# 摘要

随着音乐行业和互联网技术的飞速发展，可获取的音乐数据库规模在迅速膨胀，而音乐信息快速检索和音乐版权保护的重要性在近期不断地增长。因此，音乐版本鉴别成为了一个热门的研究话题。其目的是在现有乐库中检索出所提供待查询曲目的翻唱版本，不仅能帮助我们优化音乐推荐方式，更能促进音乐版权保护机制的完善。

虽然学术界已有了一些对于音乐版本鉴别的研究成果，但是却鲜见对于这些成果的工业应用。本文通过使用web技术结合卷积神经网络模型实现了一个音乐版本鉴别原型系统，可以让用户在网页轻松地进行各种音乐版本鉴别操作，旨在让那些对于计算机技术完全没有认知的用户方便的使用现今最前沿的音乐版本鉴别技术，并参与音乐数据库的构建。

同时，本文也进行了对于现有网络的改进尝试。现有的网络模型中大部分以t­r­i­p­let三元组损失或交叉熵损失作为损失函数。前者基于距离度量的训练思想，而后者则是典型的分类损失，二者皆有其可取之处，因此本文尝试使用结合tri­p­l­e­t和softmax的损失函数进行多损失训练。并且本文还借鉴了一些同样使用卷积神经网络的人脸识别领域的前沿损失函数进行训练，这些损失函数或是改变了现有的距离度量模式，增大了分类时的类间距离，又或是很好的将度量学习模式和分类学习模式结合到了一起，本文总结了这些损失函数的实验结果并加以分析，为音乐版本鉴别的模型改进提供了一条思路。

**关键词：**音乐版本鉴别，web技术，深度学习，损失函数

# Abstract

5号，Time new roman，1.5倍行距

**Key Words：**3-5个

# 全文目录

[摘要 9](#_Toc476314826)

[Abstract 10](#_Toc476314827)

[第一章 绪论（二号、黑体） 12](#_Toc476314828)

[1. 第一级（三号、宋体/Time new roman、加粗） 12](#_Toc476314829)

[1.1 第二级（四号，宋体/Time new roman，加粗） 12](#_Toc476314830)

[1.1.1 第三级（小四，宋体/Time new roman，加粗） 12](#_Toc476314831)

[参考文献 14](#_Toc476314832)

[本科期间的主要工作和成果 15](#_Toc476314833)

[致谢 15](#_Toc476314834)

# 第一章 引言

1.1 工作背景

音乐作为历史最为悠久的艺术形式之一，一直深刻地影响着人类社会的方方面面。它不仅可以带给人听觉的享受，也可以有着独特的象征意义，它的多样性与自由性让它成为了人们生命中无法缺失的一部分。

音乐的背后有着一套十分完善的理论体系，让编曲者们在创作时可以通过使用不同的旋律、节奏、音色等要素来表达自己的创作意图，而这种创作意图却可以让大部分听众在没有任何音乐理论基础的情况下捕捉到。然而，和很多其他艺术形式一样，音乐也有着一些理论之外的神秘性，实在有太多的因素可以影响一个音乐的内核。也就是说，音乐在有一定的可解释性的基础上，又有着传统科学理论难以说明的一部分，因此当我们试着使用传统方法将音乐数据量化并设法机械地揭示音乐的本质的时候，总是会遇到一些困难。但最近十分热门的人工智能技术似乎恰好符合音乐研究的需求，深度学习可以让计算机学习出音乐的特征表示，虽然我们无法去解释特征中每个维度的信息是什么，但我们可以应用它进行许多的音乐研究。本文就是应用神经网络所提取的特征来探究如何让计算机学习如何去判断音乐间的相似性，并利用这种相似性去进行音乐翻唱版本的检索以及辅助音乐侵权的界定。

同时，大多数人们对于人工智能并没有很深入的了解，如今还没有一个成熟的可展示的翻唱版本鉴别系统可供人们使用，如何将这一尖端技术呈现给人们，并给音乐研究者的工作带来便利，成为了一个亟待解决的问题。

1.2 工作意义

音乐信息检索（Music Information Retrieval）是计算机音乐信号处理领域的一个热点问题，而翻唱版本鉴别（Cover Song Identification），或音乐版本鉴别，又是音乐信息检索领域中最为重要的分支之一。它被定义为：给出一个音乐查询，系统在乐库中筛选出最可能是其翻唱版本的音乐版本集合。音乐版本是指对于一个音乐作品的演绎，比如对于一个音乐作品的不同歌手翻唱版本、音乐作品的录音棚版本与现场版本、人声音乐的纯音乐改编、古典音乐的现代演绎等等，都属于一个音乐作品的不同演绎，也即不同版本。

翻唱版本鉴别的研究价值如下几个方面：

音乐的个性化推荐。音乐的个性化推荐可以通过计算音乐间的相似性来实现。通过使用训练好的用于翻唱版本鉴别的神经网络提取出音乐的特征，计算用户所听音乐的最相近版本，辅助进行基于内容的推荐。

音乐版本检索。用户想要在海量的数据库中快速地查询出所给音乐片段的相似版本。

音乐版权维护。随着音乐市场的不断扩大，近年来音乐侵权问题成为了一个焦点话题。我们可以通过计算音乐特征间的相似度，并将其作为是否侵权的重要参考依据。

1.3 存在的问题

不同音乐版本之间有着错综复杂的联系与区别：大多数的音乐版本都会在很大程度上地保留最原始版本的旋律信息，但却会在节奏、音调、音色、编曲上产生很大的区别。比如男歌手翻唱女歌手的作品时会降调，现场版会比录音棚版本拥有更多的节奏变化和即兴要素，古典乐曲的现代版本会进行大幅度的乐器变更，这些都是不同的音乐版本可能存在的区别，而正是这些区别很大程度上的给翻唱版本鉴别这一任务造成了阻碍。

最初，研究者们使用手工设计的特征结构来作为音乐数据的抽象，并使用时间动态规划算法来计算不同乐曲间的最优匹配，从而减少音乐版本中乐曲结构和节奏的变换所产生的影响，取得了一定的成果。其缺点在于算法的复杂度过高，无法在如今庞大的音乐数据库中应用。其次，这种方法往往依赖于对音乐理论知识和信号处理知识的高度理解，并用这些知识去精心地设计对齐算法和特征结构，导致研究门槛高、研究开销大。

为了解决上面所述的难点，近年来有更多的研究者开始尝试使用深度学习的方法来解决这一课题。由于深度学习的特性，使用神经网络提取出的特征序列的效果往往比传统的手工方法更佳。但研究者的研究焦点一般在于如何设计神经网络的结构，很少有人去关注损失函数的使用。研究者通常使用距离度量的学习模式中的常见损失函数（metric learning），如三元组损失，或是使用softmax的交叉熵损失来进行学习。然而当我们跳出音乐信息检索这一狭窄的范畴，放眼整个人工智能领域，就会发现在更为火热的人脸识别领域，如何设计一个性能更为优秀的损失函数已经成为了一个十分热门的话题，并且已经有了很多成果。目前大部分针对翻唱版本鉴别问题的深度学习方法，其本质都是使用卷积神经网络对音乐的频谱信息进行处理，而这事实上和人脸识别的任务十分类似，因此将这些最新的损失函数应用到翻唱版本鉴别任务中显然就成为了一条十分值得尝试的道路。

