

陈浩_07160419_论文定稿

【原文对照报告-大学生版】

报告编号: ed2e31723c824ec

检测时间: 2020-05-25 18:43:45

检测字数: 15,821字

作者名称: 佚名

所属单位: 南京师范大学(教务处)

检测范围:

- | | | |
|------------------|-----------------|-------------------|
| ◎ 中文科技期刊论文全文数据库 | ◎ 中文主要报纸全文数据库 | ◎ 中国专利特色数据库 |
| ◎ 博士/硕士学位论文全文数据库 | ◎ 中国主要会议论文特色数据库 | ◎ 港澳台文献资源 |
| ◎ 外文特色文献数据全库 | ◎ 维普优先出版论文全文数据库 | ◎ 互联网数据资源/互联网文档资源 |
| ◎ 高校自建资源库 | ◎ 图书资源 | ◎ 古籍文献资源 |
| ◎ 个人自建资源库 | ◎ 年鉴资源 | ◎ IPUB原创作品 |

时间范围: 1989-01-01至2020-05-25

检测结论:

全文总相似比	=	复写率	+	他引率	+	自引率	+	专业术语
5.32%		5.32%		0.0%		0.0%		0.0%

其他指标:

自写率: 94.68%

专业术语: 0.0%

高频词: 盲文, 开发, 我们, 识别, 学习

典型相似性: 无

指标说明:

复写率: 相似或疑似重复内容占全文的比重

他引率: 引用他人的部分占全文的比重, 请正确标注引用

自引率: 引用自己已发表部分占全文的比重, 请正确标注引用

自写率: 原创内容占全文的比重

专业术语: 公式定理、法律条文、行业用语等占全文的比重

典型相似性: 相似或疑似重复内容占互联网资源库的比重, 超过30%可以访问

总相似片段: 50

期刊: 0 博硕: 34 外文: 0 综合: 0 自建库: 0 互联网: 16

颜色标注说明:

- 自写片段
- 复写片段（相似或疑似重复）
- 引用片段
- 专业术语（公式定理、法律条文、行业用语等）

摘要

在生活的多数情形中，视障人士或盲人与有视力的人交流是很重要的。盲文系统是最著名的方法，它创建了视障人士可以通过触摸来读取的文档格式。盲文系统是针对全盲者或视障者所研制，基于盲文单元，用来替代普通文本信息以供盲人进行书面交流的工具。盲文单元由一系列凸起的点组成，可以由他们的手指进行读取识别。这些点一般是用手工机器或特殊打印机在厚纸上压印而成的。

因此，一个能让盲文文档转换为自然语言文本的系统将会变得非常有用，特别是对于盲文使用者以及那些想要与盲人交流却无法阅读盲文字符的人。此外，这些系统也能使盲人能够轻松阅读盲文文档，并以低成本将其保存在较小的存储空间中。

本课题致力于开发一个针对于汉语的盲文文档转换为正常汉语的拼音字母文本的安卓应用程序。目的是服务于中国本土环境下的视障者，让他们能利用普及化的智能终端更方便快捷地阅读盲文文档，也能帮助社会上许多关心视障人群的人更简捷地理解盲文，更好地与视障人士进行沟通交流。

关键词：图像识别；盲文识别；机器学习；安卓应用

Abstract

In most situations of daily life, it is important for visually impaired or blind people to communicate with people with vision. The Braille system is the most well-known method. It creates a document format that visually impaired people can read by touch. The Braille system is a method that enables the blind or visually impaired to write and read through the concept of Braille cells. Braille cells consist of a series of raised dots that can be read and recognized by their fingers. These points are generally stamped on thick paper with a manual machine or a special printer.

Therefore, a system that can convert Braille documents to natural language text will become very useful, especially for Braille users and those who want to communicate with the blind but cannot read Braille characters. In addition, these systems enable blind people to easily read Braille documents and save them in a small storage space at low cost.

This subject is dedicated to the development of an Android application for converting Chinese Braille documents to Pinyin alphabetic text in normal Chinese. The purpose is to serve visually impaired people in China's local environment, so that they can use popularized smart terminals to read Braille documents more conveniently and quickly, and also help many people in the society who care about visually impaired people understand Braille more simply and better. To communicate with the visually impaired.

Keywords: Image recognition; Braille recognition; Machine learning; Android application

目录

摘要1

Abstract2

第1章 前言5

1.1 简介5

1.2 参考文献综述6

1.3 小结7

第2章 盲文识别8

2.1 盲文系统8

2.2 盲文识别8

2.2.1 盲文图像的采集及预处理8

2.2.2 预处理图像识别9

2.3 小结9

第3章 机器学习10

3.1 前瞻10

3.2 概念与分类10

3.3 神经网络11

3.4 深度学习12

3.4.1 历史发展12

3.4.2 解决方案与框架13

3.4.3 TensorFlow的版本更迭13

3.5 模型训练13

3.5.1 训练图集13

3.5.2 建立模型14

3.5.3 转换模型16

3.6 小结16

第4章 安卓开发17

4.1 安卓系统17

4.1.1 安卓简介17

4.1.2 系统优势17

4.2 Flutter框架18

4.2.1 Flutter简介18

4.2.2 Flutter特点18

4.3 开发准备19

4.3.1 环境配置19

4.3.2 资源准备19

4.3.3 使用第三方库20

4.4 编译运行22

4.4.1 代码编写22

4.4.2 编译打包22

4.4.3 结果展示23

4.5 小结24

第5章 总论25

5.1 论文总结25

致谢26

参考文献27

第1章 前言

1.1 简介

书面信息在我们的日常生活中起着不可否认的重要作用。从教育和休闲到笔记和信息交换，以符号形式编码的信息的记录和使用都是必不可少的。视障人士（盲人或患有视力损害的人）在这方面面临明显的劣势。为了满足这一需求，盲文作为在视障人士中使用最广泛的书写方式应运而生。自1829年成立以来，盲文和盲文媒体的生产以及印刷材料向盲文的转录都取得了长足的发展。然而，尽管现在盲文文档的产生相对容易，但是仍然存在将盲文文档转换为计算机可读形式的难题。这并不是一个小问题，首先，大量的仅在盲文中存在的书籍和文档，与其他稀有的旧文档一样，正在消失，必须加以数字化保存。其次，每天都需要复制（相当于影印）盲文文档并翻译盲文文档以供非盲文用户使用。后一种应用非常重要，因为它构成了视力正常者和视力障碍者之间书面交流的基础。

目前，国内外的盲文识别研究都有所建树，由于图像处理、计算机视觉等领域的进步与突破，连带着盲文识别系统也有了许多改进与创新。再加上如今大火的人工智能与深度学习，盲文识别的识别率将有质的飞跃。

软件与算法不再是障碍，硬件的设计也开始崭露头角，许多嵌入式设备或手持便携性设备纷纷面世，虚拟的算法终于有了实体的搭载。。

本课题的主要研究内容为能对盲文文档进行图像识别，并转换为正常汉语的拼音字母文本来进行存储或二次处理的盲文识别系统在搭载安卓系统的终端上的封装与实现。本课题将研究盲文识别系统在智能终端上实现的有效性 with 易用性，旨在完成从（盲文）文档到图像再到（汉语）字母文本的转换流程，并解决其中发生的图像识别等问题。

