Label-Specific Time-Frequency Energy-based Neural Network for Instrument Recognition

Jian Zhang, Tong Wei, Min-Ling Zhang, *Senior Member, IEEE*

***Abstract*—Predominant instrument recognition plays a vital role in music information retrieval. This task involves identifying and categorizing the dominant instruments present in a piece of music based on their distinctive time-frequency characteristics and harmonic distribution. Existing predominant instrument recognition approaches mainly focus on learning implicit mappings (such as deep neural networks) from time-domain or frequency-domain representations of music audio to instrument labels. However, different instruments playing in polyphonic music produce local superposed time-frequency representations while most implicit models could be sensitive to such local data changes. This thus poses a challenge for these implicit methods to accurately capture the unique harmonic features of each instrument. To address this challenge, considering that the complete harmonic information of an instrument is also distributed across a wide range of frequencies, we design a label-specific time-frequency feature learning approach to convert the task of building implicit classification mappings into the process of extracting and matching features that are specific to each instrument, as a result, a new explicit learning model: Label-Specific Time-frequency energy-based neural Network (LSTN) is proposed. Unlike existing implicit models, LSTN not only extracts their commonly used local time-frequency features but also incorporates time-domain factors and frequency-domain factors in its energy function to explicitly parameterize the long-term correlation and long-frequency correlation features. Using the extracted time-frequency features and the two long correlation features as instrument label-specific features, LSTN detects whether the harmonic distribution of each instrument appears in polyphonic music on both long time-frequency scales and local time-frequency scales to mitigate the challenges posed by local superposed representations. We conduct an analysis of the complexity and the convergence of LSTN, then experiments conducted on benchmark datasets demonstrate the superiority of LSTN over other established instrument recognition algorithms.**

***Index Terms*—Multi-label learning; Instrument recognition; Deep belief network; Boltzmann machine**

I. Introduction

PREDOMINANT instrument recognition is a process that focuses on identifying the primary instruments within a musical composition which contains multiple instruments. This field has significant implications for various music-related tasks, such as playlist generation and music generation [1]. In recent years, machine learning-based predominant instrument recognition models have gained significant attention. Common strategies employed by traditional machine learning approaches involve constructing time-frequency representations, including zero-crossing rate, spectral centroid, Mel-spectrogram, and Mel-scale frequency cepstral coefficients (MFCC) [2], from music signals. These representations are then utilized to recognize instruments using approaches such as support vector machine (SVM), long short-term memory (LSTM), etc. [3-4]. With the rise of deep neural networks in signal processing, a different strategy has emerged, which involves organizing these time-frequency representations into spectrograms, and deep convolutional neural networks (CNN) or self-attention neural networks are then applied to learn implicit mappings from the spectrograms to multiple instrument labels [5]. However, these implicit models usually focus on learning the data distribution to establish a mapping with labels. As a result, they can be sensitive to local changes in the data distribution [6], even if these changes do not alter the semantic content of the input [7-8]. In the context of the predominant instrument recognition task, detecting specific instruments from a mixture of different instruments is commonly required. When various instruments are playing in a piece of music, partial harmonic superposition occurs, giving rise to new local data distributions in the time-frequency representations (as Fig. 1 shows). Consequently,

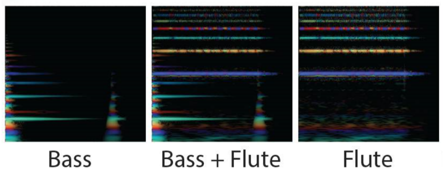


Fig. 1. An example of local superposed representations of Flute and Bass. It should be emphasized that playing two instruments simultaneously at different pitches will result in different superposed representations, which further increases the complexity of time-frequency representations [9].

due to the sensitivity of these implicit instrument recognition models, it becomes challenging to directly apply these models to recognize the superposed representations and distributions from different instruments in different frequency bands.

The task of predominant instrument recognition focuses on polyphonic music, which usually contains multiple instruments. This characteristic converts instrument recognition in such scenarios into a multi-label learning task. Multi-label learning addresses the challenge where each example is represented by a single instance (feature) while associated with a set of labels [10]. As an effective multi-label learning strategy, label-specific features, i.e., the most pertinent and

Jian Zhang is with the School of of Computer Science and Engineering, Southeast University, Nanjing 210096, China, and also with the School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China.

Tong Wei and Min-Ling Zhang are with the School of Computer Science and Engineering, Southeast University, Nanjing 210096, China, and the Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China.

Corresponding author: Min-Ling Zhang, Email: zhangml@seu.edu.cn.

discriminative features for each class label, are exploited from multi-label data. This strategy converts the task of building classification mappings into the process of extracting and matching features that are specific to each label. Therefore, the algorithm's emphasis shifts to creating specific features that are highly adaptable and customized for individual labels [11], aligning with the requirements of the predominant instrument recognition task.

In view of label-specific feature learning, there exists a fact that each type of instrument has its label-specific time-frequency characteristics and harmonic distribution [12]. Furthermore, these harmonic properties are not confined to local scales but can also stretch across wider scales in both the time-domain and frequency-domain. Based on this observation, we hypothesize that by specifically learning the local time-frequency features, the long-term and long-frequency features as instrument label-specific features, then using these label-specific features to detect whether the harmonic distribution of each instrument appears in polyphonic music on both long time-frequency scales and local time-frequency scales to alleviate the local sensitivity of the implicit models, a more effective approach for instrument recognition could be achieved. Moreover, the harmonics of instruments are explicitly present in both the time-domain and the frequency-domain. This attribute lends itself well to being modeled as explicit distributions, providing an intuitive methodology. Additionally, when extracting harmonic features across extended scales, utilizing a holistical parameterized distribution may be more intuitive in capturing the predominant spectrum of an instrument's harmonics. This perspective aligns with the common practice of employing energy-based models to explicitly represent data distribution [13]. These models provide flexibility in defining the necessary variables within their energy functions. Therefore, we explore an strategy to explicitly learn label-specific time-frequency features based on deep energy-based neural networks.