同时，数据的规模也会限制使用深度学习来解决这一问题的效果，现存的最大翻唱版本数据集*Second Hand Songs(SHS)*所包含的音频量仅有两万个，而这远远无法满足我们的需求。我们实验室构建了更加全面、庞大的数据集来提升神经网络训练的效果。

另外，当我们训练出了一个效果良好的神经网络后，如何将其效果直观的展示给人们也是一个值得思考的问题。我们可以结合web技术，用前端构建一个简洁、美观的页面来拉近人们与我们研究成果的距离，并让我们的研究成果给各行各业的人们带来便利。

1.4 本文主要工作

针对上一小节中所阐述的问题，本文分为两部分。

第一部分为研究性工作，主要尝试通过改进损失函数的方式来提升目前在翻唱版本鉴别领域中效果优越的神经网络的性能。本文所选择的网络结构为Jiang[1]所提出的“使用多级别深度序列距离矩阵的孪生神经网络”，原文中Jiang使用网络输出的Binary Cross Entropy(BCE)损失进行训练得到了不错的效果，而本文尝试使用BCE损失结合triplet损失以及预测结果的Softmax Cross Entropy(Softmax CE)损失在在实验室自建的*SHS100K*数据集上进行多损失训练，得到了不错的效果。在此之上，本文注意到了翻唱版本鉴别问题与人脸识别问题的相似性，在人脸识别领域选择了Cosface[2]、Sphereface[4]、Large Ma­r­gin Softmax(L-Softmax)[5]这三种损失函数进行了训练尝试。本文总结了实验的结果，并根据实验结果数据以及各个损失函数的特性来分析这种损失函数跨领域迁移方法的效果及可行性。

第二部分为工程性工作，在改进了损失函数并进行筛选后，我们得到了一个性能最优的翻唱版本鉴别模型。本文使用Web技术并结合第一部分所得到的最优模型，以网页的形式构建了一个界面简洁、美观的翻唱版本鉴别原型系统。

该系统的功能特性如下：

第一，用户可以任意上传音频文件，网页计算并返回乐库中是其翻唱版本的可能性排名前十位的音频。

第二，用户可以播放网页所返回的这十个音频。

第三，用户可以任意上传两个音频文件，网页计算并返回这两个音频文件的相似度，用户可以将其作为二者是否为同一乐曲的不同版本的依据，并借此来判断是否侵权。

第四，用户可以向后台的乐库中添加音乐版本，参与音乐数据库的构建。

1.5 本文结构组织

第一章为引言，介绍了本文工作的背景，结合实际分析了本文工作的意义，总结了以往类似工作中存在的问题和难点，并介绍了本文的主要工作。

第二章为相关工作，介绍了以往工作者在解决音乐版本鉴别问题时所应用的特征结构以及相似性度量方式。

第三章为本文训练时所使用的音乐数据集和实验结果评价标准的介绍。

第四章为网络的损失函数改进尝试。

第五章介绍了音乐翻唱版本鉴别原型系统网页的构建。

第六章为全文内容的总结以及对未来研究工作的展望。

# 第二章 相关工作

2.1 特征结构

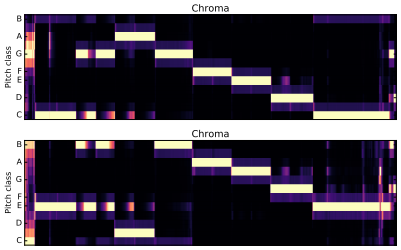
同一乐曲的不同翻唱版本的节奏、音调、编曲、音色等要素可能有着很大的区别。一些传统的特征，如梅尔频率倒谱系数MFCC（Mel-Frequency Cepstral Coefficients）特征，虽然能够得到音频的低层特征，但却无法表示旋律等高层特征，因此这些传统特征几乎不具有对旋律、节奏等因素的不变性，很难应用于音乐翻唱版本鉴别的任务之中。但不同翻唱版本往往有着相似的旋律，因此研究者们基于这一特性提出了很多有效音乐的特征表示方法。

**2.1.1 Timbral Shape Sequence**

**虽然大部分研究者都认为旋律是翻唱版本鉴别任务中最为重要的因素，因此MFCC这种传统的特征表示无法应用于此任务，但Tralie和Bendich却认为音色信息也同样关键。他们认为即使不同翻唱版本之间的音色可能有很大区别，它们的音色在时间尺度上的变化情况却应该是类似的，而MFCC恰好能够很好地表示音频的音色这一低层特征，因此他们构建了新的模型来模拟音色的这一变化， 将多个不同窗口内的MFCC特征进行嵌入，构造出了一种新的特征序列[6]。他们的方法在*Covers80*数据集上得到了52.5%的准确率，这虽然与当时最优算法的结果相差甚远，但也从某种程度上证明了传统特征表示也是有其可行性的。**

**2.1.2 Chroma**

**Chroma可能是最为常见的基于音高的音乐特征之一，它描述的是音乐片段中十二个半音的强度。具体来说，是按照预先确定的帧长来切分时间窗口，然后进行傅里叶变换得到频域信息，之后再叠加十二个半音的能量叠加。在进行了这些步骤后，我们就可以在每个窗口中得到一个代表半音强度的十二维向量，并将这些向量的集合作为音乐片段的特征表示。Chroma最初被应用在和弦检测任务中[8]，之后研究者们为了让Chroma特征更能适应节奏、音调等因素的变化，又对其进行了许多的改进。如**Tzanetakis等人在2003年提出的**同样基于音高旋律信息的Pitch histogram特征[9]，这也是最早被用于音乐版本鉴别问题的音频特征。之后**Müller 等人提出了一种叫做Chroma Energy Normalized Statistics（CENS）的增强Chroma特征[10]，这种特征拥有更好的音调不变性。此后，**Ellis等人为了提高Chroma特征的节奏不变性，提出了Beat-Aligned Chroma[11]。他们对音乐中的鼓点进行追踪，在鼓点存在的片段进行Chroma特征提取，从而尽可能的消除节奏对于特征表示的影响。另一个十分常用的Chroma变种就是Harmonic Pitch Class Profile特征（HPCP）[12][13]。相比于Chroma特征，HPCP的提取过程完成了基音和泛音的分解，并对声谱图进行了峰值检测，从而减小了局部的干扰和噪声对于特征的影响。下图是不同版本小星星的前四个小节的HPCP可视化图片。**