本课题开发的安卓应用程序拟利用基于端到端的开源机器学习框架TensorFlow的训练模型，并按照实际需要使用其中专门针对终端等设备特别优化的简化库，TensorFlow Lite。其打破了平时人们对深度学习的固有印象——许多时候机器学习皆是与服务器、集群等名词捆绑在一起的，现在我们触手可及的普通智能终端就能完全胜任这项工作，这无疑方便了我们，让我们能更快捷地享受深度学习的福利。据此实现盲文图像的处理与识别，借助深度学习提高盲文识别系统的识别率。紧接着，我们需要将盲文识别系统的逻辑实现代码进行封装，原生开发安卓应用程序会显得略繁琐，因此我们选用了Flutter框架进行辅助开发，其为Google公司全新推出的跨平台框架，与以往的虚拟机实现不同，Flutter框架并不是在原生开发的基础上套一层壳子，因此它能快速在移动终端上构建质量优秀的原生界面，甚至能与对应的原生代码混合开发，真正兼顾了开发效率与运行效率。Flutter组件还采用了现代响应式框架构建并拥有毫秒级的热重载等重磅功能。

本研究课题主要是开发出仅借助智能终端本身的硬件，在普通安卓智能终端上便能对盲文文档进行盲文识别并转换为正常汉语的拼音字母文本的安卓应用程序。

逻辑底层代码将利用TensorFlow lite的预训练模型完成从终端采集的盲文图像的识别与处理；UI框架将利用Flutter快速实现，保证在终端上的良好运行，也利用其包含的跨平台特性，为以后程序向各平台的移植提供可行性。

1.2 参考文献综述

逻辑底层代码将利用TensorFlow lite的预训练模型完成从终端采集的盲文图像的识别与处理；UI框架将利用Flutter快速实现，保证在终端上的良好运行，也利用其包含的跨平台特性，为以后程序向各平台的移植提供可行性。

盲文图像的采集到识别让许多学者付出了心力，传统基于图像处理而形成的OBR（光学盲文识别系统）[1, 2, 3]，在图像采集阶段，盲文摄取为原始图像，之后对其进行预处理，使得图像特征更易提取，进而转换为盲文单元，之后再把识别出的盲文翻译成正常文字，进而可以二次创作[4]。从单面到双面[5]，效率提升的同时，精确度就有了更高的要求，例如其中图像识别的步骤就有着许多突破。从规则提取[6]到学习利用细胞神经网络（邻域分析）[7]，最后是突破性地使用了向量机，在识别领域完成了新的尝试[8]。因此，这些进步标志着学者在传统领域中引入了深度学习的概念[9, 10]，突破了传统的光学盲文识别系统的限制，利用机器学习模型堆叠去噪编码器，不再采用人工的方式对识别图像进行手动选取，这种更加智能化的方式能够更深入地去提取样本特征，破除了传统人工

筛选图像特征的限制，此外，这种做法还解决了之前传统方法中因为随机选取初值而导致的局部极值漩涡，这与其选用深度学习的特征提取方法是分不开的。

在国内，盲文识别不是一个新鲜事物，它与图像处理等方面关系紧密，近年来大火的深度学习也成为助力，许多研究者都有所建树。例如，吉林省自然科学基金上有基于图像处理的盲文自动识别系统的项目，研究方向的基调是图像处理，而最终目的还是完成盲文的识别，换言之，即是采集盲文单元为原始图像，之后通过图像预处理，进一步提取图像特征，之后进行机器翻译，把获得的盲文单元翻译为普通文字[1]；一位国内的硕士研究生在了解到深度学习的概念之后，提出了一种突破传统的盲文识别的方法，以往利用人工完成的去噪、提取特征等操作，都将改为批量训练的深度模型，从而解决以往棘手的特征提取等难题 [9]。

在国外，盲文识别同样是学术界的宠儿。2006年IEEE神经网络程序国际联席会议期间，学者M. Namba和Z. Zhang提出了关联记忆的细胞神经网络及其在盲文图像识别中的应用，旨在利用细胞神经网络的关联记忆（CNN）在模式识别的有效性，创造了一种改进的邻域设计方法，以此进一步建立盲文识别系统[7]；在第二届创新计算、信息与控制国际会议（ICICIC 2007）上张尚军和河野洋平两位学者提出了带嵌入式摄像头的手机盲文识别系统[11]，可识别照片中的盲文字符，并由“DoJa I-appli工具包”开发的Java程序处理，以提供实时回复。

在科技技术蓬勃发展的今天，手机成为了最普及的智能终端之一，其中七成以上的智能手机搭载了安卓系统。因此，安卓应用程序的开发[12, 13]成为了推广普及实验成果的有效手段。而时至今日安卓应用程序的开发也不再局限于原生开发，许多成熟的框架诸如Flutter、React-native等能助力我们更好更快地构建易用有效的UI框架，让我们把更多的精力放在底层逻辑的代码实现上。Flutter框架基于Dart语言开发，相比较原生开发所钦定的Java/Kotlin语言，它综合了其余语言的长处，巧妙地利用不同的编译器完成各时段的编译、调试工作。当它处于JIT模式时，它能将源代码迅速编译完成，以达成热重载的功能实现，辅助开发者能够更好地完成调试工作；当它处于AOT模式时，它将源代码翻译为对应平台的原生代码，使成品能够拥有相当高的运行效率，足以媲美原生代码[14]。此外，Flutter框架的形成不仅是为了原生开发的快速实现，也是为了迎合跨平台的发展趋势[15, 16]，减少甚至去除各平台设计开发的隔阂。

在以前，实验室的算法只能在开发环境中有效运行。现在，一切都在技术的推进下悄然发生了变化，想法落地，我们有了更便捷的高速通道。普及化的智能终端设备不仅带来了信息沟通的便捷，也建立了从研究者到百姓的高速通道，让更多人不用花费额外的成本去享受新兴技术。

1.3 小结

国内外对盲文识别的研究已经具有了较为完善的系统，在基于图像处理的计算机视觉方面拥有较成熟的算法，在锐意创新方面，引入了深度学习的概念，以提升图像识别的识别率阈值。然而，盲文识别仍缺乏普及的低门槛平台，无法很好地落实到需要人群中。本课题致力于开发一个针对于汉语的盲文文档转换为正常拼音字母文本的安卓应用程序。在查阅各类文献资料的同时，我清楚地感知到一群孜孜不倦学者的努力与艰辛，也感受到了蕴藏在学术里的人文关怀。或许这些学士能轻易地用一双明眼去发现这广袤世界的许多奇妙，但他们也没有忘却自己身边仍有一些人缺失了这份能力，付出心力去建造我们与他们沟通的桥梁。站在巨人的肩膀上，我也想献出一份绵薄之力，让他们拥有科技的眼睛，去更好地观察这个世界。

第2章 盲文识别

2.1 盲文系统

盲文系统是针对全盲者或视障者所研，基于盲文单元，用来替代普通文本信息以供盲人进行书面交流的工具。盲文单元由一系列凸起的点组成，可以由他们的手指进行读取识别。这些点是用手工机器或特殊打印机在厚纸上压印而成的。