In this paper, a deep energy-based neural network named Label-Specific Time-frequency energy-based neural Network (LSTN), is proposed. Based on the aforementioned learning strategy, LSTN can be treated as a combination of two core modules. The first module is a convolutional Time-frequency correlation energy-based Neural Network (TNN), which explicitly extracts time-frequency features, long-term correlation, and long-frequency correlation features using a specially designed time-frequency energy function. The second module, the instrument Label-Specific energy-based classification neural Network (LSN), is developed on top of the TNN. Using constructed label-specific energy functions, the LSN separates instrument label-specific features from the extracted features in the TNN, allowing for explicit modeling of label-specific harmonic distributions, and ultimately outputting predicted instrument labels. Moreover, as a deep energy-based neural network, LSTN offers flexibility in terms of training methods. The first training approach involves using greedy methods, where the TNN and LSN are separately trained using sampling-based techniques. Alternatively, an end-to-end training can be performed by utilizing an unified score matching-based method. Furthermore, we analyze the complexity and convergence of LSTN. The main contributions of this paper are listed as follows:

(1) As the complete harmonics of an instrument are scattered over a long scale, to model the harmonic distribution of each instrument, LSTN not only extracts the commonly used local time-frequency features but also designs time-domain and frequency-domain factors to parameterize the long-term correlation and the long-frequency correlation as explicit probabilities.
(2) To alleviate the effects of the partial superposed time-frequency representations and distributions in polyphonic music, LSTN uses the extracted time-frequency features and the two long correlation features of harmonic distribution as instrument label-specific features, which provides a large time-scale and a general frequency-scale to detect whether the harmonic distribution of each instrument appears in polyphonic music.
(3) To effectively evaluate the performance of the proposed LSTN model, comparative studies over typical instrument recognition datasets have been conducted. Experimental results show that: (a) LSTN achieves most of the superior performance against several state-of-the-art instrument recognition algorithms. (b) LSTN effectively extracts explicit label-specific time-frequency features which can be transformed to different polyphonic music. (c) Furthermore, we conduct ablation experiments to confirm the compatibility and the resilience to noise of LSTN.

The rest of this paper is organized as follows. Section II briefly reviews existing approaches to instrument recognition, multi-label learning, and energy based models. Section III presents the proposed LSTN model. Section IV reports comparative experimental results over some representative instrument recognition data sets. Finally, section V concludes and discusses several issues for future work.

II. Related Work

A. *Instrument recognition approaches*

Traditional instrument recognition approaches commonly extract time-frequency features manually from music audio and then use simple classifiers for recognition. Following this strategy, Wang et al. [2] employ a feature representation based on the spectral peak of the fundamental frequency of the harmonic sequence to identify musical instruments. Unfortunately, these traditional approaches usually surfer from complex feature extraction process and poor instrument recognition accuracy. In recent years, deep learning approaches are widely used for signal processing, based on deep neural networks, some strategies for building effective instrument recognition models are applied.

1) ***Build powerful deep neural networks:*** Usually, deep neural networks trend to organize the time-frequency representations as a spectrogram or a cascade combination and then learn the data distribution to build implicit mappings with labels. Among them, authors use CNN or LSTM for instrument recognition. They train a deep neural network with one-second audio clips and a temporal max-pooling aggregation is made on several contiguous predictions according to the desired time resolution. Han et al. [5] exploit the proposed ConvNet structure for instrument recognition. The mel-frequency spectrogram is used as the input of ConvNet. The single-labelled training data of IRMAS [14] is used to train the network and the multi-labelled testing data is used to recognize the predominant instrument.

*2) **Characteristic of predominant instrument recognition**:* Predominant instrument recognition is a process that focuses on identifying the primary instruments within a musical composition which includes multiple instruments. During training, the music piece may include only one or a few instruments of interest, whereas the test dataset may contain multiple instruments. Consequently, label correlations may be incomplete in the training process, rendering some label correlation-based multi-label learning frameworks unsuitable for some predominant instrument recognition tasks in datasets like IRMAS, and this dataset features single-labeled training data. Therefore, emphasizing the learning of individual instrument features becomes crucial for predominant instrument recognition. To address this, various methods involving data augmentation or label augmentation have been proposed for instrument recognition.

*3) **Introduce data augmentation or label augmentation to deep neural networks**:* For deep neural networks, sufficiently mining the relation between data distribution and labels is very important. To provide more expressive data and labels for deep neural networks, scholars introduce additional methods from the viewpoint of data augmentation to construct more effective instrument recognition models. Aiming at generating more expressive labels, Yu et al. construct a network with an auxiliary classification designed based on the onset groups and instrument families [15]. The principal classification and the auxiliary classification enable the network to learn the instrument categories and groups jointly in a pattern of multi-task learning. On the other hand, in Reference[16], predominant instrument recognition in polyphonic music is addressed using convolutional recurrent neural networks (CRNN) through Mel-spectrogram, modgdgram, and their fusion. In their works, a wave generative adversarial network (WaveGAN) is employed to generate audio files for data augmentation [17-18].

B. *multi-label learning and energy based models*

Recently, multi-label learning models are applied for multiple music information retrieval tasks. For instrument recognition, several effective approaches can be applied, such as multi-label learning with Label-specific FeaTures (LIFT), and Cross-Coupling Aggregation (COCOA) [19]. Reference [20] also explores an attention mechanism for handling weakly labeled data for multi-label instrument recognition. However, rare research designs multi-label learning approaches specially for the music audio, in particular building explicit instrument feature learning and recognition processes. Moreover, energy based models are commonly applied for explicitly modeling data correlation and data distribution, among them, reference [21] uses a deep belief network for extracting features and SVM for instrument recognition. Meanwhile, many effective energy based models are proposed to model data distribution, such as score matching [22], spike and slab RBM [23], but they are rarely used or designed for modeling time-frequency features for music audio. Recently, there has been significant interest in label-specific feature learning. Some methods focus on embedding label correlations into these features for classification purposes [24], and the correlation information can be modeled at different levels [25]. These approaches offer hybrid multi-label learning frameworks of label-specific feature and label correlation. However, in the context of predominant instrument recognition tasks, the label correlation information is often incomplete, which may render these hybrid frameworks unsuitable.

