

**毕业论文(设计)**

**题目名称：乐曲演奏水平分析软件的设计与实现**

**题目类型：毕业设计**

**学生姓名：杨东泉**

**院 (系)：计算机科学学院|软件工程系**

**专业班级：软工11302**

**指导教师：向华**

**辅导教师：向华**

**时 间： 2016.11.26至2017.5.30**

目 录

[任务书 I](#_Toc482101563)

[开题报告 II](#_Toc482101564)

[指导教师审查意见 III](#_Toc482101565)

[评阅教师评语 IV](#_Toc482101566)

[答辩会议记录 V](#_Toc482101567)

[中文摘要 VI](#_Toc482101568)

[英文摘要 VII](#_Toc482101569)

[1 引言 1](#_Toc482101570)

[1.1 二级标题1 1](#_Toc482101571)

[参考文献 1](#_Toc482101572)

[致谢 1](#_Toc482101573)

[附录 1](#_Toc482101574)

任务书

开题报告

指导教师审查意见

评阅教师评语

答辩会议记录

中文摘要

# 英文摘要

**乐曲演奏水平分析软件的设计与实现**

# 1 前言

随着生活水平的提高，人们对生活的品质要求越来越高，越来越多的人更愿意培养自己的业余爱好。但是部分爱好必须在有专业人士的指导下才能进行，这就就意味着人们在保持这种爱好的同时需要付出宝贵的时间精力，这会大大打击人的兴趣。这类的的爱好最典型的就是乐曲类。这类爱好需要特殊技巧，并且需专业人士指导才能进行，若缺失指导则会事倍工半。而现实生活中找一个随时陪伴自己的专业教练，从培养兴趣的成本上来看是不切实际的。

然而得益于21世纪以来计算机技术的飞速发展，手持电子设备的性能越来越强大，功能也来越多。大多数手持Android设备具有麦克风，性能比较优秀的CPU（主频大于1000MHZ）,以及容量相当大的RAM（RAM容量大于1GB）。这使得在手持Android设备进行音频处理成为可能。

本文主要讲述乐曲演奏水平分析软件的模块构成，模块之间的联系，工作流程，设备部署，以及其核心评价模型设计的原理。

由于本文作者设计整个项目的架构以及项目内部成员分工。除了以上工作外作者主要对其中评价模型进行详细的设计，以及详细的代码实现。所以本文主要围绕项目架构，以及核心评价算法的实现展开叙述。

# 2 选题背景

## 2.1 课题来源

编写这个软件最早开始于2015年5月。因为我的导师他的女儿正在学习钢琴，但是苦于每次练习都学到很远的的钢琴培训机构，自己在家里练习弹钢琴出现错误没有专业人士及时纠正指导，导致学习钢琴进步特别慢，而且由于缺乏专业指导经常出现自己弹奏错误却又不知道而且由于多次练习错误的弹奏方法，导致很难纠正她的错误。

纵观目前软件市场，并不缺乏声音识别的软件，但是对于乐曲演奏水品分析的软件却寥寥无几。所以我们决心弥补市场空缺。

## 2.2 选题目的

通过普通的安卓设备对钢琴弹奏者弹奏的乐曲进行录音，利用软件的识音模块进行分析计算出演奏者演奏乐曲的弹奏特征（例如：键位，声音强度，声音持续时间，键的序列等等）。然后把分析的结果交付给软件的评价模块进行评价。评价模块将会根据识音模块的提供的乐曲特征对其按照一定的算法进行评价（评价算法见xxxx）。评价模块评价后的结果将会交给交互模块以及神经网络评价模块。交互模块将根据结果展示给用户看。神经网络模块将会利用评价结果来训练评价函数，以便以后提供更优的评价结果。

## 2.3 选题意义

通过对整个项目的规划，使我们对大学书本上学的的知识进行一次巩固，例如项目规划中，我们首先对软件进行UML建模，根据UML得出的模型使我们与组员间的交流更清楚明白。根据UML的模型，我们更加清楚了每个人的任务。同时我们从学习时的主要学习理论转换为从实际项目出发。

## 2.4 应解决的主要问题及应达到的技术要求

主要问题就是如何把乐曲转化为可分析的数据结构，然后利用这种数据结构对乐曲进行评价。

首先我们知道乐曲演奏是一样复杂的表演行为。他的评价标准不仅在于乐曲声的抑扬顿挫，而且经验丰富的表演者通常会将个人经历已经情感融入音乐使音乐富有表现力和感染力。然而这样就增加了机器评价音乐的难度。

好在一般来说，对于乐曲演奏的初学者来说，只要把握乐曲演奏的基本规律（如钢琴演奏时键位顺序，持续时，响度等等）。就能达到很好的学习效果。所以软件针对的人去并不是整体乐曲演奏人群。乐曲主要针对的人群为乐曲演奏的初学者，中级水平及以下的演奏者，对他们的基本的演奏技巧进行评价。以便他们在今后的学习中更快的提高自己的水平。特别是那些刚刚起步的乐曲演奏学习者有着绝佳的效果。

软件至少要达到90%的评价准确度，对于乐曲评价向来没有准确的定义。我们只是尽量软件评价的结果靠近专业的乐曲教学者给出的评价结果，相应的我们通过调整评价的算法使结果更加可靠，同时我们将利用大量的数据去训练相关的神经网络算法，将来用于更精确的评价。

## 2.5 课题研究的指导思想

我们都知道音乐是连续的量，然而目前计算机只能处理离散的量。所以我们需要将连续的音乐进行特征量提取。具体到钢琴，就是钢琴的每个键。钢琴每个键对应的不同频率的声音。在音乐上就表现出不同的按键的排列组合组成不同的乐曲，我们提取出每个按键的位置，按键发出声音的大小，按键发出声音持续时间的的长短。我们将这些特征封装成一个数据结构Key(详细结构见附录X)。键是单个按键的特征，对音乐的评价并不仅仅取决于单个键的结果。于是我们提出另一种数据额结构Music(结构见附录x)。Music概括的描述出演奏乐曲的整体特征，如Key的序列，Key的个数，音乐持续时间，音乐声音的平均大小等。不但如此，Music中还封装了便于音乐评价算法评价的成员变量I，以及成员方法Next()。Music的结构设计是的乐曲评价算法的实现成为可能。最后我们将返回的评价结果Result进行设计使得他能更恰当的描述演奏者演奏乐曲的水平。整个评价过程都立足于以上三种数据结构。三种数据结构协调的配合是评价算法有序工作，得出正确结论的基石。

## 2.6 本论文章节安排

本论文乐曲演奏水平分析软件的设计与实现核心章节安排与创新点详细如下：

第一章：前言。简要的介绍项目的范围及现状。

第二章：选题背景。说明本课题的来源、目的、意义、应解决的主要问题及应达到的技术要求。

第三章：方案论证。说明设计原理并进行方案选择，阐明为什么要选择这个设计方案(包括各种方案的分析、比较)以及所采用方案的特点等

第四章：过程(设计或实验)论述。指作者对自己的研究工作的详细表述。要求论理正确、论据确凿、逻辑性强、层次分明、表达确切。

第五章：结果分析。对研究过程中所获得的主要的数据、现象进行定性或定量分析，得出结论和推论。

第六章：结论和展望。对整个研究工作进行归纳和综合，阐述本课题研究中尚存在的问题及进一步开展研究的见解和建议。

# 3 设计论述

## 3.1 相关技术工具简述

### 3.1.1 java介绍

Java是由Sun Microsystems公司于1995年5月推出的Java面向对象程序设计语言和Java平台的总称。由James Gosling和同事们共同研发，并在1995年正式推出。