每个盲文单元或字符都包含六个点的位置，这些位置的排列和编号如图1所示。每个点都可以在这六个位置中的任意位置上压花或凸起。因此，可以使用六个位置表示64个组合。重要的是，圆点的大小和位置要一致，以便读者容易理解字符。盲文有不同的书写方法，比如利用书写等级一进行书写，一个盲文单元可以用来表示字母、数字、标点符号或空间符号。然而若是利用书写等级二进行书写，一个盲文单元可能代表一个词的缩写，甚至整个词。



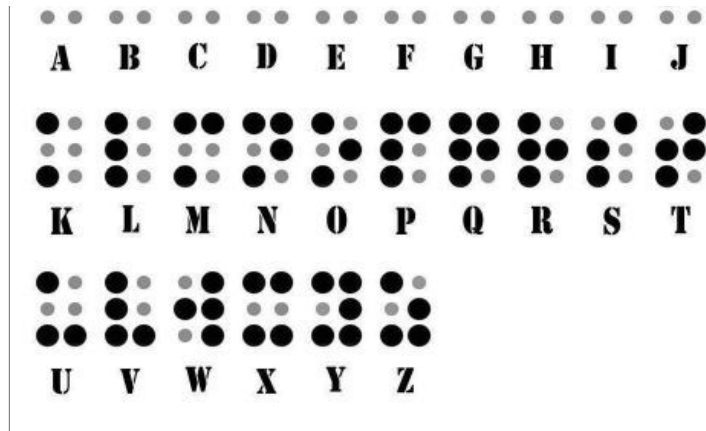


图2-1 盲文单元或字符

2.2 盲文识别

2.2.1 盲文图像的采集及预处理

分析硬件设备对盲文进行原始数据采集过程，所得的图片文件往往不尽如人意。换言之，我们需要严格规定用于识别的输入数据，以获得更高的盲文识别率。显然一味地要求摄影者的细心是不现实的，更合适的方法是对数据进行预先的处理，这里可以用到各种图像处理的手段，目的都是为了使输入数据能够拥有更鲜明的数据特征。同时，为了在实验过程中实现算法的修正，我们还得进行诸如去噪的手段，在兼顾效率的情况之下获取阈值，进而得到最适合的算法。

2.2.2 预处理图像识别

经过预处理操作后，我们采集的原始数据已经变得更适宜提取图像特征，然而对于盲文图像，我们还需精确获得图像中每个盲文单元的位置数据。换言之，自动定位算法是亟待解决的问题，关乎整个系统的完成。所以我们需要定位每个盲文单元在图像上的位置，进而提取“点”的特征，从而生成基于盲文单元的文本文件，进而通过盲文规则将此文件转换为对应字符，甚至相应声调所代表的正常含义。

2.3 小结

传统的光学盲文系统已经有相当数量的学者已经为此付出了巨大心力，在此基础上囿于硬件、算法等限制，很难再有激动人心的突破。但是时代在发展，现下大热的人工智能技术有望成为这一方面全新的生力军，通过这一新兴技术，我们可以更轻松地完成对盲文图像的识别，以达到更高的识别率，实现新的突破。

第3章 机器学习

3.1 前瞻

近年来，深度学习成为了热门词，一时间圈内圈外都对这个充满魔力的名词趋之若鹜，进而与之紧密相联的人工智能也同样获得了长足的发展，智能化、类人化等标签一个接着一个浮出水面。

从现实中的案例出发，前几年引发了全球热潮的Alpha Go在围棋比赛上战胜了柯洁，后者可称之为当代名列前茅的棋手，但也正因如此的身份，更衬托出Alpha Go的惊艳；将视线从传统弈术转到电子游戏上，OpenAI Five在Dota2比赛里同样胜出，对面是手捧冠军奖杯的OG。最后把注意力放在日常的点点滴滴，刷脸、语音助手、自动翻译等实用性更强的功能实装到了我们触手可及的各式设备中，可以说人工智能已经悄然融入了我们的生活中。虽然我们心中对于[人工智能的初印象——人工通用智能\(Artificial General Intelligence\)](#)，现今还只能存在于影视作品中，但说一句人工智能的蓬勃发展是无可厚非的。

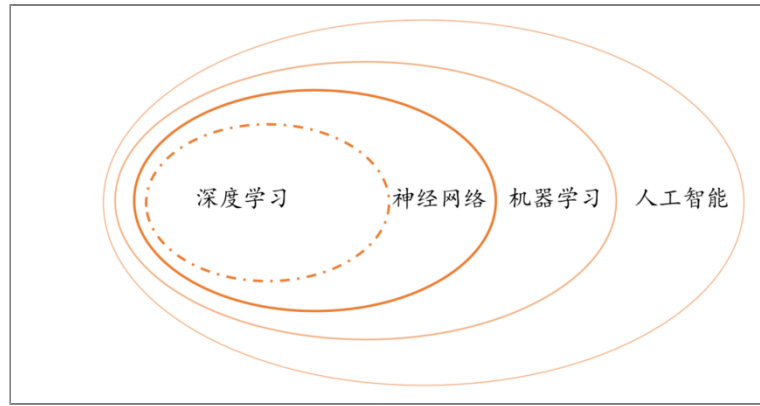


图3-1 深度学习的层级图

3.2 概念与分类

机器学习作为一个比较宽泛的概念，可以往下细分三个领域，分别是有监督学习(Supervised Learning)、无监督学习(Unsupervised Learning)和强化学习(Reinforcement Learning)，以下是各领域的简要描述，其分支图如图 3.2 所示。

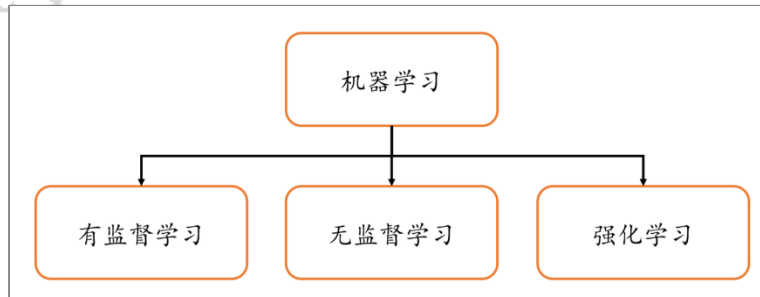


图3-2 机器学习的分类

(1) 有监督学习 (supervised learning)

对比其他领域，在进行有监督学习时拥有一个前提，即敲定结果变量。只有满足这个前提，其才能通过算法剖析特征与结果之间的联系或映射。

在此过程中，我们事先明确输入数据所对应的正确结果，而且两者的映射关系是符合一定规律的。换言之，我们希望机器能够理解既存的规律，并且此过程是完全自主化的。

(2) 无监督学习 (unsupervised learning)

相较于我们第一个介绍的类型，无监督学习没有必需的前提，相对应的，每一个输入数据也不是对应唯一的正确结果。换言之，无监督学习是尝试提取输入数据的特征，进而分析这些提取的特征。