As reviewed above, existing approaches have their common properties of recognizing instruments. In the next section, a new approach named LSTN is proposed.

III. The LSTN Approach

As mentioned above, LSTN aims to explicitly extract time-frequency features from time-frequency representations based on specially designed energy functions and separates instrument label-specific features from the extracted time-frequency features to model instrument harmonic distributions. The LST-N structure is shown in Fig.2. This section provides a detailed introduction to the LSTN, including the time-frequency representations preprocessing, algorithmic details, convergence analysis, feature analysis, and complexity analysis.

A. *Extract Time-Frequency representations as preprocessing*

The music audio is transformed into 2-dimensional sound spectrograms, where the *X-axis* describes time-domain information and the *Y-axis* describes harmonic information in frequency-domain. In the first preprocessing step, we normalize the gain of data to -15DB, and the stereo input audio is converted to mono by taking the mean of the left and right channels, and then it is downsampled to 22,050 *Hz*. Secondly, all music samples are normalized by dividing the time-domain signal with its maximum value, and then it is converted to a time-frequency representation using short-time Fourier transform with 1024 samples for the window size and 512 samples of the hop size. Next, the linear frequency scale-obtained spectrogram is converted to a Mel-scale. We use 128 for the number of Mel-frequency bins, following the representation learning papers on music annotation [5, 21]. Thirdly, the magnitude of the obtained Mel-frequency spectrogram is compressed with a natural logarithm. Finally, the commonly used features such as zero-crossing rate, spectral centroid, root mean square, spectral roll-off, bandwidth, Mel-spectogram, and MFCC are concatenated to the log Mel-frequency spectrogram to build the time-frequency representations [15]. Moreover, some effective pretrained features can be used as well.

B. *Algorithmic Details*

As an energy-based model, LSTN offers advantages of flexibility in defining the essential variables in energy functions and activation probabilities. Taking advantage of this flexibility, LSTN incorporates specially designed factors in its energy

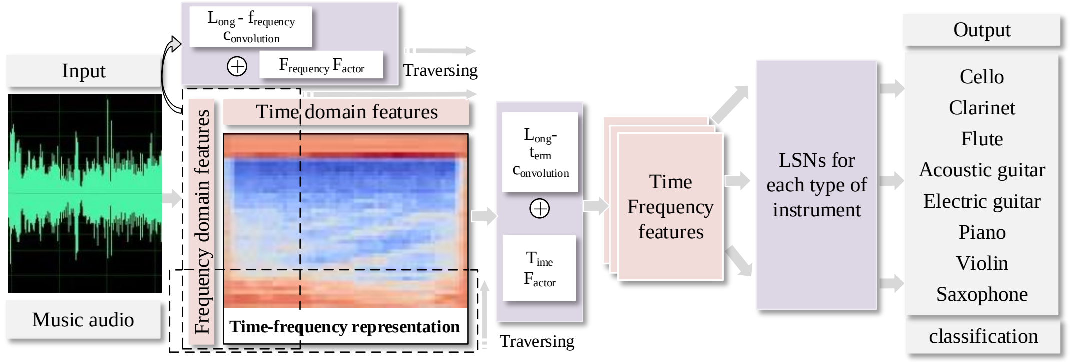


Fig. 2. The LSTN structure. The proposed LSTN designs long-term convolution and long-frequency convolution by using time-domain and frequency-domain factors to parameterize the long-term correlation and the long-frequency correlation as explicit probabilities. Then, LSTN separates instrument label-specific features from the extracted features and builds label-specific instrument classifiers.

functions to learn time-frequency features and instrument label-specific features from the input time-frequency representations. Structurally, LSTN achieves instrument recognition by sequentially passing information through two core modules: Time-frequency correlation energy-based Neural Network module (TNN) and Label-Specific energy-based classification neural Network module (LSN). Implementation details of the designed two modules in LSTN are as follows:

*1) **TNN**:* To model the harmonic distribution, following the fact that complete harmonics of an instrument are scattered over a large frequency-domain scale, TNN designs time-domain and frequency-domain factors to parameterize the long-term correlation and the long-frequency correlation as explicit probabilities. In TNN, an energy function that contains input timeCfrequency representations x, long-term correlation term $T-term$, long-frequency correlation term $F-term$, and timeCfrequency term $TF-term$ is designed as

$$\mathrm{E}(x, \textit{T-term}, \textit{F-term}, \textit{TF-term}) = \frac{1}{2}\alpha x^2 + \textit{T-term}(x,c_t,h_1) + \textit{F-term}(x,c_f,h_2) + \textit{TF-term}(x,h_1,h_2), \quad (1)$$

the long-term correlation term *T-term* is a function of the input x, the time domain factor c_t, and the corresponding time-frequency features h_1. We denote convolution and transposed convolution operations as $\otimes$ and $\odot$ in this paper. Ulteriorly, the long-term correlation term can be denoted as follow:

$$\textit{T-term}(x,c_t,h_1) = \frac{1}{2}\beta_1 c_t^2 - c_t \odot W_{c_t} h_1 - x \otimes W_1 c_t \odot W_1^* h_1,$$

where *T-term* uses variable h_1 and c_t to describe time-domain features by traversing time-frequency representations x with a common convolution (by using kernel W_{h_1}) and a long-term convolution (by using kernel W_1), among them, $W_{h_1} \in R^{K*K}$, $W_1 \in R^{K*T}$, T is the length of the current time slice, K is a hyperparameter. Using c_t and the long-term convolution kernel W_1 to cover the whole time slice, TNN aims at modeling the correlation of time-frequency representations in time-domain.

Similarly, the long-frequency correlation term *F-term* is a function of the input x, the frequency-domain factor c_f, and the corresponding time-frequency features h_2. The long-frequency correlation term can be denoted as follow:

$$\textit{F-term}(x,c_f,h_2) = \frac{1}{2}\beta_2 c_f^2 - c_f \odot W_{c_f} h_2 - x \otimes W_2 c_f \odot W_2^* h_2,$$

where *F-term* uses variable h_2 and c_f to describe frequency-domain features by traversing time-frequency representations x with a common convolution (by using kernel W_{h_2}) and a long-frequency convolution (by using kernel W_2), among them, $W_2 \in R^{F*K}$, and F is the height of frequency representation. Using c_f and the long-frequency convolution kernel W_2 to cover the whole frequency-domain representation, TNN aims at modeling the correlation of time-frequency representations in the frequency-domain.