Java分为三个体系：

JavaSE（J2SE）（Java2 Platform Standard Edition，java平台标准版）

JavaEE(J2EE)(Java 2 Platform,Enterprise Edition，java平台企业版)

JavaME(J2ME)(Java 2 Platform Micro Edition，java平台微型版)。

2005年6月，JavaOne大会召开，SUN公司公开Java SE 6。此时，Java的各种版本已经更名以取消其中的数字"2"：J2EE更名为Java EE, J2SE更名为Java SE，J2ME更名为Java ME。

主要特性：

1. Java语言是简单的：

Java语言的语法与C语言和C++语言很接近，使得大多数程序员很容易学习和使用。另一方面，Java丢弃了C++中很少使用的、很难理解的、令人迷惑的那些特性，如操作符重载、多继承、自动的强制类型转换。特别地，Java语言不使用指针，而是引用。并提供了自动的废料收集，使得程序员不必为内存管理而担忧。

2. Java语言是面向对象的：

Java语言提供类、接口和继承等原语，为了简单起见，只支持类之间的单继承，但支持接口之间的多继承，并支持类与接口之间的实现机制（关键字为implements）。Java语言全面支持动态绑定，而C++语言只对虚函数使用动态绑定。总之，Java语言是一个纯的面向对象程序设计语言。

3. Java语言是分布式的：

Java语言支持Internet应用的开发，在基本的Java应用编程接口中有一个网络应用编程接口（java net），它提供了用于网络应用编程的类库，包括URL、URLConnection、Socket、ServerSocket等。Java的RMI（远程方法激活）机制也是开发分布式应用的重要手段。

4. Java语言是健壮的：

Java的强类型机制、异常处理、垃圾的自动收集等是Java程序健壮性的重要保证。对指针的丢弃是Java的明智选择。Java的安全检查机制使得Java更具健壮性。

5. Java语言是安全的：

Java通常被用在网络环境中，为此，Java提供了一个安全机制以防恶意代码的攻击。除了Java语言具有的许多安全特性以外，Java对通过网络下载的类具有一个安全防范机制（类ClassLoader），如分配不同的名字空间以防替代本地的同名类、字节代码检查，并提供安全管理机制（类SecurityManager）让Java应用设置安全哨兵。

6. Java语言是体系结构中立的：

Java程序（后缀为java的文件）在Java平台上被编译为体系结构中立的字节码格式（后缀为class的文件），然后可以在实现这个Java平台的任何系统中运行。这种途径适合于异构的网络环境和软件的分发。

7. Java语言是可移植的：

这种可移植性来源于体系结构中立性，另外，Java还严格规定了各个基本数据类型的长度。Java系统本身也具有很强的可移植性，Java编译器是用Java实现的，Java的运行环境是用ANSI C实现的。

8. Java语言是解释型的：

如前所述，Java程序在Java平台上被编译为字节码格式，然后可以在实现这个Java平台的任何系统中运行。在运行时，Java平台中的Java解释器对这些字节码进行解释执行，执行过程中需要的类在联接阶段被载入到运行环境中。

9. Java是高性能的：

与那些解释型的高级脚本语言相比，Java的确是高性能的。事实上，Java的运行速度随着JIT(Just-In-Time）编译器技术的发展越来越接近于C++。

10. Java语言是多线程的：

在Java语言中，线程是一种特殊的对象，它必须由Thread类或其子（孙）类来创建。通常有两种方法来创建线程：其一，使用型构为Thread(Runnable)的构造子将一个实现了Runnable接口的对象包装成一个线程，其二，从Thread类派生出子类并重写run方法，使用该子类创建的对象即为线程。值得注意的是Thread类已经实现了Runnable接口，因此，任何一个线程均有它的run方法，而run方法中包含了线程所要运行的代码。线程的活动由一组方法来控制。Java语言支持多个线程的同时执行，并提供多线程之间的同步机制（关键字为synchronized）。

11. Java语言是动态的：

Java语言的设计目标之一是适应于动态变化的环境。Java程序需要的类能够动态地被载入到运行环境，也可以通过网络来载入所需要的类。这也有利于软件的升级。另外，Java中的类有一个运行时刻的表示，能进行运行时刻的类型检查。

### 3.1.2 MyEclips介绍

MyEclipse企业级工作平台（MyEclipse Enterprise Workbench ，简称MyEclipse）是对[Eclipse](http://baike.baidu.com/view/23576.htm)IDE的扩展，利用它我们可以在[数据库](http://baike.baidu.com/view/1088.htm)和JavaEE的开发、发布以及[应用程序](http://baike.baidu.com/view/330120.htm)[服务](http://baike.baidu.com/view/133203.htm)器的整合方面极大的提高工作效率。它是功能丰富的JavaEE[集成开发环境](http://baike.baidu.com/view/14867.htm)，包括了完备的编码、调试、测试和发布功能，完整支持[HTML](http://baike.baidu.com/view/692.htm)，Struts，JSP，[CSS](http://baike.baidu.com/view/15916.htm)，[Javascript](http://baike.baidu.com/view/16168.htm)，Spring，SQL，Hibernate。

MyEclipse 是一个十分优秀的用于开发Java, J2EE的 Eclipse 插件集合，MyEclipse的功能非常强大，支持也十分广泛，尤其是对各种开源产品的支持十分不错。MyEclipse目前支持Java Servlet，AJAX，JSP，JSF，Struts，Spring，Hibernate，EJB3，JDBC数据库链接工具等多项功能。可以说MyEclipse几乎囊括了目前所有主流开源产品的专属eclipse开发工具。

### 3.1.3 Rational rose介绍

Rational Rose包括了[统一建模语言](http://baike.baidu.com/item/%E7%BB%9F%E4%B8%80%E5%BB%BA%E6%A8%A1%E8%AF%AD%E8%A8%80)（UML），OOSE，以及OMT。其中统一建模语言（UML）由Rational公司3位世界级[面向对象](http://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E5%AF%B9%E8%B1%A1)技术专家Grady Booch、Ivar Jacobson、和Jim Rumbaugh通过对早期面向对象研究和设计方法的进一步扩展而得来的，它为[可视化建模](http://baike.baidu.com/item/%E5%8F%AF%E8%A7%86%E5%8C%96%E5%BB%BA%E6%A8%A1)软件奠定了坚实的理论基础。同时这样的渊源也使Rational Rose力挫当前市场上很多基于UML可视化建模的工具，例如Microsoft的Visio2002、Oracle的Designer2000，还有PlayCase 、CA BPWin、CA [ERWin](http://baike.baidu.com/item/ERWin)、Sybase PowerDesigner等等。

Rational Rose 是一个完全的、具有能满足所有建模环境（Web开发，[数据建模](http://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BB%BA%E6%A8%A1)，Visual Studio和 C++ ）灵活性需求的一套解决方案。Rose 允许开发人员，项目经理，系统工程师和分析人员在软件开发周期内在将需求和系统的体系架构转换成代码，消除浪费的消耗，对需求和系统的体系架构进行可视化，理解和精练。通过在软件开发周期内使用同一种建模工具可以确保更快更好的创建满足客户需求的可扩展的、灵活的并且可靠的应用系统。

## 3.2 系统架构简介

### 3.2.1 用户交互用例介绍



图4-1交互界面的用例图

如图4-1所示。系统前端主要有三种使用者他们分别为：

1、业余演奏者。业余演奏者是指那些下载APP但并未注册账号，以游客身份登录的用户。用户本身未在学校或者固定的钢琴培训机构内部学习。线下没有专业老师指导，在软件内部也不会接受到教师所分配的练习任务。所以对于业余演奏者来说，他只是使用APP录制弹奏特征，并且查看本次弹奏所得的分数，以及与标准乐曲的差距。以此参考学习改善提高自己的演奏水平。