因此，无监督学习可以达成以下两个目标

- 1) 聚类：通过特征提取分析出拥有类似或相同特征的对象，进而将输入数据归纳分类。
- 2) 密度估计：分析输入样本的分布疏密，进而估计数据集的密度和分组。

此外，利用无监督学习，我们还能达成数据的降维，从而将输入数据集以二维或三维图形的方式进行描述，使我们理解掌握起来更具有直观性。

(3) 强化学习 (reinforcement learning)

强化学习相较其他领域拥有不同于输入输出的两点模式，取而代之的是动作、状态、激励和环境四个关键词。

换言之，在这四个名词的交互循环中，机器将自动调整自己的决策，类比人类的思考与谋划，当周遭的发展符合预期时，决策结果才可称之为正确。而在这个过程中，思考的优化才是强化学习的重点。

3.3 神经网络

神经网络算法听起来有些玄乎，但究其本质，它终归只是一种分析数据的算法。给它披上神秘外衣的是神经网络这个名词，其脱胎于我们人类自己的神经系统，即用机器模拟我们人类的思考和分析方式。起初技术所限，神经网络一开始的层数较少，模拟和表述比较糟糕。之后随着技术的迭代以及数据分析的发展，深层神经网络开始问世，并在当年的项目中大放异彩。随后人们看到了其潜在的无穷力量，愈来愈多的人开始致力于神经网络的构建及其算法的改进。到现在，我们通常认为利用深层神经网络进行的各类计算工作便是深度学习，换言之，神经网络算法和深度学习，在某种程度上来说，是可以等同的。

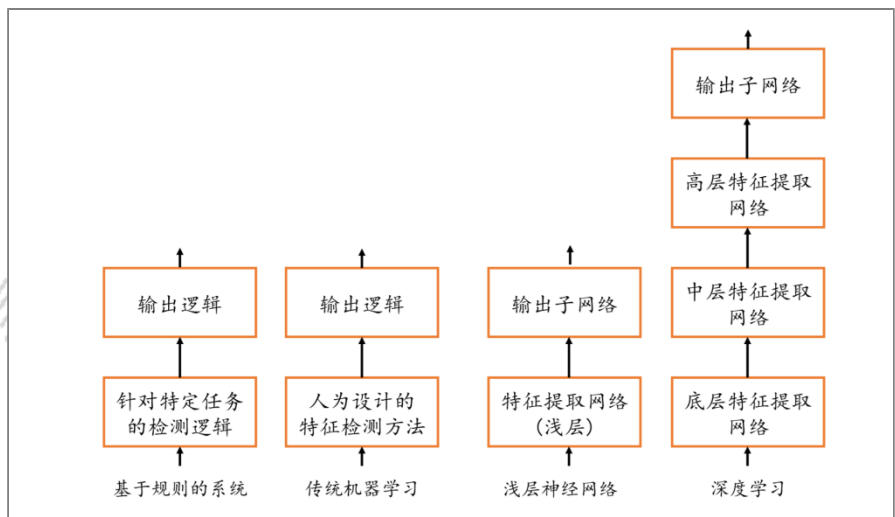


图3-3 算法分类

3.4 深度学习

3.4.1 历史发展

深度学习这个概念是由Geoffrey Hinton在其论文中点出的，然而我们在上一节提到过深度学习在某种意义上可以和深层神经网络划等号，因此我们在讨论深度学习这个概念时，不得不感恩各位学者推出自己神经网络所付出的心力。图3-4列举了一些历史上比较有分量的深层神经网络，我们可以看到，在不到二十年的时间里，深度学习一直都在稳步发展着，其中性能和效率是算法更新迭代的重要指标。

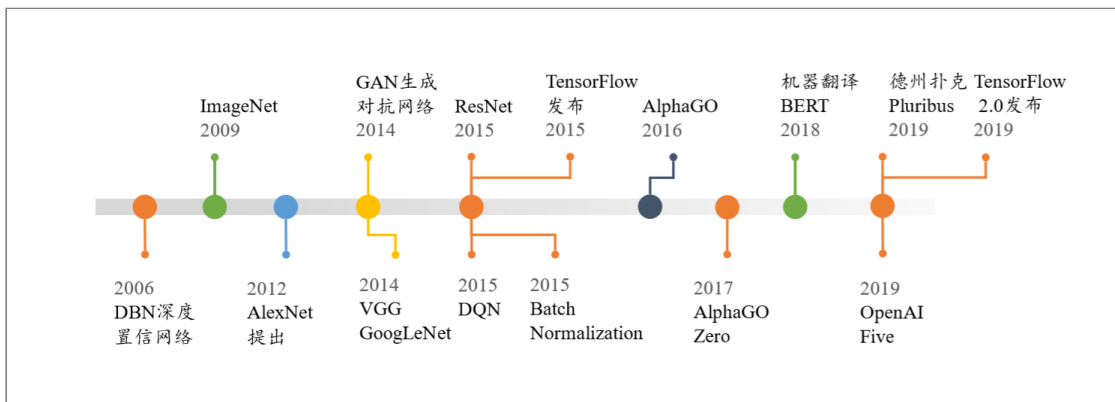


图3-4 深度学习历史

3.4.2 解决方案与框架

发展到现在，深度学习渐渐趋于成熟，对开发者或企业也愈加友好。纵观现在百花齐放般的繁荣景象，其中最大最艳的两朵花无疑是TensorFlow和PyTorch。两者各有千秋，前者在企业或工程庞大的领域有着更加完善的解决方案，同时也拥有更稳定的用户基础；后

者由于其简明的设计概念，使得开发者能够轻松上手，进而迅速构建属于自己的神经网络模型，无愧于学术界的宠儿。

此外，TensorFlow迭代之后弥补了自己入门门槛高的短板，使得开发者能够更轻松地了解与学习，也为以后与工业系统的接轨成为了可能。



图3-5 深度学习主流框架

3.4.3 TensorFlow的版本更迭

TensorFlow 2与其说是TensorFlow的迭代更新，倒不如说是另外一个独立产品的推出。一开始2代不兼容1代的缺点被人无限诟病，其从风格到接口的大变化让之前习惯于TensorFlow1.x的开发者感到了不适。因此，之前1代的代码若想迁移到2代将是个比较痛苦的过程，它依赖开发者的亲自介入和修改。

然而这样破釜沉舟般的更新并不是为了破坏用户的体验，相反地，正因为这样大刀阔斧的更新，使得TensorFlow之前冗杂的开发流程得以相当程度的简化，使得开发者的开发效率得到了极大的提升。此外2代同时也支持1代的静态图模式，以兼顾开发效率和运行效率。

3.5 模型训练

3.5.1 训练图集

机器学习需要从数据中学习，因此首先需要采集大量的真实样本数据。如图3-6所示，我们需要收集大量的A~Z的盲文图像，为了便于存储和计算，通常将准备的训练集图像统一为相同的尺寸，之后开始输入训练，在本次研究中我们将原始图像的大小固定为64个像素的行和64个像素的列(64×64)。在处理训练集图像时，除开固定尺寸外，我们还需将这些图像贴标签——即每幅图像真正的值，帮助我们辨清该幅图像的具体分类。有时候在简单分类的情况下，我们会使用数字代替具体的类别名，方便之后的处理和编程。例如分类人像图片的性别，我们可以用0代替“男性”分类，用1代替“女性”分类，当然，这个数字并不固定，用1或2同样能够达到预期效果。这种数字映射类别的方法称为数字编码。对于盲文图像识别问题，编码更为直观，我们用数字的0~25来表示类别名字为A~Z的图片。