In addition to constructing the *T-term* and *F-term*, h_1 and h_2 are also used for modeling time-frequency features by using the *TF-term* in energy function, which is expressed as follows:

$$\textit{TF-term}(x,h_1,h_2) = -x \otimes W_{h_1} h_1 - x \otimes W_{h_2} h_2.$$

Based on the energy function, h_1 is activated by:

$$p(h_1 = 1|x) = \mathrm{Sigmoid}(x \otimes W_{h_1} + \frac{1}{2}\beta_1^{-1} \odot W_{c_t}^2 + \frac{1}{2}\beta_1^{-1} \odot W_1^* (x \otimes W_1)^2 \odot W_1^* + \beta_1^{-1} \odot W_{c_t} (x \otimes W_1) \odot W_1^* + b_1), \quad (2)$$

h_1 is a 2-dimensional time-frequency feature activated by x. The time-domain factor c_t can be activated as a 1-dimensional vector by convolving special scale kernels W_1, W_1^*, and W_{ct} based on x and h_1:

$$p(c_t|x,h_1) = \mathcal{N}\left(\beta_1^{-1}(x \otimes W_1(h_1 \otimes W_1^*) + h_1 \otimes W_{ct}), \Lambda_1\right), (3)$$

where Λ_1 is a diagonal matrix built by β_1^{-1}, $\mathcal{N}$ denotes Gaussian distribution. Similar to h_1, h_2 is activated by:

$$p(h_2 = 1|x) = \mathrm{Sigmoid}(x \otimes W_{h_2} + \frac{1}{2}\beta_2^{-1} \odot W_{c_f}^2 + \frac{1}{2}\beta_2^{-1} \odot W_2^* (x \otimes W_2)^2 \odot W_2^* + \beta_2^{-1} \odot W_{c_f} (x \otimes W_2) \odot W_2^* + b_2). \quad (4)$$

The frequency-domain factor c_f is activated by:

$$p(c_f|x,h_2) = \mathcal{N}\left(\beta_2^{-1}(x \otimes W_2(h_2 \otimes W_2^*) + h_2 \otimes W_{cf}), \Lambda_2\right), (5)$$

where Λ_2 is a diagonal matrix built by β_2^{-1}. Based on x and h_2, the frequency-domain factor c_f can also be expressed as a 1-dimensional vector by convolving its special scale kernels W_2, W_2^*, and W_{cf}. The data x can be passed based on

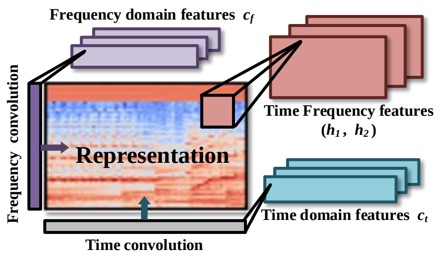

Frequency domain features c_f
Time Frequency features (h_1, h_2)
Representation
Frequency convolution
Time domain features c_t
Time convolution

Fig. 3. The TNN structure and the time frequency convolution processes.

the activation of (h_1, h_2, c_1, c_2) as a time-frequency feature extracting process after the TNN being trained based on the following energy function:

$$\begin{aligned}\mathrm{F}(x) = \tfrac{1}{2}\alpha x^2 - \text{Softplus}(x\otimes W_{h1}+\tfrac{1}{2}\beta_1^{-1}\odot W_{ct}^2+b_1+\\ \tfrac{1}{2}\beta_1^{-1}\odot W_1^*\,(x\otimes W_1)^2\odot W_1^*+\beta_1^{-1}\odot W_{c_t}\,(x\otimes W_1)\odot W_1^*)\\ -\text{Softplus}(x\otimes W_{h2}+\tfrac{1}{2}\beta_2^{-1}\odot W_{cf}^2+b_2+\\ \tfrac{1}{2}\beta_2^{-1}\odot W_2^*\,(x\otimes W_2)^2\odot W_2^*+\beta_2^{-1}\odot W_{c_f}\,(x\otimes W_2)\odot W_2^*),\end{aligned} \quad (6)$$

where $\text{Softplus}(x) = \log(1+e^x)$. As an energy-based model, TNN can be trained using the constructed energy in multiple frames with various objective functions, such as Fisher divergence, KL divergence, or adversarial loss. These objective functions lead to different training approaches, these diverse training methods offer flexibility and options for effectively training the TNN model. Moreover, as a module of LSTN, TNN can be jointly trained as a component in an unified loss function of LSTN. After training, the explicit distribution of x can be calculated using the energy function:

$$\begin{aligned}p(x|h_1,h_2,c_t,c_f)\sim\\ \mathcal{N}(\alpha^{-1}(h_1\odot W_{h1}+(h_1\odot W_1^* c_t)\odot W_1+\\ h_2\odot W_{h2}+(h_2\odot W_2^* c_f)\odot W_2),\ \Phi),\end{aligned} \quad (7)$$

where Φ is a diagonal matrix built by α^{-1}.

By minimizing the energy of TNN, time-frequency features (h_1,h_2), time domain feature c_t, and frequency domain feature c_f are explicitly extracted as parameterized distributions from the input time-frequency representations. Fig. 3 illustrates the TNN structure and the time-frequency convolution processes.

*2) **LSN:*** LSTN designs a Label-Specific energy-based classification neural Network (LSN) on TNN. The LSN separates instrument label-specific features to explicitly model label-specific harmonic distributions from extracted time-frequency features and then outputs predicted instrument labels.

