

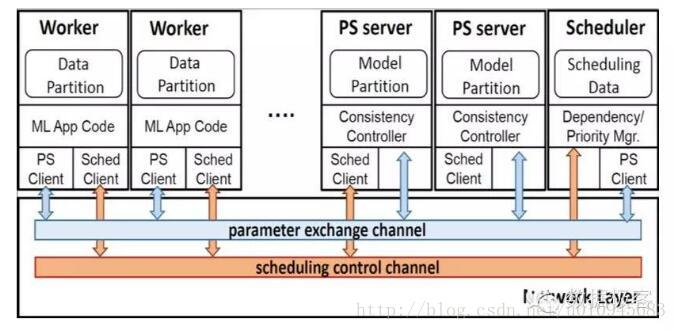


Figure 1: The working principle of System

Description

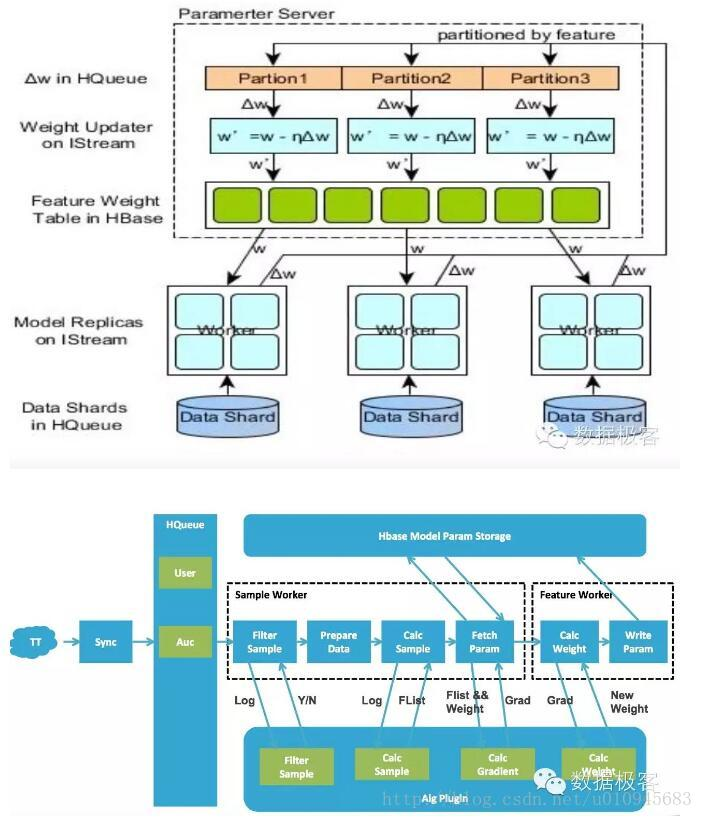


Figure 1: The working principle of System

Description

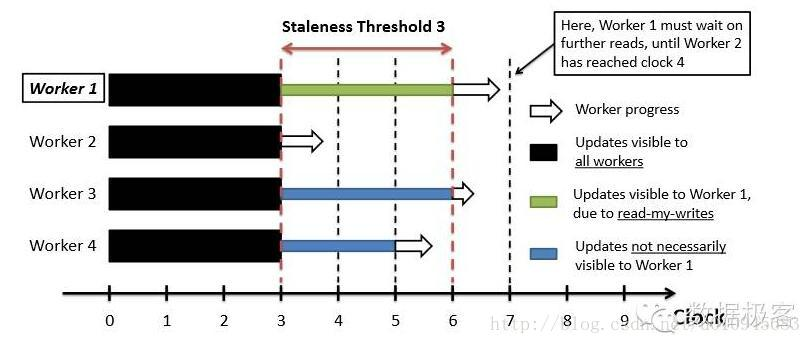


Figure 1: The working principle of Workers

Description

Parameter server

ω^'=w-η∇F(w)

Worker 1 Worker 2 Worker 3

K-batch-sync SGD: In K-batch-sync, all the P workers starts computing gradients with the same wj，whenever any worker finishes, it pushes its update to the PS and evaluates

Table I: Important Notations

|  |  |
| --- | --- |
| CONSTANTS | RANDOM VARIABLES |
| J: Total Iterations | J: Total Iterations |
| P: Total number of workers | J: Total Iterations |
| K: Number of workers or mini-batches to wait for | P: Total number of workers |
| M: mini-batches to wait for | K: Number of workers or mini-batches to wait for |
|  |  |

2.2 For software, evidence that code was properly managed using a version

control system, and followed standard good practice in structure

Developed software source code is managed in a Public GitLab repository. At: https://gitlab.com/sc18l2a/comp3931-individual-project Report detailing the processes of planning, research, development, testing and evaluation.