**图2.1 不同版本小星星前四个小节的HPCP特征**

**2.1.3 Constant-Q Transform**

**Constant-Q Transform（CQT）的目的和傅里叶变换类似，都是产生音乐的频谱[14]。而他们的过程也是相似的，首先切分窗口，然后在窗口中使用滤波器进行变换。但我们知道，音乐中所有的音都是由若干八度的十二平均律组成，而这十二平均律则对应了一个八度上的是二个半音，每对相邻半音的频率比为21/12，因此可以说音乐中的声音是以指数分布的。然而通过傅里叶变换得到的频谱却是线性分布的，这导致我们无法将音高和频谱中的频率点一一对应，从而会使某些音阶频率的估计值产生误差。但CQT可以得到以对数标度作为横轴的频谱，从而将频率直接映射到音符之上，直接得到音乐信号在各音符频率处的振幅值。同时，CQT根据不同的谱线频率来改变滤波窗的长度，来获得更好的性能。因为这些优越性，CQT也是我们本文中所选择的音乐特征的初始表示。下图是Jimi Hendrix版本及Bob Dylan版本的*All Along the Watchtower* 的CQT频谱。**

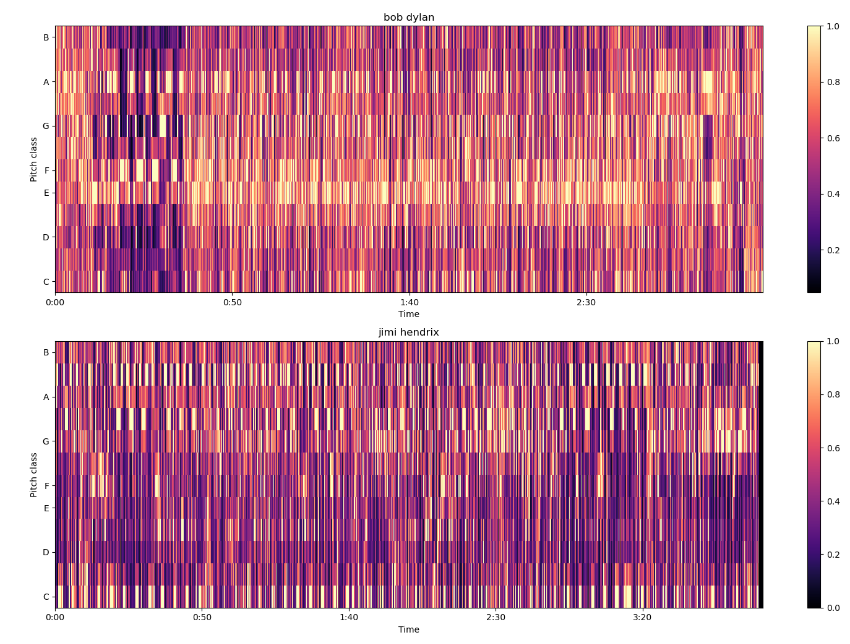


图2.2 两个版本*All Along the Watchtower*的CQT频谱图

2.2 相似性度量

翻唱版本鉴别任务中的另一个重要问题就是如何对得到的音乐特征之间进行相似度的度量，度量方式的好坏会十分直接地影响鉴别的速度以及准确率。

**2.2.1 直接计算定长向量距离**

当我们得到了特征向量后，一个最为简单、直接的想法就是直接计算这些向量之间的距离。而如果我们能够得到定长的特征向量，这些向量之间的距离计算就变得极为简单。常用的距离度量方式有余弦距离，欧氏距离（L2距离）[15]以及曼哈顿距离（L1）等。由于这种方式只是对向量进行复杂度极低的数学运算，因此其速度很快。显然，这种单纯的度量方式的效果十分依赖于所提取特征的优越性，但是因为手工构造的特征无法全面的表现音乐的全部特质，直接在这类特征向量上进行这些运算的效果并不理想。同时，由于我们需要在定长向量上进行计算，而将向量的长度固定就必定会丢失乐曲中的一些信息，这也同样会影响计算结果的精确度。

但事实上，如果我们使用深度学习的方法来提取一个能够尽可能的描述音乐所有信息的极高维度的特征，这种简单的方法反而会变得比较有效。

**2.2.2 动态时间规整**

为了消除局部结构和速度变化的影响，很多研究者使用动态时间规整（Dynamic Time Warping）算法来动态伸缩得到的音频特征，从而让特征向量间的最优匹配更加精确[16][17]。这种算法在深度学习方法出现前可以说是翻唱版本鉴别领域效果最为优秀的算法。但这种算法只能应对局部的变化，而难以处理编曲改变、现场即兴等因素所引起的音乐整体的结构变化。同时，随着互联网的发展，可用的音乐数据集的体量飞速膨胀，基于动态规划算法的DTW复杂度较高，无法在很多大型音乐数据集上应用，这极大降低了DTW的性能上限。

**2.2.3 基于神经网络的相似度计算**

随着深度学习领域的不断发展，越来越多的研究者开始将深度学习应用到翻唱版本鉴别任务中。一些研究者使用神经网络学习出定长特征作为音乐的表示，并使用2.2.1中所述的几种距离计算方式作为版本间相似度的度量。如Xu使用乐曲的Chroma序列作为卷积神经网络的输入，学习提取出音乐的定长特征向量，并使用余弦距离来计算相似度[18]。这样的方式能得到十分不错的效果，但由于使用的是Chroma这种音乐高层特征表示，会丢失一些音乐的底层信息，同时也存在前文所述的使用定长向量所带来的问题。而Jiang从人脸识别领域得到启发，使用距离度量学习模式，采取了孪生神经网络结构与卷积神经网络（CNN）进行联合训练，通过将多级深度序列的相似矩阵传入CNN来计算出相似度[1]。这种方法让序列相似度的计算不再仅依靠传统的手工特征，在多个数据集上得到了更好的效果。

# 第三章 所用数据集及评价指标

本文选择运用深度学习的方法来解决翻唱版本鉴别问题，使用Jiang[1]所提出的孪生神经网络进行训练。由于现在网络上公开的音乐版本数据集的规模普遍较小，很难满足我们训练的需求，本文选择使用本实验室所构建的*SHS100K*数据集进行训练，而在验证和测试的过程中再使用网络上流行的一些经典数据集。

而当我们在选择的数据集上得到了实验结果之后，如何去评价这些结果的优劣又成为了一个重要的问题。本文选择mAP、P@10和MR1来作为实验结果的评价指标。以下是对于这些数据集和评价指标的具体介绍。

3.1 所用数据集

**3.1.1 *Covers80***

*Covers80*是由Columbia大学的LabROSA实验室所发布的一个小规模数据集，在翻唱版本鉴别领域中十分流行[19]。它包含80首时长为2-6分钟的西方流行歌曲，歌词语种为英文，每个歌曲有两个版本。和一些只提供歌曲Chroma特征的数据集（如*SecondHandSongs Dataset*）不同的是，*Covers80*数据集直接给出了这些歌曲的音频，同时给出了歌曲名及音频之间的版本关系，这方便我们用各种不同的方式提取音频的特征进行分析。因为其规模较小，使用起来方便灵活，一般被应用于训练过程中的验证和测试阶段，几乎最近的所有工作都会将*Covers80*上的测试结果作为评价方法优劣性的基准之一，本文也不例外。