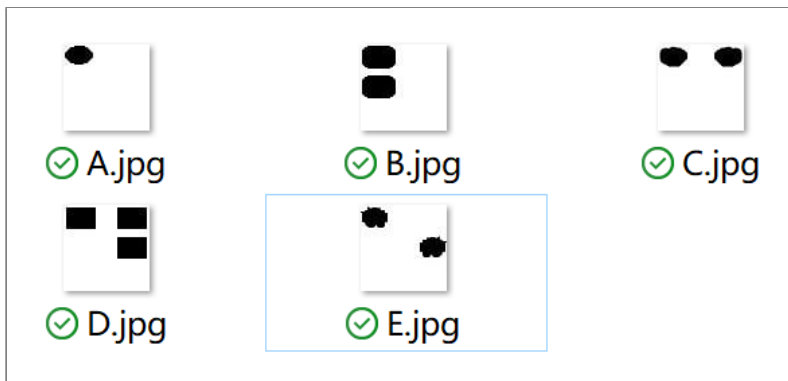


图3-6 盲文图像样例

3.5.2 建立模型

利用神经元模型构建神经网络，通过Python程序完成训练，如图3-7所示。

```
# 0oss计算
# 传入参数: 0narrs, 网络计算输出值, 0nnp, 真实值, 在这里是0或者1
```

```
# 返回参数: loss, 损失值
def losses(logits, labels):
    with tf.variable_scope('loss') as scope:
        cross_entropy = tf.nn.sparse_softmax_cross_entropy_with_logits(
            logits=logits, labels=labels, name='xentropy_per_example')
        loss = tf.reduce_mean(cross_entropy, name='loss')
        tf.summary.scalar(scope.name + '/loss', loss)
    return loss

# loss损失值优化
# 输入参数: loss, learning_rate, 学习速率。
# 返回参数: train_op, 训练op, 这个参数要输入sess.run中让模型去训练。

def training(loss, learning_rate):
    with tf.name_scope('optimizer'):
        optimizer = tf.train.AdamOptimizer(learning_rate=learning_rate)
        global_step = tf.Variable(0, name='global_step', trainable=False)
        train_op = optimizer.minimize(loss, global_step=global_step)
    return train_op

# 评价/准确率计算
# 输入参数: logits, 网络计算值。 labels, 标签, 也就是真实值, 在这里是0或者1。
# 返回参数: accuracy, 当前step的平均准确率, 也就是在这些batch中多少张图片被正确分类了。

def evaluation(logits, labels):
    with tf.variable_scope('accuracy') as scope:
        correct = tf.nn.in_top_k(logits, labels, 1)
        accuracy = tf.reduce_mean(tf.cast(correct, tf.float16))
        tf.summary.scalar(scope.name + '/accuracy', accuracy)
    return accuracy
```

图3-7 Python程序

编译运行后开始进行训练, 以构建所需的模型文件, 如图3-8所示。

```
Step 350, train loss = 0.02, train accuracy = 100.00%
Step 360, train loss = 0.02, train accuracy = 100.00%
Step 370, train loss = 0.01, train accuracy = 100.00%
Step 380, train loss = 0.03, train accuracy = 100.00%
Step 390, train loss = 0.02, train accuracy = 100.00%
Step 400, train loss = 0.01, train accuracy = 100.00%
Step 410, train loss = 0.02, train accuracy = 100.00%
Step 420, train loss = 0.02, train accuracy = 100.00%
Step 430, train loss = 0.01, train accuracy = 100.00%
Step 440, train loss = 0.01, train accuracy = 100.00%
Step 450, train loss = 0.01, train accuracy = 100.00%
Step 460, train loss = 0.01, train accuracy = 100.00%
Step 470, train loss = 0.01, train accuracy = 100.00%
Step 480, train loss = 0.01, train accuracy = 100.00%
Step 490, train loss = 0.01, train accuracy = 100.00%
```

图3-8 训练输出

训练完成后得到如下的模型文件 (如图3-9所示)。

- checkpoint
- events.out.tfevents.1557410642.USER-PC
- events.out.tfevents.1557410675.USER-PC
- events.out.tfevents.1584600034.74ad3febe...
- events.out.tfevents.1584600140.74ad3febe...
- events.out.tfevents.1584604422.74ad3febe...

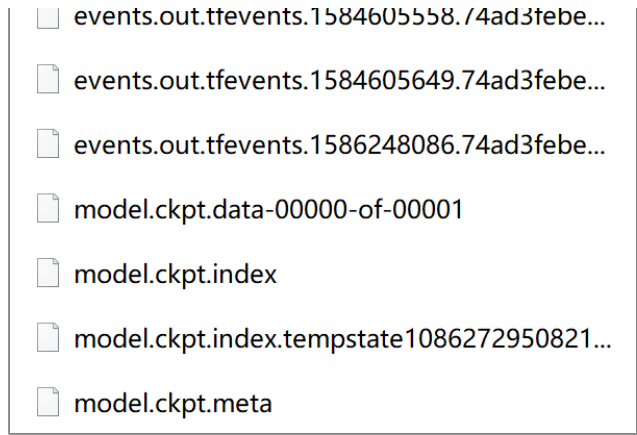


图3-9 模型文件

3.5.3 转换模型

利用Python程序生成的模型文件无法直接在移动终端中使用，需要将其进行转换。TensorFlow Lite是专门针对终端等设备特别优化的简化库，其打破了平时人们对深度学习的固有印象——许多时候机器学习皆是与服务器、集群等名词捆绑在一起的，现在我们触手可及的普通智能终端就能完全胜任这项工作，这无疑方便了我们的，让我们能更快捷地享受深度学习的福利。

利用TensorFlow自带的转换工具将所得的模型文件进行转换（如图3-10），最后得到可移植到安卓智能手机上的TensorFlow Lite模型文件（如图3-11），其以.tflite作为后缀名。



图3-10 模型转换器

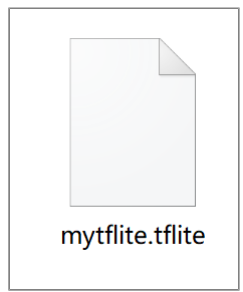


图3-11 TensorFlow Lite模型文件

3.6 小结

至此，我们得到了针对盲文图像识别的训练模型文件，并且我们还将其转换为了专门用于终端设备的TensorFlow Lite模型文件，接下来我们需要解决如何在安卓移动终端上如何调用运行它。想要迁移机器学习到移动终端设备，我们不可避免地要学习在终端系统实现应用程序的开发。在下一章节，我们将会讲述如何使用Flutter框架快速构建安卓应用程序，并在此应用程序中成功调用我们得到

的机器学习模型文件，达成在手机上实现盲文图像识别的终极目标。

第4章 安卓开发

4.1 安卓系统

4.1.1 安卓简介

Android的定义分为两部分，一个是基础，即Linux，另一个是本质，即原始码操作系统，且从开放程度来看，它是半开放的。搭载安卓系统的主要为移动终端，所以其开发和发展除开由Google主导，开放手持设备联盟也有相当的分量。