2、钢琴学习者。钢琴学习者是指那些下载APP并注册学习者账号的用户，其本身在学校学习或者固定的钢琴培训机构内部学习。线下有专业老师指导，在软件内部也可以接受到教师所分配的练习任务。他们是本软件的主要用户。所以对于钢琴学习者来说，除了业余演奏者可以使用的功能之外还应包括：

身份认证信息：一般为账号、密码、邮箱等等信息。

查看作业任务：主要是用户在线下的老师使用APP发布的练习任务。

查看作业历史完成情况：主要是学生个人完成老师所布置所有练习任务的统计情况。可以看出自己在练习过程中的趋势。

3、教师：教师是指钢琴学习者是指那些下载APP并注册教师账号的用户，其本身在学校学习或者固定的钢琴培训机构内部教学或者本身具有指导钢琴学习的资质。其使用的功能包括：

身份认证：区别于学习者的人份认证，教师必须上传其具有相关教学资质的证明。录制弹奏特征也不于其他用户，教师弹奏的特征将作为标准乐曲用来评价其他用户的弹奏水平。

上传作业任务：教师与钢琴学习者是一对多的关系。当教师指派练习任务时，所有的钢琴学习者都看看到任务。

查看学生完成情况：教师可以看到自己所带学生的练习任务的完成情况。

### 3.2.2 系统组件介绍

**图 4-2系统组件图**

如4-2所示系统被划分为4个组件。

1、声音采集系统：采集声音。钢琴演奏的声音将会被录制作为输入流进该组件，该组件将会对音乐进行分析，计算出演奏者所提供的键位的特征：

声音的键位，演奏者手指所触碰到的键位（为了方便评测我们队钢琴的键位进行了编号）。

声音按键的强度，该次按键所发出声音的响度。

声音按键的大致时间，演奏者在何时按下该键。

声音的频率，当前所按键位的振动频率。

1. 数学评价系统：数学评价系统是指根据声音采集系统提供的按键特征，建立的数学评价模型。根据模型将对输入的音乐进行评价。评价方案包括：

键位分析法：根据演奏者所按得键顺序（A）及键总数与给出的原键位（B）比较，得出数值。

键位时间分段分析法：根据演奏的总时间将A与B分成时间比例相等（如各占总时间1/10）的 键位段 将对应各段进行键位比较。

特征键位分段分析：根据A与B中识别特征的键，将A于B 分成相同个数的段将对应各段进行键位比较

按键强度与键位综合分析法：将按键强度因素与前三种分析方法结合。

1. 神经网络评价系统：神经网络评价系统将更具数学评价系统的评价结果作为神经网络系统的数据输入，借助大量的数据做出更精确的评价。
2. 用用户交互系统：用户交互系统目前主要接受用户相关操作。用户借此操作软件调用功能。同时展示其他模块完成的任务的结果给用户。完成类似于社区的功能，使得每个APP用户不是孤立的而是相互关联的。提高APP的用户黏度。

### 3.2.3 系统部署图介绍



图 4-3部署图

1. 移动设备：接受用户输入，如登录工作，填写用户信息的工作。以及接受用户输入。最主要的功能是乐曲水品评价的工作也是在移动设备上完成的，这样做有如下好处，首先移动设备目前的性能足够完成声音的转换，以及评价乐曲水平的计算任务。在移动设备端完成计算任务节省了数据传输的时间。降低了服务器的计算压力，使得我们可以配置廉价的用户服务器。节约部署成本。同时移动设备只用上传用户基本信息到服务器。上传每次弹奏的结果上传到神经网络处理集群，以便训练集群工作。
2. 服务器：主要与移动设备通讯，保存用户信息，用户与用户之间联系。同时服务器会提供用户社区类的服务。使得APP功能跟完整。
3. 神经网络处理集群，主要接受移动设备所传回来的有关演奏的数据。这些数据有两类，一类是老师提供的标准乐曲，其作用通常用来作为标准乐曲来评价摸个用户输入的乐曲的水平高低。第二类是是非教师用户提供的乐曲，这类乐曲经过数学评价模型评价后，把乐曲及评价结果都输入数据库，用来训练神经网络评价算法，以便日后作为更高效率的评价方式。

# 4 评价模型设计与实现

## 4.1 评价模型设计

### 4.1.1键顺序分析法KeySeq

算法思想：根据演奏者所按得键顺序（A）及键总数与给出的原键位（B）比较，得出数值。

算法步骤：

第一步：

将A的总键数An按顺序与B的总键数Bn对比。

若A，B键总数之差大大于B键总数的5%将判定为该评价方法不合适，选择采用评价规则2或评价规则3。

若A，B键总数之差小于B键总数Bn的5%将判定为使用该方案，进入第二步。

第二步：

将A的键顺序与B的键顺序对比，相同分数S1+1。否则进入第三步。

第三步：

当前比较的键位不匹配，相异的键位采用就近匹配（假设A的第29个键位与B的第29个键位不匹配，若A的键总数多，那么将A的第30个键与B的第29个键位匹配，若仍然不匹配将A的第31个键与B的第29个键位匹配，但是向后推进的键数不超过AB键数之差的1/5。超过1/5时，则开始移动B，将A的第29键与B的第30个键匹配若不匹配则将A的第29个键与B的第31个匹配向后推进的键数不超过AB键数之差的1/5），若就近匹配就近匹配失败分数S 不变，B键位比较回到原始位置，A键位位置+1，若成功S1+0.5分，从成功位置依次向后匹配.重复第二步和第三步直到A和B中某一个比较完成。

分数计算公式：

S=s1/An\*100.

算法优势：算法优势在于以最大限度的匹配来搜索出用户所按的正确键位，实现思想简单易于操作。同时也不失为一种最基本的评测方案。其他方案可以加以利用。

算法劣势：评价因素单一，仅仅取键位及其顺序作为决策的因素，精确度可能不高。

例4-1

A的键位为：a b d e f g h i k l m n o p q r s t u v w

B的键位为：a b c d e f g h i j k l m n o p q r s t

AB的键符合键位顺序分析法的前提。

按顺序比较A:c与B:d比较不匹配采用就近匹配原则来匹配附近的键位s=2。可以的得知B：c与A:e f g都不匹配。于是将A:d与B：d比较匹配成功s=2.5。接着向后匹配e匹配s=3.5。f匹配s=4.5。g匹配s=5.5。h匹配s=6.5。i匹配s=7.5。A：k与B：j不匹配就近匹配原则B：j和A:l m n都不匹配。A：k与B:k 匹配s=8。

l匹配s=9，m匹配s=10，n匹配s=11，o匹配s=12，q匹配s=13，r匹配s=14，t匹配s=15，这时B已经匹配完成。所有A所有剩余的键位都是多余的不等分。这是15/21\*100=71.42把分数转化为百分制。

### 4.1.2 按键强度与键位分析法KeySeqStrong

算法思想:将按键强度因素与键顺序分析法方法结合。其原理类似于键顺序分析法在评价的方式上把按键的强度也作为评价的因素。其中有个因素——比例因子，比例因子的主要作用为当演奏者音乐的整体音量要小于标准音乐，但是键位响度整体上跟标准响度成一定比例，比例因子就是用来区分这种差异的，将Bn每个键响度的乐曲的响度乘以比例再将两个键位比较。这样有效的屏蔽了演奏者因为整体声音小，但是旋律却是正确的情况被误判。