As a label-specific classifier, LSN is constructed for each class of instrument, and for a specified class l, the instrument label-specific energy function is designed as Formula (8):

$$\mathrm{E}(f,g,y_l) = \tfrac{1}{2}\mu f^2 - gWf - cg - dy_l - gUy_l, \quad (8)$$

where f denotes concatenated feature of $(h_1^{'},h_2^{'},c_f,c_t)$, W and U are weight matrices, c and d are biases, y_l is the corresponding label vector of class l, it should be emphasized that $h_1^{'}$ and $h_2^{'}$ are flattened vectors (or encoded vectors by using a CNN named $Encoder$) of matrices h_1 and h_2. The probability of y_l based on feature f is calculated as follow:

$$\begin{aligned}p(y_l=1|f) = \text{Sigmoid}(\sum_j \text{Softplus}(c_j+U_j+W_jf)-\\ \sum_j \text{Softplus}(c_j+W_jf)+d).\end{aligned} \quad (9)$$

To establish a recognition process, the LSN can be trained in various ways using Formula (9). One option is to treat the encoding process, denoted as $Encoder$, from (h_1,h_2) to $(h_1^{'},h_2^{'})$, and the classification process mentioned based on Formula (9) as separate components. Initially, a pre-training of an unique backbone $Encoder$ is conducted. Subsequently, LSNs for each instrument type are optimized. Moreover, to effectively train a label-specific classifier, LSN also minimizes the discriminative loss, and the loss function of LSN is expressed as follow:

$$\begin{aligned}Loss\,(\text{Data}) = L_1\,(\text{Data}) + L_2\,(\text{Data})\\ = -\sum_t \log\, p\left(y_l^{[t]}|f^{[t]}\right) - \sum_t \log\, p\left(f^{[t]}\right),\end{aligned}$$

where the loss function contains two terms: the first term is a discriminative loss, and the second term is a likelihood function, t represents the index.

Moreover, LSN can be trained in a frame of score matching or a frame of joint energy-based model [26]. Treating LSN as a joint energy-based model, each combination of the unique backbone $Encoder$ and a classifier of Formula (9) for each class of instruments can be jointly optimized as a deep energy-based neural network. For each label y, the $Encoder$ and the classifier of Formula (9) can be denoted as $q_\theta(\boldsymbol{h},\,\boldsymbol{c})$, the LSN can be trained according to the following loss function:

$$\begin{aligned}\text{Loss}_{LSN_y} = \text{CrossEntropy}(\frac{\exp(q_\theta(\boldsymbol{h},\,\boldsymbol{c})_y)}{\Sigma_y \exp(q_\theta(\boldsymbol{h},\,\boldsymbol{c})_y)})\\ -\log\sum_y \exp(q_\theta(\boldsymbol{h},\,\boldsymbol{c})),\end{aligned}$$

where, $\boldsymbol{h}$ are (h_1,h_2) and $\boldsymbol{c}$ are (c_1,c_2).

*3) **Training LSTN in an unified frame:*** As mentioned previously, the TNN, $Encoder$, and LSN in LSTN can be trained using a framework of the Joint Energy-based Model (JEM) with sliced score matching method. Let y_n represent the specific labels in LSN, where $n\in[0,N]$, and N is the total number of classes. When the classification processes for specific labels utilize a single $Encoder$, but employ distinct classifiers, the logarithm of the joint distribution can be represented as:

$$\begin{aligned}&\log p(x,y_1,y_2,...,y_N)\\ &= \log p(x)+\log p(y_1|x)+...+\log p(y_N|x) = \log p(x)+\\ &\log(\frac{\exp(f_\theta(x)[y_1])}{\Sigma_{y_1}\exp(f_\theta(x)[y_1])})+...+\log(\frac{\exp(f_\theta(x)[y_N])}{\Sigma_{y_N}\exp(f_\theta(x)[y_N])}),\end{aligned}$$

where the $f_\theta(x)[y_N]$ denotes the output of the N_{th} class. Furthermore, based on the theory of the JEM and the aforementioned formulas, the classification neural network inherently contains a corresponding energy function [26-27]. Thus, the distribution of x and the corresponding energy function can be expressed as follow:

$$p(x)=\sum_{y_1,\dots,y_N}p(x,y_1,\dots,y_N)=\prod_i\sum_{y_i}\exp(f_\theta(x)[y_i])/Z',$$

$$\mathrm{E}_{\mathrm{JEM}}(x)=-\sum_i\log\sum_{y_i}\exp(f_\theta(x)[y_i]). \tag{10}$$

In light of this, the energy of the LSTN can be expressed as:

$$\mathrm{E}_{\mathrm{LSTN}}(x)=\mathrm{E}_{\mathrm{JEM}}(x)+\mathrm{F}(x). \tag{11}$$

In this expression, $\mathrm{F}(x)$ is the energy described in Formula (6). In the task of instrument recognition, our main focus lies on the extraction of time-frequency features and the subsequent recognition processes. To simplify calculations, we can only minimize both the classification loss and the mentioned energy function $\mathrm{F}(x)$.

C. Convergence Analysis

***Lemma 1*:** After sufficient training iterations, the proposed LSTN will converge to its local optimal solution.

***Proof*:** The LSTN architecture mainly contains a TNN and an LSN. Firstly, we illustrate that independently training the TNN and the LSN leads to convergence. Subsequently, we explain that training the LSTN uniformly within the framework of JEM also achieves convergence.

Separately, the TNN aims to minimize its energy, which can be converted to optimizing its log-likelihood function [27] (or Fisher divergence). The likelihood function can reach a local optimal solution by using gradient descent [28]. Thus, if we can illustrate that the gradients of the TNN can be computed, the TNN will converge. The gradients are:

$$\begin{aligned}\frac{\partial\mathrm{F}(x)}{\partial\theta}=&-\int_c\sum_h p(h_1,h_2,c_t,c_f|x)\frac{\partial\mathrm{E}(x,h_1,h_2,c_f,c_t)}{\partial\theta}dc_t,c_f\\&+\int_{c,x}\sum_h p(h_1,h_2,c_t,c_f,x)\frac{\partial\mathrm{E}(x,h_1,h_2,c_f,c_t)}{\partial\theta}dc_t,c_f,x,\end{aligned} \tag{12}$$

where θ denotes parameters. Formula (12) can be further denoted as a summation of two expectations as Formula (13),

$$\begin{aligned}\frac{\partial\mathrm{F}(x)}{\partial\theta}=&-\mathbb{E}_{p(h_1,h_2,c_t,c_f|x)}\left[\frac{\partial\mathrm{E}(x,h_1,h_2,c_f,c_t)}{\partial\theta}\right]\\&+\mathbb{E}_{p(h_1,h_2,c_t,c_f,x)}\left[\frac{\partial\mathrm{E}(x,h_1,h_2,c_f,c_t)}{\partial\theta}\right],\end{aligned} \tag{13}$$

where $\mathbb{E}$ denotes the expectation operator, the first term in Formula (12) and (13) is an expectation of $p(h_1,h_2,c_t,c_f|x)$.