Android 系统的开拓者为Andy Rubin，他一开始的目的就只是想创造一个用于手机这个终端的智能系统，之后这个优秀的作品被Google出资收购。2007年11月，Google与多个领域的资方进行合作，联合多家公司组成开放手持设备联盟（Open Handset Alliance），目的在于对Android系统进行更好地迭代优化，随后，Google以Apache免费开源许可证的授权方式，发布了Android的源代码。让生产商推出搭载Android的智慧型电话，Android操作系统后来更逐渐拓展到平板电脑、智能电视及其他领域上。

4.1.2 系统优势

(1) 开源

相对于封闭的iOS，Android系统是开源的，换言之，所有人都可以自由地在它的基础上发挥自己的创意。但是Google作为商业公司，不可能任由自己的产品成为社区驱动的成果，因此Google依据开源条款，对Android进行了修改，添加了许多商业化必不可少的重磅功能，使其更适于在移动终端上运行，之后根据GNU条款，Google对Android的Linux内核同样进行了改进，并且此处修改信息完全透明。到现在，Google仍在不断地对Android系统进行更新迭代，使其成为始终走到时代前沿的智能手机系统。

(2) 用户群体广泛

在2010年的年底总结盘点中，以往占据榜首的霸主Sybian系统遭遇了折戟，距离问世之期不过两年的Android系统一举超越了老大哥，这个由诺基亚研发主导的老牌智能手机系统终是被新秀拍在了沙滩上。

用户群体规模大意味着基于安卓系统开发的应用程序拥有庞大的潜在用户群体，无形之中也削减了普通民众接触新兴技术的门槛。

4.2 Flutter框架

4.2.1 Flutter简介

近年来移动端的应用开发日新月异，传统的原生开发越来越难适应现时紧凑的开发节奏和频繁的迭代需求，因此市场上涌现出各式辅助开发的框架或工具。其中，Flutter作为Google新推出的技术，一经发布便引起了开发者的学习热潮。

Flutter框架基于Dart语言开发，相比较原生开发所钦定的Java/Kotlin语言，它综合了其余语言的长处，巧妙地利用不同的编译器完成各时段的编译、调试工作。当它处于JIT模式时，它能将源代码迅速编译完成，以达成热重载的功能实现，辅助开发者能够更好地完成调试工作；当它处于AOT模式时，它将源代码翻译为对应平台的原生代码，使成品能够拥有相当高的运行效率，足以媲美原生代码。

此外，Flutter框架的形成不仅是为了原生开发的快速实现，也是为了迎合跨平台的发展趋势，减少甚至去除各平台设计开发的隔阂。

4.2.2 Flutter特点

Flutter是Google新研发的用户界面框架，用于构建跨平台的用户界面，与原生代码轻松交互。

(1) 开发效率

Flutter拥有热重载功能，让开发的应用程序可在在几毫秒内焕发生机。而使用丰富的完全可定制的小部件集，甚至在几分钟内就能构建移动终端的原生界面。

(2) 界面优化

以原生终端用户体验为重点，可快速发布新增功能。分层架构允许开发者自己进行个性化更改，从而实现渲染的快速和准确以及界面设计的简洁和灵活。

(3) 原生性能

Flutter的小部件针对各平台都进行了相当的优化，使其拥有媲美原生代码的运行效率，使用户界面更加顺滑。并且开发编写的Flutter代码使用Dart的原生编译器编译成原生ARM机器代码，这意味着Flutter不存在诸如虚拟机之类的潜在损耗。

4.3 开发准备

4.3.1 环境配置

(1) Android Studio

Android Studio是Google送给Android开发者的大礼，之前繁琐的编译工具链让许多初学者望而却步，许多想法不能付诸实践。现在，一个开箱即用的集成开发环境诞生了，越来越多的人可以更加便捷地开始自己的Android应用程序开发。Android Studio严格来说不是一个独立的产品，它的大部分核心功能依托于大名鼎鼎的 IntelliJ IDEA，其作为顶级的Java集成开发环境，受到了许多开发者的热捧。Android Studio在前辈的基础上针对Android开发进行了个性化改造和特殊定制，得益于IntelliJ IDEA的跨平台特性，Android Studio同样能在各平台发光发热。

Android Studio内部预置了Android开发环境的初始化引导，你可以任意添加或卸载其中的组件。一般来说，为了满足Android开发的基本需求，我们至少要安装任意版本的Android SDK以及Platform-Tools等工具链。换言之，无论是原生开发还是构建Flutter应用程序，以上组件都是必需的。

(2) Flutter SDK

Flutter SDK包含了针对各平台或设备的开发工具链，即其为源码的封装，暴露给我们的只是用于调用的API接口。因此，在开发Flutter应用程序的时候，我们无需通读Flutter的开源代码，利用这个工具包便能轻松上手其强大的功能。

需要注意的是，开发Flutter使用的是Dart语言，但我们无需再另外配置Dart的开发环境，因为Flutter的内部已经预先集成了这个基本环境，这无疑大大方便了开发者。

(3) JDK

尽管在开发Flutter应用程序时，我们完全可以只使用Dart语言进行编写，但由于Android的原生开发依赖于java开发环境，而Flutter框架在调用类似系统底层资源时仍是通过原生代码去驱动，所以JDK还是不可或缺的。

另外，需要特别注意的，Flutter目前无法兼容最新版本的JDK，稳妥起见，我们使用作为长期维护版本的JDK8。

4.3.2 资源准备

类似于Android的原生开发，开发Flutter应用程序也需要在特定文件上声明项目内添加的本地资源，之后在编译打包的时候才能正确处理，使得程序在运行时能正常访问、调用资源文件。换句话说，我们准备移植的TensorFlow Lite模型文件是作为asset打包到我们开发的应用程序之中。

工程之中的pubspec文件即为这个特殊的声明文件，它不仅管理着本地资源的导入，还拥有着管理第三方库等其他功能，如图4-1所示。

```
pubspec.yaml x
pubspec.yaml
32 | path: ../
33 |
34 | # For information on the generic Dart part of this file, see the
35 | # following page: https://www.dartlang.org/tools/pub/pubspec
36 |
37 | # The following section is specific to Flutter.
38 | flutter:
39 |
40 |   # The following line ensures that the Material Icons font is
41 |   # included with your application, so that you can use the icons in
42 |   # the material Icons class.
43 |   uses-material-design: true
44 |
45 |   # To add assets to your application, add an assets section, like this:
46 |   assets:
```

47	- assets/mytflite.txt
48	- assets/mytflite.tflite
49	

图4-1 pubspec文件一隅

因此，我们在引入TensorFlow Lite模型文件时，需要预先在此声明文件内添加资源在项目内的相对路径。声明之后，Flutter在编译打包时会把这些本地资源统一归纳到一个特定存档中，以便在程序运行时正常调取它们。

4.3.3 使用第三方库

得益于Google的强大影响力，Flutter的生态环境极为丰富，我们可以轻松调用社区内各开发者维护的第三方软件工具包。因此，我们能够根据需求选择合适的第三方库，提高我们的开发效率。