**3.1.2 *Mazurkas***

*Mazurkas*是一个古典音乐数据集[20]。顾名思义，它包括49首肖邦谱曲的马祖卡舞曲。与*Covers80*相比，*Mazurkas*中每个舞曲的版本数更多，在41到95之间，而总音频数为2914。此外，大部分的音乐版本数据集都是流行乐或是有人声的音乐，而*Mazurkas*是少有的古典音乐数据集，用其进行测试可以更全面地评估神经网络的效果。

**3.1.3 *Youtube350***

*Youtube350*是Silva等人收集构建的一个包含50首歌曲的音乐数据集[21]。这个数据集的优势在于其曲风多样，同时组织清晰。每首曲目有七个版本：一个录音室版本、一个现场版本、五个翻唱版本，故共计350个音频。我们使用这个数据集进行训练中的测试，用翻唱版本作为询问，原唱版本作为参考。

**3.1.4 *SHS100K***

以上三个数据集都是小规模数据集，只能用来验证和测试。而现存最大的数据集*Second Hand Songs*（SHS）也仅仅包含两万个音频，这也是之前很少有研究者将深度学习成功应用于翻唱版本鉴别问题的原因。为了解决这一问题，我们利用secondhandsongs.com中的翻唱标签信息和歌曲在youtube上的链接来爬取音频，构成一个一个全新的数据集。由于最后爬取的8858首歌曲中所有版本的总音频数量为108523个，因此将这个数据集命名为*Second Hand Songs 100K*（SHS100K）。

在本工作中，我们使用*SHS100K*进行监督学习。我们将*SHS100K*中的曲目按8:1:1的比例随机分为三个子集：*SHS100K-TRAIN*、*SHS100K-VAL*、*SHS100K-TEST*，分别作为训练集、验证集和测试集。

3.2 评价指标

**3.2.1 mAP**

Mean Average Precision（mAP），即平均精度均值，是预测目标类别或位置这类算法的常用性能度量标准。

想要理解mAP，首先要了解Average Precision（AP），即平均检索精度。若某个乐曲在数据库中有*n*个版本，而这*n*个版本在我们的检索结果中的排名分别为*r1, r2…rn*，则该乐曲检索的*AP*计算如下：

设我们共检索了T首歌曲，其中第*i*首歌曲的*AP*值为*AP(i)*，那么mAP的计算公式如下：

从公式中可以看出，歌曲版本在检索结果中的排名越靠前，得到的mAP值越高，说明检索的效果越好。

**3.2.2 P@10**

Precision at 10（P@10），即前十准确度。它反映的是在检索结果中前十名的结果是正确鉴别的比例。若我们共进行了*M*个歌曲查询，第*i*次询问的歌曲拥有*Ni*个版本，其中第*j*个版本在检索结果中的排名为*rij*，定义公式如下：

其中I为指示函数（括号中条件为真，其值为1，否则为0）。

由于在实际应用过程中，用户可能只关心排名靠前的检索结果中的内容，因此我们需要使用这一评价指标来评估我们的检索效率和应用效果。

**3.2.3 MR1**

Mean rank of first correctly identified cover（MR1），即第一个被正确鉴别的版本的平均排名。我们用这一指标评估我们检索的精确程度，若这个指标越接近1，就代表我们返回的与查询曲目最接近的结果越可能是待查询曲目的版本。

# 第四章 孪生神经网络损失函数改进尝试

大多数翻唱版本鉴别领域的研究者在尝试改进现有方法时，都着眼于网络结构，而忽视了损失函数的重要性。本文尝试引入三元组损失（Triplet Loss）和Softmax交叉熵损失（Cross Entropy Loss）来进行多损失训练，从而在让训练的效果更加全面。此外，在人工智能界更为热门的计算机视觉领域，如何创造一个训练效果优越的损失函数，以及如何给自己的网络结构和任务需求选择一个正确的损失函数已经成为了十分热门的研究焦点。近年来在人脸识别领域涌现了很多优秀的损失函数如ArcFace[3]、Circle Loss[7] 、Sphereface[4]、 Cosface[2]、Large Margin Softmax（L-Softmax）[5]等，本文选择了其中的Sphereface、 Cosface、L-Softmax进行训练尝试，具体思路和流程在后文阐述。

本文在众多使用深度学习解决翻唱版本鉴别问题的方法中，选择了Jiang[1]所提出的使用多深度序列相似矩阵的孪生神经网络进行损失函数改进尝试。这一结构使用了度量学习的思路，将计算相似度的卷积神经网络和提取序列特征的孪生神经网络融合在了一起，摆脱了传统的仅仅使用定长特征的相似度计算模式，在多个数据集上都得到了很好的效果。

4.1 神经网络结构介绍

下面介绍本文所使用的由Jiang提出的使用多深度序列相似矩阵的孪生神经网络。

整体的网络结构如图4.1所示，可以分为两部分[1]。

首先将我们所提取的两个乐曲的Constant-Q 时频图作为网络的输入传入共享权重的孪生神经网络中，经过四个卷积块可以分别得到四个不同深度级别的序列特征向量。对同级别的两个向量*Ak、Bk*计算它们的欧氏距离得到相似度矩阵*Ck*，四个相似度矩阵就是我们第一部分孪生神经网络的输出。

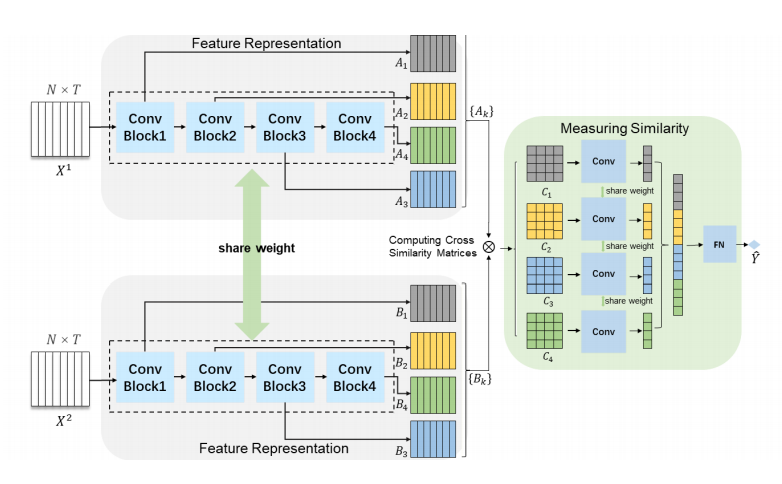


图4.1 总体网络结构图（省略了一些细节）

从第一部分的网络得到了相似度矩阵之后，我们分别将四个矩阵传入第二部分的共享权重的卷积神经网络中，得到四个输出向量。最后将四个向量传入全连接层，计算得到最后的相似度。