By leveraging the Gibbs sampling method, the TNN can sample the second term in Formulas (12) and (13), which represents the expectation of the joint probability $p(h_1,h_2,c_t,c_f,x)$. Through a series of state transitions guided by Formulas (2-5) and Formula (7), the Markov chain associated with Gibbs sampling converges to a stable state. This convergence ensures that TNN can be trained to attain a local optimal solution. The LSN combines a likelihood function and a discriminative loss in its objective function. The likelihood function can be optimized using Gibbs sampling, similar to TNN. Additionally, minimizing the discriminative loss in LSN is equivalent to training an MLP, which belongs to the family of neural networks [28]. MLPs have been extensively researched and are known to converge with sufficient training data. Moreover, reference [27, 29] illustrates that the likelihood function and the discriminative loss can be jointly optimized using backpropagation and gradient descent. Therefore, independently training TNN and LSN under Gibbs sampling and gradient descent leads to convergence.

As mentioned above, training the LSTN uniformly within the framework of JEM also achieves convergence. LSTN can be firstly trained as classifiers for each type of instrument as deep neural networks, and then the corresponding energy function of LSTN can be optimized by using sliced score matching method. Reference [29] illustrates that jointly optimizing the classification neural network and its corresponding energy is convergent. Moreover, for recognition tasks, we can only optimize the energy term $\mathrm{F}(x)$ for simplifying calculations. □

D. Feature Analysis

In this subsection, we analyze the extracted features. LSTN extracts local time-frequency features (h_1,h_2), long-frequency features (c_f), and long-time features (c_t). Specifically, TNN utilizes rectangular local convolutional kernels over the time-frequency representation x to extract (h_1,h_2) for modeling the local time-frequency features at different positions of x. Based on the (h_1,h_2), we have the following lemma:

***Lemma 2*:** The long time-frequency features and correlation features of the time-frequency representation in LSTN are explicitly modeled as parameterized Gaussian distributions.

***Proof*:** We analyze the features from 2 situations: learning the instrument label-specific features and detecting multiple instruments from polyphonic music.

In the learning process of the instrument label-specific features: the long-frequency features and long-term features are extracted in TNN as Formula (14) and Formula (15).

$$p(c_f|x,h_2)=\mathcal{N}\left(\beta_2^{-1}(x\otimes W_2(h_2\otimes W_2^*)+h_2\otimes W_{cf}),\Lambda_2\right), \tag{14}$$

$$p(c_t|x,h_1)=\mathcal{N}\left(\beta_1^{-1}(x\otimes W_1(h_1\otimes W_1^*)+h_1\otimes W_{ct}),\Lambda_1\right), \tag{15}$$

Lemma 2 begins the analysis with the long-frequency features. TNN learns long-frequency features by introducing the c_f factor. It uses the convolution kernel W_2 to cover the entire frequency-domain and extract global harmonic features in the frequency-domain (as term $x\otimes W_2$ in Formula(14)). On this basis, TNN combines the effects of different local features' variations in the frequency-domain by further using W_2^* and W_{cf} to cover the frequency-domain of h_2 (as term $h_2\otimes W_2^*$ and $h_2\otimes W_{cf}$ in Formula (14)), and $W_2^*\in R^{H*K}$, $W_{cf}\in R^{H*K}$, where H is the height of feature h_2. The final expression, Formula (14), incorporates the variations of these local features into global harmonic features and is parameterized as a Gaussian distribution. Formula (14) captures the long-frequency harmonic features. Moreover, based on h_2, the components of c_f are independent, reflecting the distribution changes of harmonic features in the frequency-domain at different time intervals.

Correspondingly, TNN models long-term features by introducing the c_t factor. TNN utilizes the convolution kernel W_1 to span the entire time-domain and extract global time-domain features for various frequency ranges from the input time-frequency representation x. Additionally, TNN combines the effects of different local feature variations in the time-domain by employing W_1^* and W_{ct} to cover the frequency-domain of h_1 (as term $h_1 \otimes W_1^*$ and $h_1 \otimes W_{ct}$ in Formula (15)), and $W_1^* \in R^{K*W}$, $W_{ct} \in R^{K*W}$, where W is the width of feature h_1. Formula (15) captures the long-term harmonic features as parameterized Gaussian distribution. Based on x and h_1, the components of c_t are independent, reflecting the time-domain variations across different frequency ranges.

Following TNN, the input of LSN is the extracted features, and the instrument label-specific harmonic features based on hidden units can be modeled as Formula (16):

$$p(f|y_l)=C\frac{exp(-\frac{1}{2}\alpha f^2)\sum_j\text{Softplus}(c_j+U_j+W_jf)}{\sum_j\text{Softplus}(c_j+U_j+\alpha^{-1}W_j^2)}, \quad (16)$$

where *C* is a integral coefficient. Following the above analysis, time-frequency features and instrument label-specific features can be explicitly modeled by LSTN.

In the process of detecting multiple instruments from polyphonic music: LSTN utilizes h_1 and h_2 to model the local time-frequency features, which include the time-frequency features of the superimposed parts and the local harmonic features of the remaining undistorted instruments. Based on this, c_f reflects a parameterized distribution of long-frequency features along the time-domain. It can be used to capture the onset and termination of instrument superimposition in the time-domain by utilizing parameterized distribution variations based on the conditional independence of its components. Similarly, c_t models a parameterized distribution of long-term features in the frequency-domain to capture the beginning and ending of local instrument superimposition in the frequency-domain through parameterized distribution variations according to the conditional independence of its components. Thus, the time-frequency characteristics of polyphonic music can be modeled based on the local, long-term, and long-frequency features.