算法步骤：

第一步：

计算出A,B每个键强度之和As,Bs。 得出比例因子factor=As/Bs。

第二步：

将A的键顺序与B的键顺序对比，相同分数S1+1，进入第三步。若不相同进入第四步。

第三步：

将B的当前键（keyStrongB）的声音强度乘以factor与A当前键（keyStrongB）比较。若diff=keyStrongB\*factor- keyStrongA的值差别在规定的范围类那么S1=S1+x。x视diff的大小取值0-1。重复第二步直到A和B中某一个比较完成。

第四步：

当前比较的键位不匹配，相异的键位采用就近匹配（假设A的第29个键位与B的第29个键位不匹配，若A的键总数多，那么将A的第30个键与B的第29个键位匹配，若仍然不匹配将A的第31个键与B的第29个键位匹配，但是向后推进的键数不超过AB键数之差的1/5。超过1/5时，则开始移动B，将A的第29键与B的第30个键匹配若不匹配则将A的第29个键与B的第31个匹配向后推进的键数不超过AB键数之差的1/5），若就近匹配就近匹配失败分数S 不变，B键位比较回到原始位置，A键位位置+1，若成功S1+0.5分，从成功位置依次向后匹配.重复第二步和第四步直到A和B中某一个比较完成。

得分公式：

S=S1/Bn\*100/2

算法优势：算法优势在于以最大限度的匹配来搜索出用户所按的正确键位，实现思想简单易于操作。同时增加了评价音乐好坏的因素声音强度。其他方案可以加以利用。

算法劣势：当演奏者按键顺序差别较大或者演奏者有着自己独特的演奏方式（声音强度控制比较优秀，比标准音乐更富有表现力）会造成误判。

### 4.1.3 键位误差分析法KeyGap

算法思想：演奏者在演奏过程中本该按x键，但是由于疏忽按了x键旁边的y键，这时候我们将根据y键在钢琴上离x键的距离（误差gap）的大小对此次按键做出评价。

算法步骤：

第一步：

计算出A,B每个键强度之和As,Bs。 得出比例因子factor=As/Bs。

第二步：

将A的当前键（keyNumA）与B的当前键（keyNumB）比较键对比，求出键的在键盘上的距离gap= keyNumA- keyNumB。若gap的值差别在规定的范围类那么S1=S1+y（y视gap的大小取值0-1）。进入第三步。若超出给定范围S1=S1+0进入第四步。

第三步：

将B的当前键（keyStrongB）的声音强度乘以factor与A当前键（keyStrongB）比较。若diff=keyStrongB\*factor- keyStrongA的值差别在规定的范围类那么S1=S1+x。x视diff的大小取值0-1，重复第二步直到A和B中某一个比较完成。

第四步：

当前比较的键位不匹配，相异的键位采用就近匹配（假设A的第29个键位与B的第29个键位不匹配，若A的键总数多，那么将A的第30个键与B的第29个键位匹配，若仍然不匹配将A的第31个键与B的第29个键位匹配，但是向后推进的键数不超过AB键数之差的1/5。超过1/5时，则开始移动B，将A的第29键与B的第30个键匹配若不匹配则将A的第29个键与B的第31个匹配向后推进的键数不超过AB键数之差的1/5），若就近匹配就近匹配失败分数S 不变，B键位比较回到原始位置，A键位位置+1，若成功S1+0.5分，从成功位置依次向后匹配.重复第二步和第四步直到A和B中某一个比较完成。

得分公式：

S=S1/Bn\*100/2

算法优势：算法优势在提高了键位比较时用户按键错误的宽容度，使得即使用户误按旁边的键我们也将给予一定分数。同时也参考声音强度。其他方案可以加以利用此方案尽心比较。

算法劣势：当演奏者在演奏过程中不是误按时多按了某个键采用此种方案评价将会把多按键认为是误按键相比于方案二或者一会降低用户得分。

### 4.1.4 键位时间分段分析法KeyTimeDiv

算法思想：根据演奏的总时间将A与B分成时间比例相等（如各占总时间1/10）的 键位段 将对应各段进行KeyGap键位比较。

算法步骤：

第一步：

根据B演奏的总时间分段，将演奏总时间划分为等长段，一共划分为X段。

同时将A也划分为时间等长（但是不一定和B的划分等长，因为AB总时长可能不同）的X段。

第二步：

将A1与B1采用KeyGap分析得到s1

将A2与B2采用KeyGap分析得到s2

将A3与b3采用KeyGap分析得到s3

······

将Ax与bx采用KeyGap分析得到sx

得分公式：

总得分s=(s1+s2+s3.......+sx)/x

算法优势：弥补了KeyGap会把演奏者在演奏过程中把多按键认为是误按键，当进行时间分段后即使演奏者在演奏过程中某一段多按键了，但是由于有多个段其他段并不受影响，降了了误判对整体的影响。

算法劣势：当演奏者在演奏过程中节奏把握的非常差分段将会造成每个段都对不上得分非常低。

### 4.1.5 特征键位分析法FeatureKeyDiv

算法思想：根据A与B中识别特征的键，将A于B 分成相同个数的段将对应各段进行键位比较。特征键取法：AB中次数相等，而且次数介于15-30次

算法步骤：

第一步：

取特征键。首先分析AB的键位特征，假设根据分析得出在A中Q键50次，P键20次,N键2次，在B中Q键53次，P键20次,N键2次，那么将选取P键位特征键（键次数太少分段太少，不合适。特征键取法：AB中次数相等，而且次数介于15-30次，若无特征键方法失效）根据特征键分段进入第二步。

第二步：

将A1与B1采用KeySeq分析或KeySeqStrong分析得到s1

将A2与B2采用KeySeq分析或KeySeqStrong分析得到s2

将A3与b3采用KeySeq分析或KeySeqStrong分析得到s3

······

将Ax与bx采用KeySeq分析或KeySeqStrong分析得到sx

得分公式：

s=(s1+s2+s3.......+sx)/x

算法优势：弥补了KeyGap会把演奏者在演奏过程中把多按键认为是误按键，当进行时间分段后即使演奏者在演奏过程中某一段多按键了，但是由于有多个段其他段并不受影响，降了了误判对整体的影响。同时与时间分段互补，降低了音乐演奏过程中节奏对最终结果的影响。

算法劣势：当需要评测的音乐寻找特征键失败时，此方法将不再适用。

## 4.2 评价模型实现

### 4.2.1 评价模型Java类综述



图 4-1类的关联图

Mod类主要是对现实中一些数据对象的抽象，大部分可以直接存储到数据库。

Caloulate类是对输入的音乐进行初步处理来选择合适的评价算法来进行评价音乐。

Compare接口是对评价算法的高度抽象出的统一对外接口。

BasicCompare基类实现了Compare接口，具体化就近匹配算法和其他compare类通用的比较方法。

### 4.2.2 Mod类的设计

1、Key：Key是抽象的钢琴的键位特征，其中seq代表Key在Music中的序列。num代表该键在钢琴上的编号，用来区分不同键位，name代表该键所代表的

钢琴按键的所发出声音的名字，strong代表该按键发出声音的大小，但是其单位并不是分贝，其数值只是代表声音大小的一个衡量。HZ代表该声音的频率，钢琴上每个键发出的声音频率并不相同。借此可以标记不同的声音。Times因为硬件采样率的原因我们不能区分声音持续长度但是可以检测到的声音1S内出现的次数，每40MS采样一次借此可以算出声音持续时间。Duration声音持续时间。