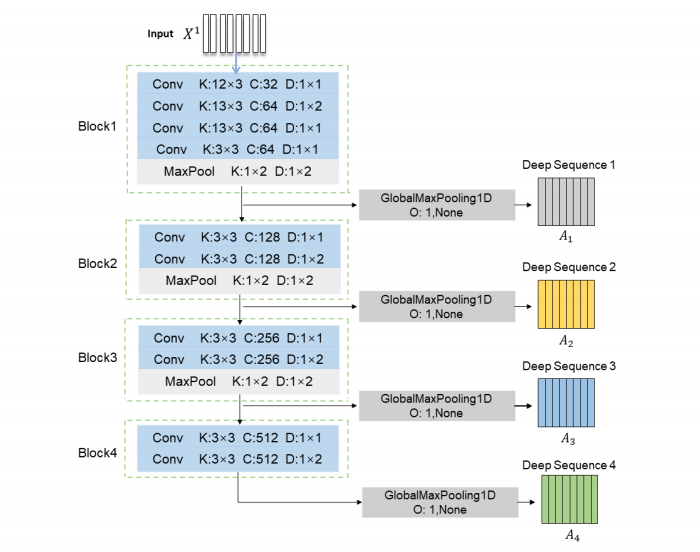


图4.2 第一部分的详细网络结构

图4.2展示了第一部分孪生神经网络的详细结构，其中K代表kernel size，即卷积核的维度；C代表channel，即通道数；D代表dilation，即膨胀系数；O代表output size，即输出维度。

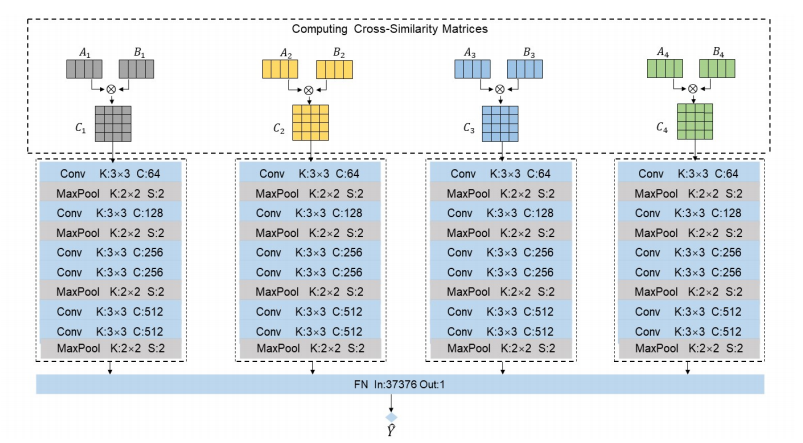


图4.3 第二部分的详细网络结构

图4.3展示了第二部分网络的详细结构，在这里我们通过计算两个输入产生的同深度的维度为*Nk \* Tk*的序列*Ak*{, , …, }*、Bk*{, , …, }的欧氏距离来得到维度为*Tk* \* *Tk*的相似矩阵*Ck*，计算公式如下：

将得到的每个相似矩阵传入同样的卷积网络块得到四个输出向量，最后将这些输出向量拼接起来，传入全连接网络层计算出最后的相似度，代表传入的两个音频是同一乐曲的不同版本的概率。

4.2 不同损失函数的训练思路

Jiang在论文中利用计算得到相似度，使用二元交叉熵损失Binary Cross Entropy Loss（BCE）来进行训练。这种训练方式不同于使用定长向量的相似度计算，而是将神经网络引入了相似度的计算过程。如果我们仅仅使用网络的最后输出来进行训练，那么BCE Loss可以说是最优选择。但事实上我们可以将第一部分网络输出的多深度序列拼接起来，作为我们学习得到的音乐特征表示，并使用这部分输出进行一些损失函数的计算和训练。

本节将对各种损失函数进行介绍，并对使用它们进行训练的思路和流程进行阐述。

4.2.1 使用二元交叉熵损失训练

网络的提出者Jiang使用Binary Cross Entropy Loss（BCE Loss），即二元交叉熵损失进行训练[1]。BCE 损失是交叉熵（Cross Entropy）损失的一种特殊形式，常用于二元分类问题。在这里，网络最后的输出值经过规范化后可以作为输入的两个音频是否为同一歌曲的不同版本的概率，本质上也是一个二分类问题，因此我们可以在这里使用BCE损失。

设网络的输入的第n组训练数据为{（）,}，（）是我们提取的两个音频的CQ时频图，维度均为84\*100；代表两个音频是否为来自同一歌曲，那么在网络中使用BCE损失的公式如下：

在这里我们对于BCE损失的使用就十分类似于孪生神经网络中常用的对比损失（contrasive loss），我们使用该损失进行训练的目的是增强网络对于一对样本数据的相似程度的判断能力。当两个音频来自于同一样本时，即为1时，Loss中第二项就为零，并且此时我们希望预测得到的两个音频是来自同一歌曲的预测概率尽可能的大，那么我们的目标就是最小化与的交叉熵，缩小学习得到的概率分布与实际分布的差距。而当为0时，则与之相反。

4.2.2 引入三元组损失与交叉熵损失

在深度学习领域，训练模式基本分为两种：第一种是基于成对训练数据的度量学习模式，第二种是基于输出分类概率的分类学习模式。如果我们单纯的使用BCE损失进行训练，虽然也能得到很好的效果，但却基本只考虑了度量学习的模式。在计算机视觉领域，很多研究者会将不同深度卷积块的输出拼接起来作为输入数据的特征表示，我们可以参考这一点使用第一部分网络的输出构造特征向量，从而引入分类学习的方式。在这里，我们将深度为2、3、4的卷积块的输出，即，拼接为一个定长向量（）作为音频的特征，并在特征上引入三元组损失（triplet loss）及交叉熵损失（cross entropy loss）来优化特征表示。

这里首先介绍三元组损失（triplet loss），它经常在度量学习问题中被使用，我们可以引入三元组损失来强化我们度量学习的效果。和我们之前所使用的对比损失的思路不同的是，三元组损失每次需要输入三组数据：一组固定数据（Anchor），一组正样本（Positive），以及一组负样本（Negative）。在我们的问题中正样本代表与固定数据来自同一歌曲的音频数据，负样本则相反，也就是说我们的三组输入可以包含一组正样本对（Anchor and Positive）和一组负样本对（Anchor and Negative）。三元组损失的训练目的是尽可能的拉近正样本对之间的距离，推开负样本对之间的距离，增强模型正确表达样本间距离的能力。设我们共有N组数据，而得到的第i组目标数据、正样本数据、负样本数据的特征表示分别为，那么三元组损失的公式如下：

分别为正样本对及负样本对之间的欧氏距离，为两个距离间的最小间隔（margin）。可以看出，只有当与间的差值小于时，损失才大于零，否则损失为零。当 margin 值越小时，Anchor 与 Positive 不需要拉的太近，Anchor 与 Negative 也不需要拉的太远，这样能使得 loss 很快的趋近于 0，但由于类内距离较大且类间距离较小，会降低训练效果。反之，若margin值设置的过大，又会使训练难度过大，导致loss无法收敛。本文将margin值设置为0.9。