针对TensorFlow Lite模型文件的调用和运行，我们将使用Pub 包仓库中的tflite库（如图4-2所示）。

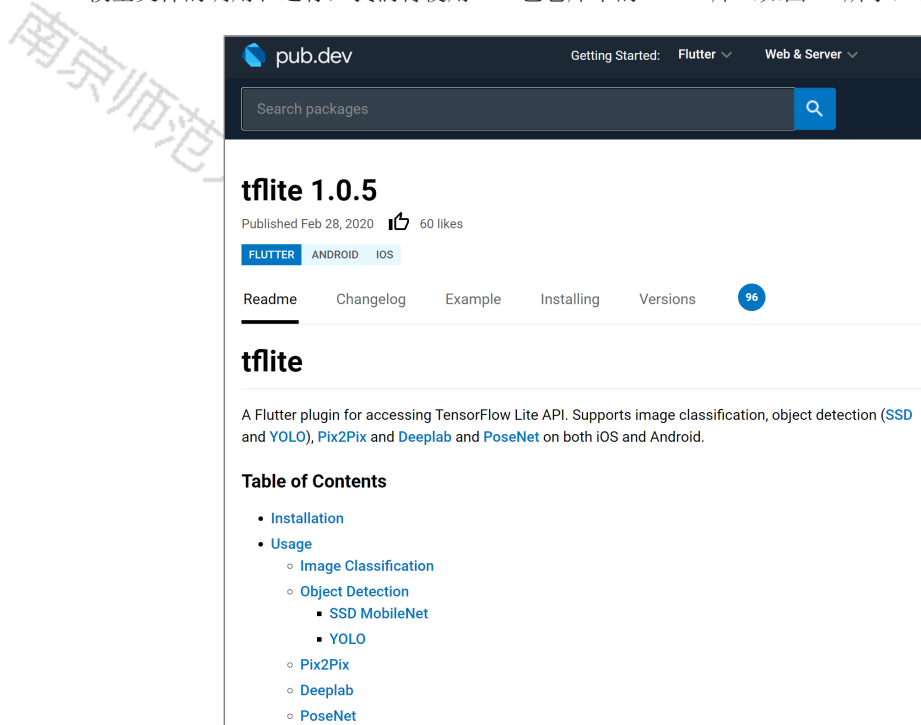


图4-2 tflite库

与引入asset类似，使用第三方库同样需要在pubspec.yaml文件中进行声明，如图4-3所示。



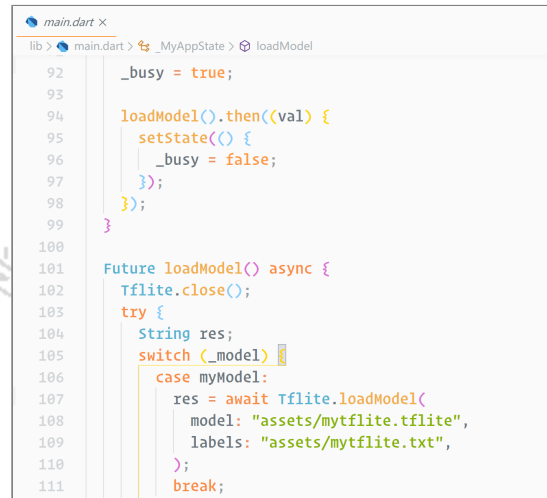
图4-3 pubspec.yaml文件引入第三方库

其中，image_picker和image库是为了实现调用手机里的系统相册以完成盲文图像的选取，在此不多做赘述。至此，我们已经基本完成了此次实验开发应用程序的前期工作。

4.4 编译运行

4.4.1 代码编写

关于在Flutter框架中如何调用TensorFlow Lite模型文件，我们参照了在Pub 包仓库中发布tflite库的作者的示例工程（如图4-4所示），实现在Flutter应用程序中调用我们自己生成的TensorFlow Lite模型文件。



```

92  _busy = true;
93
94  loadModel().then((val) {
95    setState(() {
96      _busy = false;
97    });
98  });
99  }
100
101  Future loadModel() async {
102    Tflite.close();
103    try {
104      String res;
105      switch (_model) {
106        case myModel:
107          res = await Tflite.loadModel(
108            model: "assets/mytflite.tflite",
109            labels: "assets/mytflite.txt",
110          );
111          break;

```

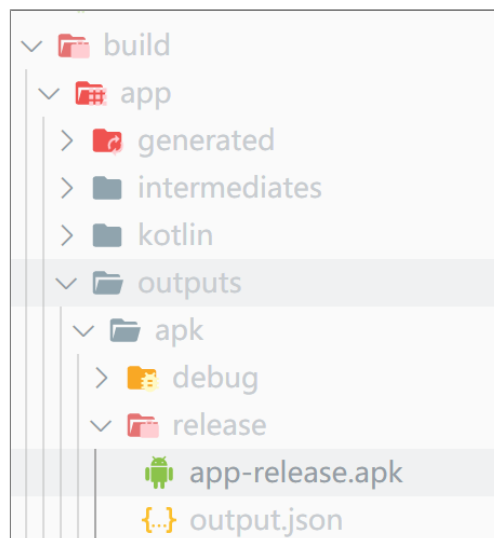
图4-4 调用mytflite文件

4.4.2 编译打包

在研发Flutter应用程序的前中期，我们在调试过程中编译生成的皆为debug级别的应用程序安装包。

而当我们已经完成了所有的调试工作，也就意味着我们可以着手构建Android应用程序的release版本了。

通过Flutter命令行输入命令或直接点击IDE内对应的功能选项，我们将会得到release级别的应用程序安装包，其生成在如图4-5所示的路径下。



```

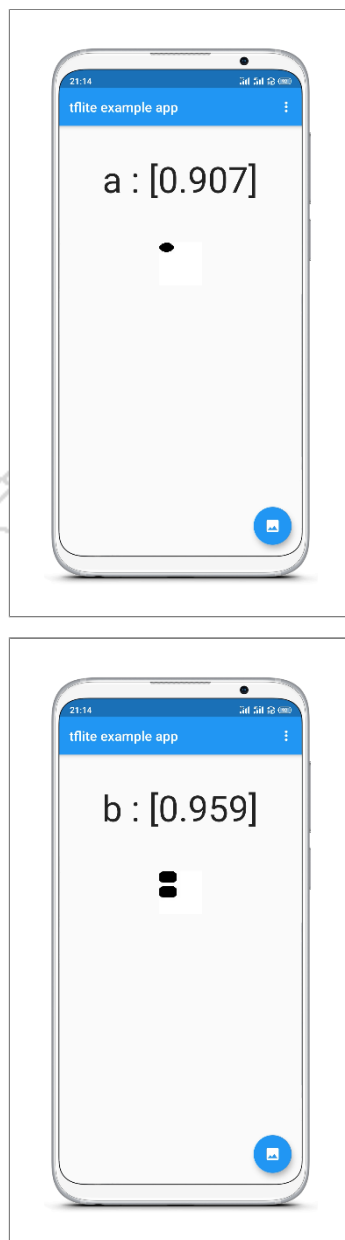
build
├── app
│   ├── generated
│   ├── intermediates
│   ├── kotlin
│   ├── outputs
│   │   ├── apk
│   │   │   ├── debug
│   │   │   └── release
│   │       ├── app-release.apk
│   │       └── output.json