Furthermore, it is important to emphasize that the constructed c_f and c_t capture the correlation within the input time-frequency representation x. Without c_f and c_t, the input time-frequency representation can be expressed as:

$$\begin{aligned}&p(x|h_1,h_2)\sim\\&\mathcal{N}(\frac{h_1\odot W_{h_1}+\beta_1^{-1}(h_1\otimes W_1^*)^2\odot W_1}{\sqrt{\alpha-\beta_1^{-1}h_1\otimes(W_1^*)^2\odot W_1^2-\beta_2^{-1}h_2\otimes(W_2^*)^2\odot W_2^2}}\\&+\frac{h_2\odot W_{h_2}+\beta_2^{-1}(h_2\otimes W_2^*)^2\odot W_2}{\sqrt{\alpha-\beta_1^{-1}h_1\otimes(W_1^*)^2\odot W_1^2-\beta_2^{-1}h_2\otimes(W_2^*)^2\odot W_2^2}},\\&(\alpha-\beta_1^{-1}h_1\otimes(W_1^*)^2\odot W_1^2-\beta_2^{-1}h_2\otimes(W_2^*)^2\odot W_2^2)). \quad (17)\end{aligned}$$

Formula (17) shows that the correlation in time-frequency representations can be modeled as covariance in a parameterized Gaussian distribution. When the covariance matrix is a positive definite matrix, Formula (17) holds, and it can be realized by amplifying α. By introducing the c_t and c_f factors, the covariance matrix can be diagonalized:

$$\begin{aligned}&p(x|h_1,h_2,c_t,c_f)\sim\\&\quad\mathcal{N}(\alpha^{-1}(h_1\odot W_{h1}+(h_1\odot W_1^*c_t)\odot W_1+\\&\quad\quad h_2\odot W_{h2}+(h_2\odot W_2^*c_f)\odot W_2),\ \Phi). \quad (18)\end{aligned}$$

thus, the constructed c_f and c_t can be used to model the correlation of the time-frequency representation. □

E. Complexity Analysis

In this section, we delve into the computational complexity of LSTN. Structurally, LSTN contains three components: TNN, *Encoder*, and LSN. It can be trained using multiple frameworks: (1) Separate training of TNN, *Encoder*, and LSN, following the greedy algorithm philosophy. (2) Joint training of TNN and *Encoder* using the JEM framework, followed by fine-tuning of instrument classification using LSN. (3) Uniform training within the JEM framework.

For situation (1): The training complexity can be divided into three parts. Assuming the basic computational complexity of the *Encoder* is denoted as $O_{Encoder}(l)$, which is similar to traditional instrument recognition neural networks. Optimizing the TNN with sliced score matching requires $2O(d_x)$ computation, where d_x is the dimensionality of input data [22]. On the other hand, if the TNN is optimized using the CD algorithm based on its free energy, it involves $O_{free}(l) = O_{CD}(l)+O_{BP}(l)$ computation. Here, l represents the number of layers and l is 2 in this paper, $O_{CD}(l)$ costs a step of Gibbs sampling with 5 probability calculations (Formulas (2-5) and Formula (7)) while $O_{BP}(l)$ refers to the complexity of calculating the gradients of free energy and the complexity of back-propagation. It is important to note that the Gaussian distribution used in Gibbs sampling has a diagonal covariance matrix. These probabilities can be easily parameterized as a single-layer neural network. Therefore, $O_{CD}(l)$ also represents the computation required for 5 times forward propagation, which is similar to a 5-layer feed-forward convolutional neural network. Optimizing the LSN solely as label-specific classifiers requires $K{*}O_{NN}(l)$ computation, where $O_{NN}(l)$ denotes the complexity of training a single hidden layer neural network classifier, and *K* is the categories of instruments. If the LSNs are jointly optimized using both classification and sliced score matching loss, they entails $K*(O_{NN}(l)+2O(d_l))$ computation. If the LSN is optimized only as classifier they entails $K{*}O_{NN}(l)$ computation. Therefore, in situation (1), the LSTN gets at most $K*[O_{NN}(l)+2O(d_l)]+O_{free}(l)$ additional computations, and at least $K*O_{NN}(l)+2O(d_x)$ additional computations compared to traditional instrument recognition neural networks. However, in this situation, the LSN in LSTN only requires less than 10 epochs of fine-tuning. Therefore, the total complexity of LSTN is acceptable. Moreover, if the $Encoder$ is trained for each type of instrument, it achieves higher complexity by adding $K*O_{Encoder}(l)$. The complexities of situations (2) and (3) are similar to situation (1). Although the TNN and the Encoder can be jointly optimized, the gradients of the sliced score function are taken with respect to x, while the gradients of the classification loss are taken with respect to the parameters. Thus, the complexities of situations (2) and (3) approximate to situation (1).

In experiments, we analyze the instrument recognition results of these 3 situations.

IV. Experiments and analysis

This section first introduces the datasets and then verifies the effectiveness of the proposed LSTN. To effectively evaluate the performance of LSTN, comparative studies are conducted in experiments, among them, some commonly used traditional instrument recognition models, deep neural network-based instrument recognition models, data augmentation and label augmentation-based instrument recognition models are used in this comparative study. Moreover, the local superposed time-frequency representations can be treated as noise, prompting the use of denoising neural networks for instrument recognition in our experiments. Experimental results are analyzed to show that: (a) LSTN achieves most of superior performance against several state-of-the-art instrument recognition algorithms; (b) LSTN effectively extracts explicit label-specific time-frequency features which can be transformed to different polyphonic music; (c) The TNN and LSN in LSTN are both effective for instrument recognition across different training frames and LSTN performs well under partial data changes.

A. *Datasets*

In the context of the predominant instrument recognition task, the detection of specific instruments from a mixture of different instruments in music is commonly required. To cover this task, in experiments, we aim to validate that the instrument label-specific features extracted by LSTN can be effectively generalized to unseen music audio for detecting the concerned instruments. Briefly, a piece of unseen music audio can be monophonic or polyphonic. To cover the two situations, we first use a monophonic dataset Melody-solos-DB to verify that the extracted instrument label-specific features can be transferred to unseen music audio for detecting the concerned instrument, then, we use a dataset IRMAS (its training data is monophonic while the testing data is polyphonic) to verify whether the proposed LSTN can learn the instrument label-specific features in the training data and sperate these features from superimposed time-frequency representations in polyphonic music of the testing data.