既然我们得到了音频的特征表示，那么我们也可以使用分类的思路来进行训练，增强网络对于特征类别正确表示的能力。这里我们加入一个全连接层，分别将输入并得到维度为10000的输出，10000代表的是训练集中所有的歌曲类别数。要注意的是，为了防止过拟合，我们需要在全连接层后加入一个dropout层来按照一定的概率使神经元失活（即输出为0），我们选择的失活概率为20%。将这三个输出使用softmax进行归一化后我们就可以得到三个数值在0到1之间的1\*10000维向量，Anchor的softmax的公式如下（正负例同理）：

得到的向量中每个维度的值就可以作为这个音频属于该维度所代表歌曲的概率。在这三个向量上我们使用交叉熵损失，公式如下：

这里、代表对应数据的one-hot向量，在正确类别的位置值为1，其他位置都为0。

同理，我们分别计算正负样本对间的BCE损失，并将这三组损失组合起来进行多损失训练。由于每个损失在优化的过程中都不会考虑其他损失的优化情况，我们需要对每个损失加以一个权重来控制不同损失对于总损失的影响程度，公式如下：

由于我们最后使用模型时是依据模型最后的输出，通过度量音频间的距离来判断他们是否来自同一歌曲，因此基于度量学习的损失函数可能更适合这一任务，也就说是我们需要让BCE损失和三元组损失的权重更大些。在这里我们的设置为。

4.2.3 使用基于角度距离的损失函数

我们的方法实际上是判断两个音频的CQ时频图的相似程度，这和人脸识别的目标十分接近。与此同时，音乐和人脸都有着类内样本较少且类别较多的特性，而很多计算机视觉领域的研究者提出了一些可以增大类间距离的使用角度距离的损失函数来利用这一特性优化模型的训练效果。这里我们测试的基于角度距离的损失函数包括Large Margin Softmax（L-Softmax）[5]、CosFace[2]、SphereFace[4]。由于这些损失在提出时主要是为了改进传统的全连接层加Softmax CE损失，并且都是在其基础上进行了些许改变，因此应用方式完全和前文所述的类似，这里不赘述实现细节，而专注于介绍函数原理。

**4.2.3.1 Large Margin Softmax**

Large Margin Softmax（L-Softmax）由Weiyang Liu等人在2016年所提出，其目标是改进Softmax使其分类条件更加严苛[5]。想要理解L-Softmax，我们首先要观察Softmax。按照类似前文的风格，我们设全连接层的输入为，设最后得到的标签概率向量为，的第j个元素可以表示为，这里W为连接层的权重矩阵（为了表达简单省略了a，p，n的标识）。而这里是两个向量的内积，因此我们可以将其表示为，为两个向量的夹角。那么假设我们有一个二分类问题，我们的输入属于第一类，那么我们希望的是：

这样就说明输入属于第一类的概率更大。而Large Margin Softmax的核心思想就是将判断条件改为

由于cos函数在0到π上单调递减，因此。也就是说，m的值越大，学习得到的类间间隔就越大，同时学习的难度也越大（当m为1时退化为普通的Softmax）。在此基础上，论文中给出了L-Softmax的公式：

这里有：

其中是一个满足

的单调递减函数。我们实现时和原文中使用相同的函数，即：

k为[0, m – 1]上的整数。

**4.2.3.2 SphereFace**

SphereFace，即A-Softmax，同样由Weiyang Liu提出，其思路与L-Softmax十分类似，都是修改了原始Softmax的判断条件，加上了一个系数m计算乘法间隔[4]。但L-Softmax在计算距离是未对权重向量W进行归一化，这样在优化时不仅学习了角度的余弦值，还隐式地学习了特征销量的L2 norm，这会导致我们对cosine的约束变弱。而Sphereface则将W进行归一化，使得，这样整个优化过程就只依赖于余弦值来获得具有区分力的特征，使预测更加精确。除此之外其他原理以及公式基本与L-Softmax无异，在这里不做赘述。

**4.2.3.3 CosFace**

CosFace[2]由Wang等人在2018年提出。和SphereFace类似，它也将权重归一化。同时，作者认为计算角度距离时所用的输入向量的长度也没有太大的作用，且若其值过小，会导致训练loss过大（softmax值过小），于是他直接将其固定为一个值s。由于W被归一化，s值为常数，此时预测类别为1的判断条件为：

前面两个损失函数中，都是使用m对角度进行约束。但此时由于约束带来的间隔margin是不统一的，若越小，则margin越小。因此作者考虑将m用作加法来对cosine值进行约束，这样可以使m约束带来的间隔影响不随的改变而改变，更加平衡且稳定。那么判断条件变为：

我们在上文所述条件的基础上修改Softmax CE损失就可以得到CosFace损失的表达，其公式如下：

4.3 实验结果及分析

和绝大多数的深度学习任务一样，我们选择Python语言来编写我们的实验程序。在模型方面，我们使用Pytorch来进行模型的搭建以及训练。在数据预处理方面，我们使用Librosa库中的函数来提取音频的Constant-Q时频图。由于不同音频的时长不同，为了统一网络的输入维度，我们随机选择乐曲中的两分钟片段来提取CQT特征。若音频的时长小于两分钟，我们会在其缺失部分补零。经过这样的处理，我们得到的CQ时频图的维度为84\*100，84为频率维度、100为乐曲长度。同时，我们使用mini-batch的训练方法，每次将20份输入数据组成一个batch输入进神经网络进行训练。在优化器方面，我们选择Adam来优化损失，初始学习率为，权重衰减为。在训练中我们还会根据训练的具体情况来调节学习率，若损失连续四个epoch下降程度小于阈值，就将学习率调整为原来的0.8倍。

由于4.2.3中所介绍的三个损失函数是Softmax CE损失的改进，我们将多损失训练中的Softmax CE损失分别替换为这三个损失来进行训练，并和原方法进行比较。

# 第五章 音乐版本鉴别原型系统的构建

在前面我们经过训练得到了一个性能最优的模型，而单单有了模型是远远不够的，我们还需要将这一模型实用化，构建一个音乐版本鉴别原型系统来实现其应用价值。

为了让系统的应用更加方便，界面更加美观，我们选择将其以网页的形式实现。本章首先分析系统所要满足的应用需求，再根据需求总体介绍系统的技术路线，之后对各个功能模块的实现方法进行详细介绍。

5.1 应用需求

既然我们所训练的模型是通过计算音频间的相似度来进行音乐版本的鉴别，那么我们首先要实现的就应该是让用户也能使用我们的系统来尝试模型的效果。根据我们最开始所介绍的翻唱版本鉴别问题的定义，用户给出一个音频文件作为查询，我们应该返回乐库中和其最为相似的一些音频文件。这很接近各大音乐软件中的“听歌识曲”功能。但与之不同的是，“听歌识曲”很多时候无法正确识别环境中播放的音乐，而且仅能返回一个查询结果。而我们通过输入音频文件进行查询，可以返回很多与之可能是同一歌曲的结果，这样的查询结果更加精确、多元。