2、Music：music是整个乐曲的抽象，每个乐曲都有自己的一些特征，Music就是用来表现这些特征的数据结构，其中也加入了一些成员方法，这些方法主要是方便评价乐曲演奏水平的好坏。Keylst 是一个容器，主要是存储key的序列，将乐曲按键按时间先后顺序放入keylst。keyNum表示该乐曲中有多少个键，只有get方法get方法调用keylst.size()返回键的总数。Duration表示乐曲总持续时间。根据第一个键和最后一个键输入的时间计算出整个乐曲持续时间。I表示两个乐曲比较时当时当前乐曲比较在哪一个位置，初始为-1，为-1主要是为了使第一个key可以和其他Key一样调用next方法，统一比较时的方法调用。为了方便比较Music有一个next的方法，其原理就是是I自增。I还有一个SET方法，用于乐曲比较时，当采用临近匹配法进行匹配失败时用于I的回溯。是I回到首次匹配失败的地方。Residue用来记录当前剩余多少键未比较，这个值有着重要作用。当遇到不匹配的键时我们不会两个同时往后移，我们更倾向于认为剩余键多的可能出现多按键的错误（或者说键位少的是漏掉了某个键，当然这并不矛盾）。我们将会将剩余键位多的I值加1。使得剩余的键位尽量对齐在采用临近匹配原则来进行匹配。

3、Result: result(结果)主要用来记录整个评价的过程产生的有价值的数据。

Stand记录当前输入的标准音乐的总键数，input记录当前需要评价的音乐的总输入键数，diff记录了当前两首音乐的键数差，我们把它的值定为input-stand。其结果可正可负。diff\_ratio是指当前键数差在标准乐曲中所占的比例，他的值为diff/stand。结果可正可负。由diff和diff\_ratio可以大致描述当前音乐的漏按键位或者是多按键位的情况。

standTime记录当前输入的标准音乐的持续的总时间，inputTime记录当前需要评价的音乐的总输入的总时间，Timediff记录了当前两首音乐的持续时间差值，我们把它的值定为inputTime-standTime。其结果可正可负。Time\_ratio是指当前输入值得时间差在标准乐曲中所占的比例，他的值为Timediff/standTime。结果可正可负。由Timediff和Time\_ratio可以大致描述当前音乐的节奏快慢，Time\_ratio为负则表示输入音乐节奏偏快，负的越多节奏越快。反之就是节奏偏慢了。

standStrong记录当前标准音乐的输入声音响度的值，是每个键的响度值之和。inputStrong记录的是输入音乐的声音的响度值，是每个键的响度值之和。StrongDiff是声音响度的差值。我们将它定义为inputStrong-standStrong，StrongDiff大于0时表示当前输入的声音比较洪亮，小于0时表示当前输入声音比较柔弱。

### 4.2.3核心评价类的设计

1、BasicCompare:这是所有评价类的算法基类，他实现了Compare接口的两个方法compareKey和evaluate。

compareKey是用来比较两个key的特征的，需要传入两个待比较的Key。在其中我们最主要参考的值为键在钢琴的序号，以此来区分可见是否相同。其次是比较key的声音响度，当然在基类中我们默认声音响度得的风都是满分，因为基类本身的评价是最基础的键位顺序评价方法，所以忽略掉声音强度这个因素。但是在其他评价算法中需要覆写该方法。该方法会返回一个float的数组，在基类中数组长度为3。第一个位置放的值为0或1。0表达的意思键位不匹配。第二个位置值表达的意思是键位得分，若匹配得1分，否则0分。第三个位置值表达的意思是声音响度得分，因为基类并不参考声音响度所以都是得1分。

evaluate是用来计算整首音乐的得分的。需要传入三个参数一个标准乐曲(Music)，一个待比较乐曲(Music)，一个结果（Result）。因为结果（Result）中一些有价值的数据在传入evaluate之前已经被计算出来先放入结果对象（Result）中，所以这里需要传入结果对象（Result）以便把所有结果综合写入Result对象中。

divide是直接调用evaluate。

2、CompareStrong：本类继承于BasicCompare类，相当于间接的实现的Compare接口，本类与BasicCompare类最大的区别在于覆写了BasicCompare 类的compareKey的方法。

compareKey方法的覆写：CompareStrong类的campareKey方法是用来比较两个key的特征的，需要传入两个待比较的Key。在其中我们最主要参考的值为键在钢琴的序号和声音的响度，以此来评价按键的得分。但是声音大小的比较要特别注意，我们在比较两个声音大小是否相同时，不能直接比较，因为演奏者演奏的整体音量都很低，我们不能判定成演奏者声音强度不够。所以我们在比较声音强度时首先计算出标准音乐声音平均强度（StandStrong）和输入音乐声音平均强度（InputStrong）计算出声音强度比系数（factor= InputStrong/ StandStrong）。将当前标准声音输入的强度乘以强度比系数factor得到的值在于当前输入键的声音强度比较。因为声音强度相同的可能非常低，所以我们制定了在比较强度时给的是一个区间，区间总共有5个，由小到大，落在每个区间所得分数也不同。落到小的区间表示声音强度差别小得分高。反之亦然。区间大小以及每个区间得分都被定义为常量（参见5.2.4其他）。方便以后调整。返回一个float数组第一个位置放的值为0或1。0表达的意思键位不匹配。第二个位置值表达的意思是键位得分，若匹配得1分，否则0分。第三个位置值表达的意思是声音响度得分，根据落在区间得不同的分（0-1）。  
 3、CompareGap：本类同CompareStrong类一样，继承于BasicCompare类。于CompareStrong相比CompareGap类在比较两个Key的时候不仅仅参考了key的键号（num）和声音强度（strong）而且考虑了另外一种因素按键误差（gap）。

compareKey方法的覆写：简单来说就演奏者在演奏过程中本该按A键，但是由于疏忽按了A键旁边的B键，这时候我们将根据B键在钢琴上离A键的距离（误差gap）的大小对此次按键做出评价。具体来说我们需要先计算两个键的误差（gap）

gap=inputNum-standNum

inputNum是标准音乐中当前正在进行比较的键的键号，standNum是需要评价输入音乐中当前正在进行比较的键的键号。因为我们在对键位进行编号时时从左到右依次编号所以键号差就可以代表按键的误差。在评价过程中我们将误差大小分成4个等级。等级误差被定义为常量（参见5.2.4其他），每个等级的等分也被定义为常量放置在一个文件中方便以后配置调整。

此外CompareGap类的声音强度的评价方案同CompareStrong类。

4. CompareTimeDivide:此类继承于CompareGap类，也就是说他其实也是comapare结构的间接实现者。于CompareGap相比他额外覆写了divide的方法。CompareGap,BasicCompare,CompareStrong类都没有将音乐划分成小段，而是在方法类直接调用evaluate方法。方法的思想是将标准音乐（stand）和输入音乐(inut)按照时间的等比的分成N段。把没一下段作为输入传入evlauate进行比较再将每段平评价的结果进行汇总。这样做的优点在于因为是按时间等比划分可以相对的比较出演奏者整首音乐的节奏，若是节奏不对按时间划分后比较KEY将会出现很多错误。恰好节奏也是对音乐评价的一部分，这样做符合我们的预期。

divide方法的覆写：首先我们计算出输入和标准音乐的各自的总时长，然后用各自总时长除以类的成员变量（segments）得到每段的时长（这里得到的时长有两个，因为输入音乐和标准音乐时长不同），segments的是需要划分成多少段，在生成CompareTimeDivide，需要在构造反方法中给出。然后输入音乐和标准音乐根据各自每段时长都划分成小段。将没一小段传入evlauate进行比较，再将结果汇总。