```

图4-5 release版本的apk文件

4.4.3 结果展示

我们将打包好的APK文件传输到安卓真机上，本次实验所用的机型为MEIZU 16th，传输完成后立即安装运行，打开软件后传入测试用的盲文图像，界面上会显示选中的盲文图像，识别转换的字母以及其识别率，如图4-6所示。



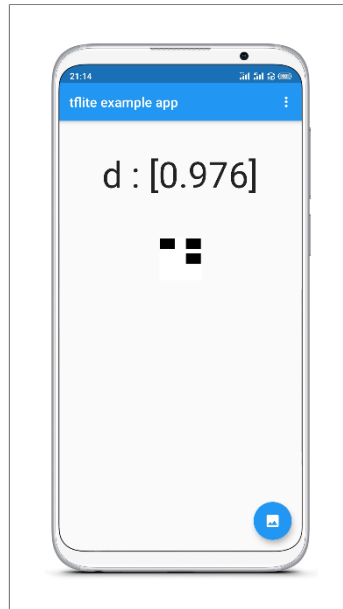


图4-6 实机展示图

4.5 小结

至此，我们基本实现了本次研究的既定目标，即完成针对于汉语的盲文文档转换为正常汉语的拼音字母文本的安卓应用程序的开发。该应用程序利用TensorFlow Lite模型文件在安卓移动终端上实现基于机器学习的盲文图像识别，可以识别规定格式的盲文图像并将其转换为对应字母文本。

第5章 总论

5.1 论文总结

在本次研究中，我们借由TensorFlow开源框架进行机器学习，训练模型以实现盲文图像的识别，并将模型文件转化为终端用的TensorFlow Lite模型文件，之后将模型文件进行移植，利用Flutter框架快速构建了可以调用运行TensorFlow Lite模型文件的安卓应用程序，达成了开发一个针对于汉语的盲文文档转换为正常拼音字母文本的安卓应用程序的最终目标。

在此次研究中，我了解并学习了有关机器学习的基础知识，掌握了利用Flutter框架快速构建安卓应用程序的方法。尽管最终成果仍有不少瑕疵，但我仍希望和我拥有同样想法的人能从这篇文章中能够获得一些启发。

致谢

岁月匆匆，为期四年的大学生活在我们不经意间如指间的流沙悄悄溜走了，手心里仅余美好的记忆。当年踌躇满志踏入校园的我，对一切都感到好奇，但同时又对很多东西感到陌生。现如今我即将踏入社会的大熔炉，面对着这四年间给予了我磨砺和呵护的校园，我的心中不由得涌起强烈的感恩与眷恋。南京师范大学这个名字，不仅仅是在这短短四年，在我以后的人生中，它都是我不可磨灭的标签。

家庭是人们最温馨的港湾。我感恩父母在我大学进修期间对我的支持与鼓励，没有了他们，我不可能安稳地完成学业。所以，纵使我们将相隔大半个地球，我们的心始终都是紧贴着的。

春蚕到死丝方尽，蜡炬成灰泪始干。大学四年，给予我最多帮助的无疑是学校里的各位老师，他们将自己的知识转化成了生动的课件和简明的语言，只是为了让我们能将这份知识继续传承下去。因此，对知识的探索，离不开导师的指引。

恰同学少年，风华正茂。四年的校园生活中，周围的同学都拥有属于自己的闪光点，学习他人的长处，共同克服各自的短板，朋友就是互相较劲，又互相进步的存在。

在此，我要着重感谢我的毕设导师何秉然老师，是他的认真负责让这篇论文的撰写更加顺利。在知识领域的探索中，他是不可或缺的

指路明灯。

最后,感谢所有鼓励我、帮助我、支持我的人,道一声感恩,道一声珍重。

参考文献

- [1] 李念峰;董迎红;肖志国. 基于图像处理的盲文自动识别系统研究[J]. 制造业自动化2012年03期, 2012.
- [2] 尹佳;李杰;王丽荣. 盲文自动识别方法研究[N]. 长春大学学报2010年08期, 2010.
- [3] S. D. Al-Shamma and S. Fathi, "Arabic Braille Recognition and transcription into text and voice," 2010 5th Cairo International Biomedical Engineering Conference, Cairo, 2010, pp. 227-231.
- [4] A. S. Al-Salman, A. El-Zaart, Y. Al-Suhaibani, K. Al-Hokail and A. O. Al-Qabbany, "An Efficient Braille Cells Recognition," 2010 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM), Chengdu, 2010, pp. 1-4
- [5] C. M. Ng, V. Ng and Y. Lau, "Regular feature extraction for recognition of Braille," Proceedings Third International Conference on Computational Intelligence and Multimedia Applications. ICCIMA'99 (Cat. No.PR00300), New Delhi, India, 1999, pp. 302-306.
- [6] M. Namba and Z. Zhang, "Cellular Neural Network for Associative Memory and Its Application to Braille Image Recognition," The 2006 IEEE International Joint Conference on Neural Network Proceedings, Vancouver, BC, 2006, pp. 2409-2414.
- [7] Jie Li and Xiaoguang Yan, "Optical Braille character recognition with Support-Vector Machine classifier," 2010 International Conference on Computer Application and System Modeling (ICCASM 2010), Taiyuan, 2010, pp. V12-219-V12-222.
- [8] 李婷. 基于深度学习的盲文识别方法[J]. 计算机与现代化2015年06期, 2015.
- [9] 李荣瑞;施霖. 基于深度学习的盲文自动识别系统[J]. 电子科技2018年09期, 2018.
- [10] S. Zhang and K. Yoshino, "A Braille Recognition System by the Mobile Phone with Embedded Camera," Second International Conference on Innovative Computing, Informatio and Control (ICICIC 2007), Kumamoto, 2007, pp. 223-223.
- [11] 郭霖. 第一行代码 Android 第2版[M]. 人民邮电出版社, 2016.
- [12] 任玉刚. Android开发艺术探索[M]. 电子工业出版社, 2015.
- [13] [美] Chris Strom. Dart语言程序设计[M]. 韩国恺, 译. 北京: 人民邮电出版社, 2013.
- [14] 亢少军. Flutter技术入门与实战 第2版[M]. 机械工业出版社, 2019.
- [15] 何瑞君. Flutter: 从0到1构建大前端应用[M]. 电子工业出版社, 2019.
- [16] 任玉刚. Android开发艺术探索[M]. 电子工业出版社, 2015.

• 说明:

相似片段中“综合”包括:

《中文主要报纸全文数据库》 《中国专利特色数据库》 《中国主要会议论文特色数据库》 《港澳台文献资源》
《图书资源》 《维普优先出版论文全文数据库》 《年鉴资源》 《古籍文献资源》 《IPUB原创作品》

• 声明:

报告编号系送检论文检测报告在本系统中的唯一编号。

本报告为维普论文检测系统算法自动生成,仅对您所选择比对资源范围内检验结果负责,仅供参考。

客服热线：400-607-5550 | 客服QQ：4006075550 | 客服邮箱：vpcs@cqvip.com

唯一官方网站：<http://vpcs.cqvip.com>



关注微信公众号