其次，当我们返回音频文件后，若只将乐曲的各种信息罗列出来，用户无法了解查询结果是否正确。因此，我们还需要让用户能够在线播放这些返回的歌曲，用自己的耳朵来感受深度学习的力量。

同时，既然我们要实现的是一个系统，那么我们还需要能够对后台的音乐数据库进行操作。而我们应该提供给用户的接口应该为添加乐曲（删除、管理乐曲为管理员在后台的操作），这样可以借助用户的力量来扩大我们的乐库，为我们后续的研究提供更多方便。

此外，一些用户还可能会好奇某两首歌曲的相似度，将其作为是否侵权的参考。因此我们还应该允许用户使用模型计算自己所上传的两个音频的相似度，以其作为二者是来自同一原曲的概率。

5.2 技术路线

有了上一节所述的各种应用需求后，下一步要做的就是根据这些需求来制定一条技术路线。我们的最主要目的是让任何一个用户都能轻松地体验我们所得模型的效果，这就需要我们让整个操作流程简单化。因此我们选择了响应式网页这一方案，将网页上线后任何用户都可以登录网站，通过和网页的交互来感受深度学习的效果。

想要实现一个响应式网页，我们首先需要选择一个合适的前端框架来构建我们的网页界面。本文选择Twitter基于HTML、CSS、JavaScript所开发的Bootstrap[[1]](#footnote-1)作为我们的前端框架。它提供优雅的HTML和CSS规范，包含了极多的可用控件，可以极大方便我们的前端页面编写。我们用HTML语言构建网页框架，JavaScript语言来编写交互脚本，CSS语言细化网页外观。

而同时，我们需要在后端进行各种数据的预处理、利用深度学习模型计算相似度、乐库信息的管理等。为了和前文的研究工作更好的对接，我们同样使用Python语言来实现后端的各种功能。在深度学习模型的处理和使用方面，依旧需要使用Pytorch。

最后要考虑的就是如何将前后端连接在一起。由于在后端使用了Python语言，我们选择Flask框架[[2]](#footnote-2)，利用HTML请求和Ajax技术来进行前后端的交互。Flask框架十分轻量，应用灵活，可以自由定制，恰好满足我们项目的要求。

根据我们上面所述的需求及技术路线分析，我们可以画出如下的技术路线图：

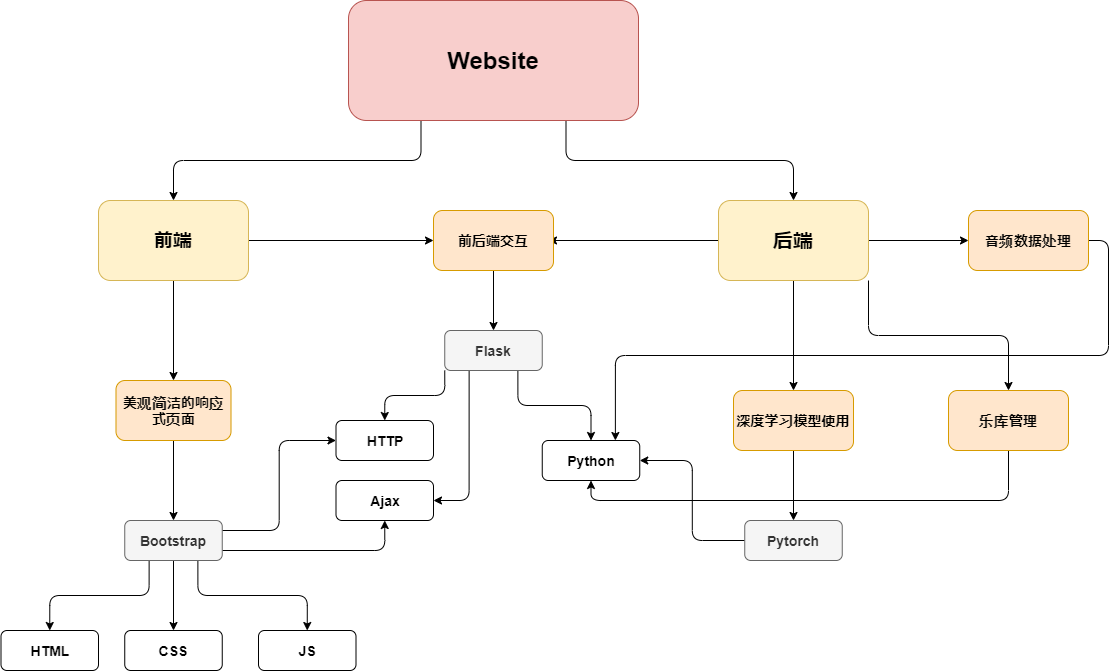


图5.1 系统的技术路线图

5.3 功能模块具体实现

根据需求，我们将功能划分为三个模块：“以曲搜曲”模块、相似判断模块以及添加乐曲模块。我们按照模块编写了三个前端页面：分别对应上面三个模块的query.html、judge.html、add.html。本小节将从前端出发，详细介绍这几个模块的具体实现。

5.3.1 “以曲搜曲”模块

在此模块中，最先要解决的是如何让用户将自己所要查询的音频文件传送给服务器，在这里我们使用Bootstrap-fileinput[[3]](#footnote-3)插件来实现用户上传功能。我们通过编辑插件的CSS文件来更改插件的外观（修改了字体及显示文字、增加了毛玻璃效果等），编辑插件的JS文件来更改插件的设置细节（一次最多上传一个文件，使用Ajax的方式上传）。文件上传页面如下图：