另外两个方法继承于CompareGap类，没有做改变。

5、CompareFeatureKey:此类也继承于CompareGap类，也就是说他其实也是comapare结构的间接实现者。于CompareGap相比他额外覆写了divide的方法。CompareGap,BasicCompare,CompareStrong类都没有将音乐划分成小段，而是在方法类直接调用evaluate方法。方法的思想是将标准音乐（stand）和输入音乐(inut)按照某个特征键（featureKey）将原本的音乐划分成若干个小段落。此时值得注意的是stand和input的段数应该是相同的。如不同则需要换另外一个特征键来分段。把每一小段作为输入传入evlauate进行比较再将每段平评价的结果进行汇总。这样做当演奏者在演奏过程中每一段可能表现不同，若是在某一段演奏的特别差按照其他评价方法将会得到很差的结果，但是如果使用此方法将会弥补整首局部缺陷。

divide方法的覆写：首先我们把stand和input进行分段处理，由于类的内部

有一个成员变量featureKey这个是用来标记stand和input将如何进行分段的成员变量，遍历stand和input每当遇到一个featureKey就将这一部分划分成一个小段。然后每一小段都调用evlauate进行比较，再将结果汇总。

另外两个方法继承于CompareGap类，没有做改变。

### 4.2.4其他类

1、Calculate：评价类的调用者，主要负责计算出输入音乐和标准音乐的初步比较，例如计算出输入音乐和标准音乐的键数，键数差，声音强度，强度的比值，挑选出特征值。初步根据这些特征挑选出输入音乐和标准音乐应该用那种方法进行比较。在选择需要传入参数的Compare时参数的设置也是有Calculate计算出或者指派的。同时在Compare中生成Result的对象用来记录评价结果。在Calculate完成计算或评价时。Result将会返回给调用者。评价过程完成。

2、Convert: 在软件的设计过程中由于声音采集模块和评价的模块并不是一个人完成的。所以MOD类的成员不完全一致Convert的设计主要是完成声音采集模块输出的结果转换成评价的模块需要的数据模型。他完成的工作是评价方案正确进行的基础。

3、DATA类：这是一个常量类其中定义了

//比较失败后的尝试偏移的长度

**public** **static** **final** **int** ***OFFSET\_LENGTH*** =2;

//声音差别的等级 即所得分数

**public** **static** **final** **int** ***DIFF\_STRONG\_LEVEL1***=1;

**public** **static** **final** **float** ***DIFF\_STRONG\_SCORE1***=1;

**public** **static** **final** **int** ***DIFF\_STRONG\_LEVEL2***=4;

**public** **static** **final** **float** ***DIFF\_STRONG\_SCORE2***=0.6f;

**public** **static** **final** **int** ***DIFF\_STRONG\_LEVEL3***=8;

**public** **static** **final** **float** ***DIFF\_STRONG\_SCORE3***=0.3f;

**public** **static** **final** **int** ***DIFF\_STRONG\_LEVEL4***=16;

**public** **static** **final** **float** ***DIFF\_STRONG\_SCORE4***=0.1f;

//键位差别的等级

**public** **static** **final** **int** ***DIFF\_GAP\_LEVEL1***=0;

**public** **static** **final** **float** ***DIFF\_GAP\_SCORE1***=1;

**public** **static** **final** **int** ***DIFF\_GAP\_LEVEL2***=1;

**public** **static** **final** **float** ***DIFF\_GAP\_SCORE2***=0.5f;

**public** **static** **final** **int** ***DIFF\_GAP\_LEVEL3***=2;

**public** **static** **final** **float** ***DIFF\_GAP\_SCORE3***=0.1f;

//段数的默认值

**public** **static** **final** **int** ***DEFAULT\_SEG***=10;

这些参数是用来动态配置整个评价模型，通过这配置这些参数将很容易对评价的结果进行调整。方便我们矫正评价模型的精确度。

# 5 结果分析

## 5.1测试数据的准备

音乐经过声音采集模块得到的结果：

标准乐曲：

04-29 18:35:46.460: V/KeyCount(15675): 7次：音名：#F(bG)\_5, 键号：58, 频率：739.989Hz,误差：未计算(精度：0.8） 强度：20.708987024212735

04-29 18:35:46.540: V/KeyCount(15675): 11次：音名：E\_5, 键号：56, 频率：659.255Hz,误差：未计算(精度：0.8） 强度：27.702004901212455

04-29 18:35:47.240: V/KeyCount(15675): 33次：音名：G\_5, 键号：59, 频率：783.991Hz,误差：未计算(精度：0.8） 强度：28.5792987420318

04-29 18:35:47.280: V/KeyCount(15675): 7次：音名：E\_5, 键号：56, 频率：659.255Hz,误差：未计算(精度：0.8） 强度：22.12825355899109

······（限于篇幅省略）

04-29 18:39:10.510: V/KeyCount(15675): 40次：音名：A\_5, 键号：61, 频率：880.0Hz,误差：未计算(精度：0.8） 强度：23.505036920406173

04-29 18:39:10.560: V/KeyCount(15675): 6次：音名：B\_5, 键号：63, 频率：987.767Hz,误差：未计算(精度：0.8） 强度：11.840122555690083

04-29 18:39:10.600: V/KeyCount(15675): 13次：音名：D\_5, 键号：54, 频率：587.33Hz,误差：未计算(精度：0.8） 强度：19.716655781171003

需评测乐曲：

04-29 18:35:46.460: V/KeyCount(15675): 7次：音名：#F(bG)\_5, 键号：58, 频率：739.989Hz,误差：未计算(精度：0.8） 强度：20.708987024212735

04-29 18:35:46.540: V/KeyCount(15675): 11次：音名：E\_5, 键号：56, 频率：659.255Hz,误差：未计算(精度：0.8） 强度：27.702004901212455

04-29 18:35:47.240: V/KeyCount(15675): 33次：音名：G\_5, 键号：59, 频率：783.991Hz,误差：未计算(精度：0.8） 强度：28.5792987420318

04-29 18:35:47.280: V/KeyCount(15675): 7次：音名：E\_5, 键号：56, 频率：659.255Hz,误差：未计算(精度：0.8） 强度：22.12825355899109

······（限于篇幅省略）

04-29 18:39:10.510: V/KeyCount(15675): 40次：音名：A\_5, 键号：61, 频率：880.0Hz,误差：未计算(精度：0.8） 强度：23.505036920406173

04-29 18:39:10.560: V/KeyCount(15675): 6次：音名：B\_5, 键号：63, 频率：987.767Hz,误差：未计算(精度：0.8） 强度：11.840122555690083

04-29 18:39:10.600: V/KeyCount(15675): 13次：音名：D\_5, 键号：54, 频率：587.33Hz,误差：未计算(精度：0.8） 强度：19.716655781171003