There is always the possibility of some unforeseen circumstance (such as a corrupted hard drive or a fire) interfering with this project. For this reason, the version control software ’Git’ will be used to manage the source code relating to this project. It will serve as a backup and will also allow for easy accessibility to prior versions. A link to the GitLab repository can be found here. 图形用户界面, 文本, 应用程序

描述已自动生成

2.3 Evidence of data collection and testing

文本

描述已自动生成

文本

描述已自动生成

图形用户界面, 文本, 应用程序

描述已自动生成

文本

描述已自动生成

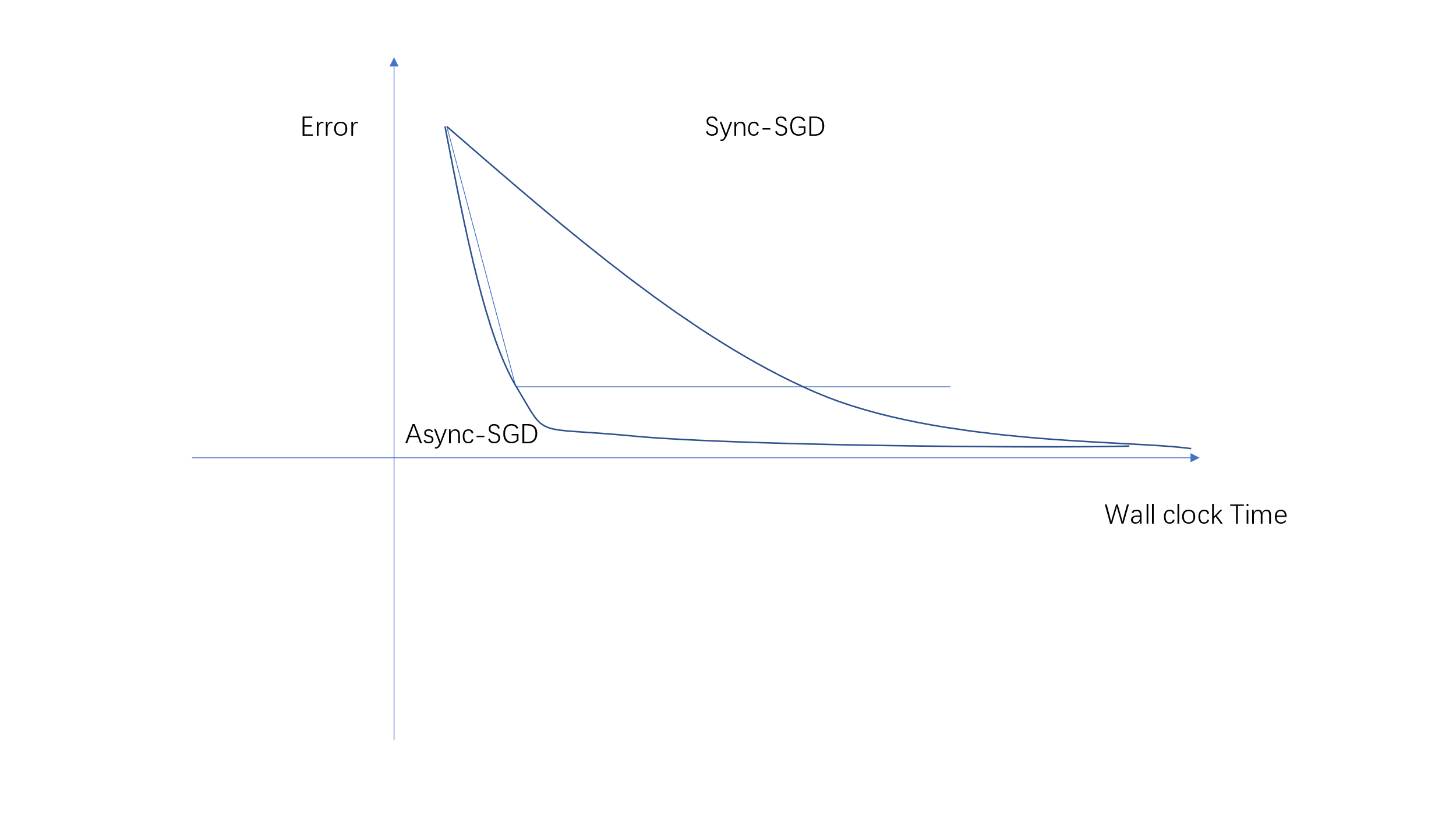
电脑屏幕的照片

低可信度描述已自动生成

2.4 Derivation of theoretical proofs

Parameter server

Worker 1 Worker 2 Worker 3



2.5 UML class diagrams describing structure of software solution

2.6 Software validation, e.g. unit tests, reproduction of known results etc

2.7 Details of project management methodology, e.g. sprints

Developed software source code is managed in a Public GitLab repository. At: https://gitlab.com/sc18l2a/comp3931-individual-project Report detailing the processes of planning, research, development, testing and evaluation.

There is always the possibility of some unforeseen circumstance (such as a corrupted hard drive or a fire) interfering with this project. For this reason, the version control software ’Git’ will be used to manage the source code relating to this project. It will serve as a backup and will also allow for easy accessibility to prior versions. A link to the GitLab repository can be found here.

# Chapter 3 Results

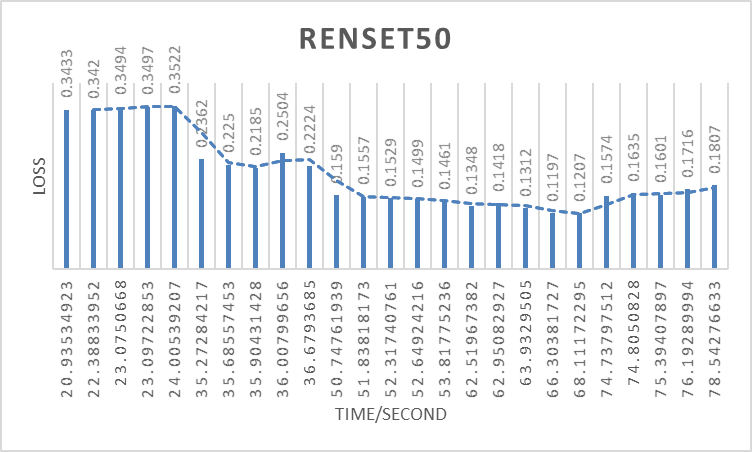
<Results, evaluation (including user evaluation) *etc*. should be described in one or more chapters. See the `Results and Discussion' criterion in the mark scheme for the sorts of material that may be included here.>

• Quantitative and systematic approach providing objective evidence of

the quality of the solution.

• Appropriate technical and/or user evaluation.