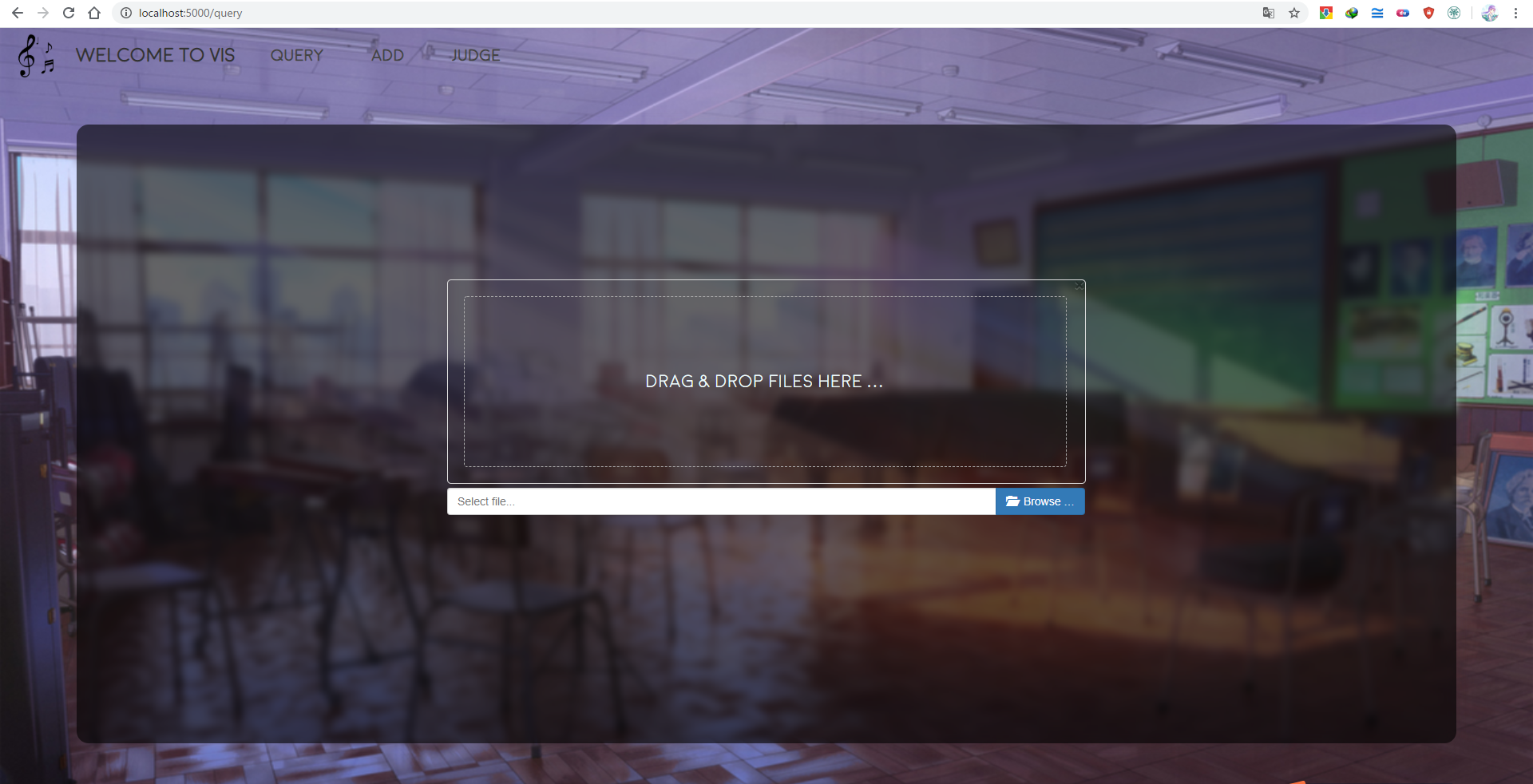


图5.2 文件上传页面

**参考文献**

1. Jiang C, Yang D, and Chen X. “*Similarity Learning For Cover Song Identification U-sing Cross-Similarity Matrices of Multi-Level Deep Sequences*.” In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, **2020**: 26-30.
2. Wang H, Wang Y, Zhou Z *et al*. “*Cosface: Large margin cosine loss for deep face r­ecognition.*” In: *Proceedings of the IEEE Conference on Computer Vision and Patter-n Recognition*. **2018**: 5265-5274.
3. Deng J, Guo J, Xue N *et al.* “*Arcface: Additive angular margin loss for deep face r-ecognition.*” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. **2019**: 4690-4699.
4. Liu W, Wen Y, Yu Z *et al*. “*Sphereface: Deep hypersphere embedding for face reco-g­nition*.” In: *Proceedings of the IEEE conference on computer vision and pattern rec-o­gnition*.**2017**: 212-220.
5. Liu W, Wen Y, Yu Z *et al*. “*Large-margin softmax loss for convolutional neural net-w­orks*.” In: *ICML*. **2016**,2(3): 7.
6. Tralie CJ and Bendich P. “*Cover song identification with timbral shape sequences*”.*a-r­Xiv preprint arXiv:1507.05143*, **2015**.
7. Sun Y, Cheng C, Zhang Y *et al*. “*Circle loss: A unified perspective of pair similarity optimization*.” *arXiv preprint arXiv:2002.1085*7, **2020**.
8. Fujishima T. “*Real-time chord recognition of musical sound: A system using common lisp music*”.*Proc. ICMC, Oct. 1999*, **1999**: 464–467.
9. Tzanetakis G, Ermolinskyi A and Cook P. “*Pitch histograms in audio and symbolicm-usic information retrieval*”. *Journal of New Music Research*, **2003**, 32(2): 143–152.
10. Müller M, Kurth F and Clausen M. “*Audio Matching via Chroma-Based Statistical F-e­a­tures*.” In: *ISMIR*, **2005**: 6th.
11. Ellis DP and Poliner GE. “*Identifying cover songs’ with chroma features and dynamicprogramming beat tracking*”. In: *2007 IEEE International Conference on Acoustics, S-p­eech and SignalProcessing-ICASSP’0*7, **2007**: IV–1429.
12. Serra J. “*Identification of versions of the same musical composition by processing au-d­io descriptions*”. *Department of Information and Communication Technologies*, **2011**.
13. Serra J, Gómez E and Herrera P. “*Audio cover song identification and similarity: background, approaches, evaluation, and beyond*”. In: *Advances in M­us­ic Information Re-t­rieval*. Springer, **2010**: 307–332.
14. Brown JC. “*Calculation of a constant Q spectral transform*”. *The Journal of the Aco-ustical Society of America*, **1991**, 89(1): 425–434.
15. Ellis DP and Thierry BM. “*Large-scale cover song recognition using the 2DFourier t-ransform magnitude*”. **2012**.
16. Bello JP. “*Audio-Based Cover Song Retrieval Using Approximate Chord Sequences: T-e­sting Shifts, Gaps, Swaps and Beats*.” In: *ISMIR*, **2007**: 239–244.
17. Silva DF, Falcão F and Andrade N. “*Summarizing and Comparing Music Dataand Its Application on Cover Song Identification*.” In: *ISMIR*, **2018**: 732–739.
18. Xu X, Chen X and Yang D. “*Key-invariant convolutional neural network toward effi-c­ient cover song identification*”. In: *2018 IEEE International Conference on Multimed-i­a and Expo(ICME)*, **2018**: 1–6.
19. Ellis DP. “*The" covers80" cover song data set*”. URL: *http://labrosa.ee.columbia.edu/p­rojects/coversongs/covers80*, **2007**.
20. Bello JP. “*Measuring structural similarity in music*”. *IEEE Transactions on Audio, S-p­eech, and Language Processing*, **2011**, 19(7): 2013–2025.
21. Silva DF, de Souza VM and Batista GE. “*Music Shapelets for FastCover Song Reco-g­nition*.” In: *ISMIR*, **2015**: 441–447.

1. https://www.bootcss.com/ [↑](#footnote-ref-1)
2. https://palletsprojects.com/p/flask/ [↑](#footnote-ref-2)
3. http://bootstrap-fileinput.com/ [↑](#footnote-ref-3)