## 5.2 测试结果

1、采用键顺序分析法KeySeq分析

input的键号序列: 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 56 58 56 59 53 61 53 58 54 59 56 63 70 64 56 59 56 68 61 54 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 63 56 58 59 56 61 53 59 54 63 70 64 58 59 56 68 54 61 54 20 17 18 58 58 59 53 61 55 59 54 55 63 70 57 59 56 54 61 63 54 58 58 59 53 61 55 59 54 55 63 70 57 59 56 54 61 63 54

stand的键号序列: 58 56 59 56 61 53 58 59 54 63 70 64 58 56 61 54 56 58 56 59 53 89 53 58 54 59 56 63 70 64 56 59 56 68 61 54 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 72 56 58 59 56 61 53 59 54 63 70 64 58 59 56 68 54 61 54 20 17 18 58 58 59 53 61 55 54 54 55 63 70 57 59 56 54 61 63 54 58 58 59 54 55 63 70 57 59 56 54 61 63 54

stand的键号：58 比较 input的键号：58

stand的键号：56 比较 input的键号：56

stand的键号：59 比较 input的键号：59

stand的键号：56 比较 input的键号：56

stand的键号：54 比较 input的键号：54

······

stand的键号：63 比较 input的键号：63

stand的键号：54 比较 input的键号：70

stand的键号：54 比较 input的键号：57

stand的键号：54 比较 input的键号：59

stand末尾剩余键数：--------------------------->0

input末尾剩余键数：--------------------------->8

标准乐曲的总按键个数----------------------------->110

输入乐曲的总按键个数----------------------------->106

输入乐曲与标准乐曲的总按键个数差值---------------->-4

按键差值所占的比例------------------------------->-0.036363

标准音乐时长------------------------------------->204s

输入音乐时长------------------------------------->204s

时间差值：标准-输入------------------------------>0

时间差值所占的比例------------------------------->0.0

标准音乐按键键平均声音强度------------------------>20.223

输入音乐按键键平均声音强度------------------------>20.44057

强度差值----------------------------------------->0.21757126

综合得分----------------------------------------->91.818184

2、采用键顺序分析法KeySeq分析

input的键号序列: 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 56 58 56 59 53 61 53 58 54 59 56 63 70 64 56 59 56 68 61 54 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 63 56 58 59 56 61 53 59 54 63 70 64 58 59 56 68 54 61 54 20 17 18 58 58 59 53 61 55 59 54 55 63 70 57 59 56 54 61 63 54 58 58 59 53 61 55 59 54 55 63 70 57 59 56 54 61 63 54

stand的键号序列: 58 56 59 56 61 53 58 59 54 63 70 64 58 56 61 54 56 58 56 59 53 89 53 58 54 59 56 63 70 64 56 59 56 68 61 54 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 72 56 58 59 56 61 53 59 54 63 70 64 58 59 56 68 54 61 54 20 17 18 58 58 59 53 61 55 54 54 55 63 70 57 59 56 54 61 63 54 58 58 59 54 55 63 70 57 59 56 54 61 63 54

stand的键号：58 比较 input的键号：58

stand的声音强度：20.931787 比较 input的声音强度：：20.708986

stand的键号：56 比较 input的键号：56

stand的声音强度：28.000042 比较 input的声音强度：：27.702005

stand的键号：59 比较 input的键号：59

stand的声音强度：28.886772 比较 input的声音强度：：28.579298

stand的键号：56 比较 input的键号：56

stand的声音强度：22.366323 比较 input的声音强度：：22.128254

stand的键号：61 比较 input的键号：61

······

stand的键号：54 比较 input的键号：54

stand的声音强度：27.372774 比较 input的声音强度：：28.475283

stand的键号：61 比较 input的键号：55

stand的键号：61 比较 input的键号：63

stand的键号：61 比较 input的键号：70

stand的键号：63 比较 input的键号：55

stand的键号：63 比较 input的键号：63

stand的声音强度：11.967505 比较 input的声音强度：：19.249468

stand的键号：54 比较 input的键号：70

stand的键号：54 比较 input的键号：57

stand的键号：54 比较 input的键号：59

stand末尾剩余键数：----------------->0

input末尾剩余键数：----------------->8

标准乐曲的总按键个数----------------------------->110

输入乐曲的总按键个数----------------------------->106

输入乐曲与标准乐曲的总按键个数差值---------------->-4

按键差值所占的比例------------------------------->0.0

标准音乐时长------------------------------------->204s

输入音乐时长------------------------------------->204s

时间差值：标准-输入------------------------------>0

时间差值所占的比例------------------------------->0.0

标准音乐按键键平均声音强度------------------------>20.223

输入音乐按键键平均声音强度------------------------>20.44057

强度差值----------------------------------------->0.21757126

综合得分----------------------------------------->89.409096

3、键位误差分析法KeyGap分析

input的键号序列: 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 56 58 56 59 53 61 53 58 54 59 56 63 70 64 56 59 56 68 61 54 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 63 56 58 59 56 61 53 59 54 63 70 64 58 59 56 68 54 61 54 20 17 18 58 58 59 53 61 55 59 54 55 63 70 57 59 56 54 61 63 54 58 58 59 53 61 55 59 54 55 63 70 57 59 56 54 61 63 54

stand的键号序列: 58 56 59 56 61 53 58 59 54 63 70 64 58 56 61 54 56 58 56 59 53 89 53 58 54 59 56 63 70 64 56 59 56 68 61 54 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 72 56 58 59 56 61 53 59 54 63 70 64 58 59 56 68 54 61 54 20 17 18 58 58 59 53 61 55 54 54 55 63 70 57 59 56 54 61 63 54 58 58 59 54 55 63 70 57 59 56 54 61 63 54

stand的键号：58 比较 input的键号：58

stand的声音强度：20.931787 比较 input的声音强度：：20.708986

stand的键号：56 比较 input的键号：56

stand的声音强度：28.000042 比较 input的声音强度：：27.702005

stand的键号：59 比较 input的键号：59

stand的声音强度：28.886772 比较 input的声音强度：：28.579298

······

stand的键号：61 比较 input的键号：61

stand的声音强度：23.75792 比较 input的声音强度：：23.505037

stand的键号：63 比较 input的键号：63

stand的声音强度：11.967505 比较 input的声音强度：：11.840122

stand的键号：54 比较 input的键号：54

stand的声音强度：19.92878 比较 input的声音强度：：19.716656

stand末尾剩余键数：----------------->0

input末尾剩余键数：----------------->0

标准乐曲的总按键个数----------------------------->110

输入乐曲的总按键个数----------------------------->106

输入乐曲与标准乐曲的总按键个数差值---------------->-4

按键差值所占的比例------------------------------->0.0

标准音乐时长------------------------------------->204s

输入音乐时长------------------------------------->204s

时间差值：标准-输入------------------------------>0

时间差值所占的比例------------------------------->0.0

标准音乐按键键平均声音强度------------------------>20.223

输入音乐按键键平均声音强度------------------------>20.44057

强度差值----------------------------------------->0.21757126

综合得分----------------------------------------->83.54548

4、键位时间分段分析法KeyTimeDiv 分析

input的键号序列: 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 56 58 56 59 53 61 53 58 54 59 56 63 70 64 56 59 56 68 61 54 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 63 56 58 59 56 61 53 59 54 63 70 64 58 59 56 68 54 61 54 20 17 18 58 58 59 53 61 55 59 54 55 63 70 57 59 56 54 61 63 54 58 58 59 53 61 55 59 54 55 63 70 57 59 56 54 61 63 54

stand的键号序列: 58 56 59 56 61 53 58 59 54 63 70 64 58 56 61 54 56 58 56 59 53 89 53 58 54 59 56 63 70 64 56 59 56 68 61 54 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 72 56 58 59 56 61 53 59 54 63 70 64 58 59 56 68 54 61 54 20 17 18 58 58 59 53 61 55 54 54 55 63 70 57 59 56 54 61 63 54 58 58 59 54 55 63 70 57 59 56 54 61 63 54