• Results clearly related to motivation/goals as appropriate.



• Outcome of the study and ideas for future work.

Online test and graphical by special API which transform the data in EXCEL to a Echarts.

To do more and significant test for more distributed machine learning models.

Make the application a suitable script that could be easily used on others’ EI(e)or research

• Performance/complexity/accuracy analysis and interpretation.



• Results and discussion of user evaluation questionnaires.

# Chapter 4 Discussion

<Everything that comes under the `Results and Discussion' criterion in the mark scheme that has not been addressed in an earlier chapter should be included in this final chapter. The following section headings are suggestions only.>

## 4.1 Conclusions

<Text in 11-point size and 1.5 line spacing.>

**Ethical concerns**

**Social Implications**

Automation can lead to rapid change in job markets and otherwise change people’s lives

significantly. The scope of this project is too small to drastically make any meaningful impact

on a societal level.

**Privacy Concerns**

This project will be using images of textures using under the creative commons or with the

permission of the creators. No infringement on people’s human right to privacy will take place as no image of people will be used.

**Self-Appraisal**

I was pleased with the outcome of this project. I felt like I had achieved the goals that I had

set out to accomplish, despite the task being a challenging one. The hardest part of the project

for me was the background research, since I was not used to reading academic papers

(particularly the ones that were mathematically dense). Fortunately none of the risked

identified in the Risk mitigation section applied to me over the course of this project and

overall I felt like the current lockdown due to the Coronavirus pandemic was beneficial to me

because it allowed me to focus on my studies.

## 4.2 Ideas for future work

<Text in 11-point size and 1.5 line spacing.>

**Further Work and Improvements**

If I had additional time to spend on this project I would have liked to implement a feature that

allowed pixel-based colour blending between non-adjacent edges on the UV map. Since the

textures generated are not seem-less, the current implementation creates an unpleasant effect between separate islands in the UV map. This could be dealt with by interpolating between two pixels’ colours for non-adjacent edges. This would likely create a blurred effect on the texture however this is still better than harsh edges.

I would have also liked to add the ability to add new models from the user interface. This

would be simple to implement however I had limited time at the end of the project and felt

other aspects took priority.

Additional training could also be performed and if more time was made available more models with different parameters would be created and tested.

Lastly, I believe that additional effort put into the pre-processing of the textures datasets

would have increased the realism of the textures. Many textures found in the original datasets contain elements of other classes (such as a white tabletop in the background of an image of a wooden toy). Within the DTD a mask was included that could have been incorporated to only include the textured item in the image.

# List of References

*<It is expected that the list would reflect the breadth and depth of scholarly research undertaken by the student during the course of the project.>*

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat,

G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In 12th

{USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16),

pages 265–283, 2016. Accessed: 08/05/2020.

[2] B. F. Antonio Torralba, Rob Fergus. TinyImages Apology, 2020. Accessed: 30/12/2020.

[4] A. Brock, J. Donahue, and K. Simonyan. Large Scale GAN Training for High Fidelity

Natural Image Synthesis, 2019. Accessed: 30/12/2020.

[5] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing Textures in

the Wild. In Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition

(CVPR), 2014. Accessed: 30/12/2020.

[6] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing Textures in

the Wild Website, 2014. Accessed: 30/12/2020.

[7] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber. Deep Big Simple

Neural Nets Excel on Handwritten Digit Recognition. CoRR, abs/1003.0358, 2010.

Accessed: 08/05/2020.

[8] B. O. Community. Blender - a 3D modelling and rendering package. Blender Foundation,

Stichting Blender Foundation, Amsterdam, 2018. Accessed: 08/05/2020.

[9] T. S. community. NumPy Documentation, 2020. Accessed: 30/12/2020.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale

hierarchical image database. In 2009 IEEE conference on computer vision and pattern

recognition, pages 248–255. Ieee, 2009. Accessed: 07/05/2020.

[11] L. A. Gatys, A. S. Ecker, and M. Bethge. A Neural Algorithm of Artistic Style, 2015.

Accessed: 15/04/2020.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair,

A. Courville, and Y. Bengio. Generative adversarial networks. Communications of the

ACM, 63(11):139–144, 2020. Accessed: 30/12/2020.

[13] C. Hesse. Image-to-Image Demo, 2017. Accessed: 30/12/2020.

[14] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, G. Klambauer, and S. Hochreiter.

GANs Trained by a Two Time-Scale Update Rule Converge to a Nash Equilibrium. CoRR,

abs/1706.08500, 2017. Accessed: 07/05/2020.

[16] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-Image Translation with

Conditional Adversarial Networks, 2018. Accessed: 30/12/2020.

[17] A. Jacq. Neural transfer using pytorch. https://github.com/pytorch/tutorials/blob/

master/advanced\_source/neural\_style\_tutorial.py, 2019. Accessed: 14/04/2020.

[18] T. Karras, S. Laine, and T. Aila. A Style-Based Generator Architecture for Generative

Adversarial Networks, 2019. Accessed: 30/12/2020.

[19] P. Lee. Learning from Tay’s introduction, 2016. Accessed: 30/12/2020.

[20] C. Li and M. Wand. Precomputed Real-Time Texture Synthesis with Markovian

Generative Adversarial Networks, 2016. Accessed: 08/05/2020.

[21] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised Image-to-Image Translation Networks,

2018. Accessed: 30/12/2020.

[22] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet. Are GANs Created Equal?

A Large-Scale Study, 2018. Accessed: 07/05/2020.

[23] J. Maillot, H. Yahia, and A. Verroust. Interactive texture mapping, 08 1993. Accessed:

30/12/2020.

[24] P. Manisha and S. Gujar. Generative Adversarial Networks (GANs): What it can generate

and What it cannot?, 2019. Accessed: 13/04/2020.

[25] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous

activity. The bulletin of mathematical biophysics, 5(4):115–133, 1943. Accessed:

08/05/2020.

[26] M. Mirza and S. Osindero. Conditional generative adversarial nets, 2014. Accessed:

30/12/2020.