stand的键号：58 比较 input的键号：58

stand的声音强度：20.931787 比较 input的声音强度：：20.708986

stand的键号：56 比较 input的键号：56

stand的声音强度：28.000042 比较 input的声音强度：：27.702005

stand的键号：59 比较 input的键号：59

stand的声音强度：28.886772 比较 input的声音强度：：28.579298

••••••

stand的键号：61 比较 input的键号：61

stand的声音强度：23.75792 比较 input的声音强度：：23.505037

stand的键号：63 比较 input的键号：63

stand的声音强度：11.967505 比较 input的声音强度：：11.840122

stand的键号：54 比较 input的键号：54

stand的声音强度：19.92878 比较 input的声音强度：：19.716656

stand末尾剩余键数：----------------->0

input末尾剩余键数：----------------->0

标准乐曲的总按键个数----------------------------->110

输入乐曲的总按键个数----------------------------->106

输入乐曲与标准乐曲的总按键个数差值---------------->-4

按键差值所占的比例------------------------------->0.0

标准音乐时长------------------------------------->204s

输入音乐时长------------------------------------->204s

时间差值：标准-输入------------------------------>0

时间差值所占的比例------------------------------->0.0

标准音乐按键键平均声音强度------------------------>20.223

输入音乐按键键平均声音强度------------------------>20.44057

强度差值----------------------------------------->0.21757126

综合得分----------------------------------------->83.54548

5、特征键位分析法FeatureKeyDiv

input的键号序列: 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 56 58 56 59 53 61 53 58 54 59 56 63 70 64 56 59 56 68 61 54 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 63 56 58 59 56 61 53 59 54 63 70 64 58 59 56 68 54 61 54 20 17 18 58 58 59 53 61 55 59 54 55 63 70 57 59 56 54 61 63 54 58 58 59 53 61 55 59 54 55 63 70 57 59 56 54 61 63 54

stand的键号序列: 58 56 59 56 61 53 58 59 54 63 70 64 58 56 61 54 56 58 56 59 53 89 53 58 54 59 56 63 70 64 56 59 56 68 61 54 58 56 59 56 61 53 58 59 54 63 70 64 59 56 61 54 72 56 58 59 56 61 53 59 54 63 70 64 58 59 56 68 54 61 54 20 17 18 58 58 59 53 61 55 54 54 55 63 70 57 59 56 54 61 63 54 58 58 59 54 55 63 70 57 59 56 54 61 63 54

stand的键号：58 比较 input的键号：58

stand的声音强度：20.931787 比较 input的声音强度：：20.708986

stand的键号：56 比较 input的键号：56

stand的声音强度：28.000042 比较 input的声音强度：：27.702005

stand的键号：59 比较 input的键号：59

stand的声音强度：28.886772 比较 input的声音强度：：28.579298

······

stand的键号：54 比较 input的键号：54

stand的声音强度：27.372774 比较 input的声音强度：：27.081415

stand的键号：61 比较 input的键号：61

stand的声音强度：23.75792 比较 input的声音强度：：23.505037

stand的键号：63 比较 input的键号：63

stand的声音强度：11.967505 比较 input的声音强度：：11.840122

stand的键号：54 比较 input的键号：54

stand的声音强度：19.92878 比较 input的声音强度：：19.716656

stand末尾剩余键数：----------------->0

input末尾剩余键数：----------------->0

score2----------------------->179.40007

81.54549

标准乐曲的总按键个数----------------------------->110

输入乐曲的总按键个数----------------------------->106

输入乐曲与标准乐曲的总按键个数差值---------------->-4

按键差值所占的比例------------------------------->0.0

标准音乐时长------------------------------------->204s

输入音乐时长------------------------------------->204s

时间差值：标准-输入------------------------------>0

时间差值所占的比例------------------------------->0.0

标准音乐按键键平均声音强度------------------------>20.223

输入音乐按键键平均声音强度------------------------>20.44057

强度差值----------------------------------------->0.21757126

综合得分----------------------------------------->81.54549

# 6 总结和展望

## 6.1 项目过程中遇到的问题及解决方案

在我们开发人员当中并没有人掌握专业的钢琴演奏技巧，以及钢琴评价的准则。简单的说就是我们对钢琴义务所知，这导致我们被项目开始的第一步所困扰无从下手，好在这个问题很快被我们解决了，老师组织我们学习基本的乐理。这让我们一群工科生对音乐有了初步的了解，根据所了解的知识初步拟定了评价的方案。

后来在实验设备上有遇到困难我们没有一架可以用来试验的钢琴，这架钢琴很重要，它是声音的来源。没有钢琴我们就没法做出识别钢琴按键的识别模块，在这个问题上得益于学校音乐学院的支持，同意我们开发人员进入音乐学院琴房进行编程测试。

评价方案的确定是个复杂的过程，项目组成员及指导老师不断地开会研讨最终将整个评价过程化繁为简。确定了清晰明确的评价方案。组织各个评价指标按照音乐评价的经验制定评价方法，并加以实现。

评价参数的确定。评价方法确定后，需要设置一些参数，这些参数表示评价模型的容错度。这个容错度与评价结果息息相关。容错度的确定需要有专业老师的参于。我们将软件计算出来结果与专业老师的结果相比较，根据比较的结果调整容错度，直到评价模型得出的结果接近月专业老师给出的结果。

## 6.2 前景展望

就音乐的计算机自动识别是新兴的交叉学科，其研究涉及物理学、信号处理、人机交互、音乐理论和音乐心理学等诸多学科知识。而其最主要的任务则是通过对音频信号的处理和特征提取，获取音乐内容的相关信息，进而用于比较、分类乃至自动录谱等。乐曲演奏水平分析软件的技术层面正是通过将计算机多媒体技术、信号处理与模式识别的相关知识和技术同音乐理论相结合，用计算机模拟人对音乐的认知和分析过程，分析解析音乐演奏并评价演奏的优劣，实现了对钢琴音乐演奏的特征识别与正误判定。钢琴演奏音乐的机器识别，在一定程度上可以辅助钢琴演奏的教学。

最终，本产品将实现APP程序录制音频文件（支持wav、mp4格式等）、对用户操作进行应答，并具备上传乐谱信息到服务器端进行比对，实现APP为乐曲演奏水平评估并给出最直观的结果的功能，并存储相应信息的数据文件，同时使用大数据平台云技术，根据用户提供的信息和历史需求进行“私人订制”，自主地向用户推送合适的商务项目。

本产品将大大降低学习钢琴的成本。市场前景非常广阔。

致谢

首先，要感谢我的导师向华老师，本论文从开题到定稿，向老师都给予我莫大的支持与帮助，他以博大的胸怀与宽广的知识面对论文提出了至关重要的改进建议，再有大学以来，我就在向老师的实验室中学习，跟着向老师也做过不少的项目，使我的能力得到了锻炼和提升；在升学和找工作的徘徊迷茫期，向老师为我拨开云雾，给与我帮助，向老师在学习生活的各个方面影响着我，由此我树立了正确的人生观和价值观。

其次，要感谢我的班主任科老师，科老师在大学四年为我们劳心劳力，我们13 班的同学都会记得您，还有长江大学计算机系的全体老师，没有你们的认真栽培，没有你们的付出，都不会有我们的今天。

再有，不得不感谢跟我一起做项目王琮吕超 等同学，项目中王琮同学负责呻吟采集模块的设计，吕超同学负责界面设计与实现，没有我们的齐心协力，就没有这套乐曲评价系统，没有你们的帮助，我也不能完成本论文的撰写。还有汉科528号寝室的兄弟们，短短三年的相处，我们从陌生到熟悉，从室友到朋友，我们虽没有血缘，但我们甚是兄弟，未来的路不长不短，希望我们还能一起走。

最后，我要感谢我的家人，感谢你们养育我二十多年，感谢你们的关爱与理解，感谢你们一路陪我，支持我。小时候感谢你们对我的保护和关怀，长大了，感谢你们对我的“放开”，并支持我做我想做的事，有你们这么慈爱开明的父母我何其荣幸，在这里谢谢你们！

感谢答辩组的老师和工作人员，你们严格的审核是我们高质量论文