[27] T. U. of Leeds. Leeds Arc, 2020. Accessed: 30/12/2020.

[28] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. Semantic Image Synthesis with

Spatially-Adaptive Normalization, 2019. Accessed: 30/12/2020.

[29] PyTorch. Pytorch Documentation, 2020. Accessed: 30/12/2020.

[31] M. Seitzer. pytorch-fid: FID Score for PyTorch.

https://github.com/mseitzer/pytorch-fid, August 2020. Version 0.1.1.

[32] L. Sharan, R. Rosenholtz, and E. Adelson. Material perception: What can you see in a

brief glance? Journal of Vision - J VISION, 14(9), 2014. Accessed: 30/12/2020.

[33] L. Sharan, R. Rosenholtz, and E. H. Adelson. Accuracy and speed of material

categorization in real-world images. Journal of Vision, 14(10), 2014. Accessed:

08/05/2020.

[34] J. Simon. Artbreeder, 2019. Accessed: 30/12/2020.

[35] A. Tsui and R. Jeżewski. Blender off addon.

https://github.com/alextsui05/blender-off-addon, 2014. Accessed: 14/04/2020.

Deep Representation for Volumetric Shapes, 2014. Accessed: 30/12/2020.

[38] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired Image-to-Image Translation using

Cycle-Consistent Adversarial Networks, 2020. Accessed: 30/12/2020.

# Appendix A Self-appraisal

<This appendix must contain everything covered under the ’self-appraisal’ criterion in the mark scheme. Although there is no length limit for this section, 2-4 pages will normally be suﬃcient. The format of this section is not prescribed, but you may like to consider the following sections and subsections.>

## A.1 Critical self-evaluation

## A.2 Personal reﬂection and lessons learned

**Self-Appraisal**

I was pleased with the outcome of this project. I felt like I had achieved the goals that I had

set out to accomplish, despite the task being a challenging one. The hardest part of the project

for me was the background research, since I was not used to reading academic papers

(particularly the ones that were mathematically dense). Fortunately none of the risked

identified in the Risk mitigation section applied to me over the course of this project and

overall I felt like the current lockdown due to the Coronavirus pandemic was beneficial to me

because it allowed me to focus on my studies.

## A.3 Legal, social, ethical and professional issues

<Refer to each of these issues in turn. If one or more is not relevant to your project, you should still explain *why* you think it was not relevant.>

### A.3.1 Legal issues

<Discussion of legal issues>

Distributed machine learning can greatly improve the processing performance of the machine, and on this basis, improve the speed of processing and analyzing big data so as to tap more value in big data, but due to the influence of the whole research in the process Too small and not in direct contact with people, making it difficult to have any meaningful impact at the legal level.

### A.3.2 Social issues

### <Discussion of social issues>

Distributed machine learning can greatly improve the processing performance of the machine, and on this basis, improve the speed of processing and analysing big data so as to tap more value in big data, but due to the influence of the whole research in the process Too small and not in direct contact with people, making it difficult to have any meaningful impact at the social level.

### A.3.3 Ethical issues

### <Discussion of ethical issues>

Distributed machine learning can greatly improve the processing performance of the machine, and on this basis, improve the speed of processing and analysing big data so as to tap more value in big data, but due to the influence of the whole research in the process Too small and not in direct contact with people, making it difficult to have any meaningful impact at the ethical level.

**Ethical concerns**

**Social Implications**

Automation can lead to rapid change in job markets and otherwise change people’s lives

significantly. The scope of this project is too small to drastically make any meaningful impact

on a societal level.

**Privacy Concerns**

This project will be using images of textures using under the creative commons or with the

permission of the creators. No infringement on people’s human right to privacy will take place as no image of people will be used.

### A.3.4 Professional issues

<Discussion of professional Issues>

Distributed machine learning can greatly improve the processing performance of the machine, and on this basis, improve the speed of processing and analysing big data so as to tap more value in big data, but due to the influence of the whole research in the process Too small and not in direct contact with people, making it difficult to have any meaningful impact at the professional level.

# Appendix B External Materials

<This appendix should provide a brief record of materials used in the solution that are not the student's own work. Such materials might be pieces of codes made available from a research group/company or from the internet, datasets prepared by external users or any preliminary materials/drafts/notes provided by a supervisor. It should be clear what was used as ready-made components and what was developed as part of the project. This appendix should be included even if no external materials were used, in which case a statement to that effect is all that is required.>