*1) **Medley-solos-DB Dataset:*** Medley-solos-DB is a cross-collection dataset for automatic musical instrument recognition in solo recordings [30]. It consists of a training set of 3-second audio clips, which are extracted from the MedleyDB dataset of Bittner et al. as well as a test set of 3-second clips, which are extracted from the solosDB dataset. Each of these clips contains a single instrument among a taxonomy of eight: clarinet, distorted electric guitar, female singer, flute, piano, tenor saxophone, trumpet, and violin. The Medley-solos-DB contains 21571 audio clips as WAV files, sampled at 44.1 kHz, with a single channel (mono), at a bit depth of 32. Every audio clip has a fixed duration of 2972 milliseconds, that is, 65536 discrete-time samples. After preprocessing, the test data set contains 36708 pieces of data, while the training data contains 28005.

*2) **IRMAS Dataset:*** The IRMAS dataset is intended to be used for automatic instrument recognition [14]. The training dataset contains 6705 audio files with excerpts of 3s from more than 2000 distinct recordings, and 11 pitched instruments from selected music tracks are labeled. On the other hand, the testing dataset contains 2874 audio files with lengths between 5s and 20s, and no tracks from the training data are included.

As the training data is single labeled, and the testing data is multiple labeled in polyphonic music. Using this dataset can effectively evaluate whether the proposed model is able to learn the instrument label-specific features in the training data and separate these features from superimposed time-frequency representations in the testing data. To effectively transform the trained LSTN model into testing dataset, following the method in reference [5, 15], we perform short-time analysis using overlapping windows (the same window size from the training phase) to obtain local instrument information in the audio excerpts to analyze testing audio. Moreover, it should be emphasized that, following the commonly used strategy [5, 15], a development set for parameter tuning is applied in experiments. The attributes of IRMAS are shown in Table 1.

TABLE I
INSTRUMENTS USED IN EXPERIMENTS OF IRMAS DATASET.

No.	Instruments	Abbreviations	Training	Testing
1	Cello	cel	388	111
2	Clarinet	cla	505	62
3	Flute	flu	451	163
4	Acoustic guitar	acg	637	535
5	Electric guitar	elg	760	942
6	Organ	org	682	361
7	Piano	pia	721	995
8	Saxophone	sax	626	326
9	Trumpet	tru	577	167
10	Violin	vio	580	211
11	Voice	vio	778	1024

B. *Experiments*

*1) **Evaluation:*** For a single labeled dataset Melody-solos-DB, we use the accuracy indicator for evaluation. However, as the IRMAS dataset is imbalanced and the test data is multi-labeled, the accuracy indicator is unsuitable for its evaluation. Therefore, following the evaluation method widely used in the instrument recognition task [5, 15, 16], we compute the precision, recall, and F1 scores.

*2) **Performance and analysis of LSTN:*** In this subsection, we start by comparing the LSTN algorithm with existing algorithms. We conduct experiments to obtain optimal hyper-parameter settings and analyze how the LSTN can effectively identify time-frequency features of specific instruments even when the data is partially corrupted by noise. Next, we present the process of selecting hyper-parameters and discuss the rationale behind LSTN's superior performance compared to state-of-the-art algorithms. In LSN, we utilize a Muti-task ConvNet (No.6 model in Table 3) as the $Encoder$ to map 2-dimensional feature matrices (h_1, h_2) to 1-dimensional feature vectors $(h_1^{'}, h_2^{'})$. Other neural networks are also effective for

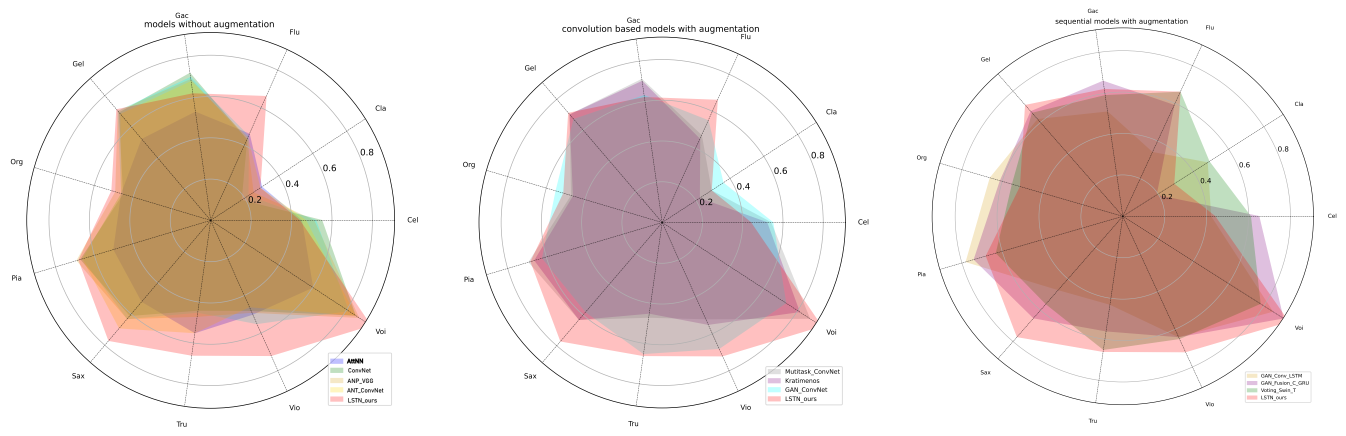


Fig. 4. The test F1 scores on each instrument compared with some commonly used instrument recognition models, and the red area covers the results of LSTN. Among them, the left sub-figure shows the results between models (without augmentation) and the LSTN, the middle sub-figure shows the results between CNN-based augmented models and the LSTN, and the right sub-figure shows the results between sequential augmented models and the LSTN.

this mapping, such as attention networks, we analyze the recognition results of different backbones in Table 6. Towards the end of this subsection, we analyze the training results of LSTN under different training frameworks. In our experiments, the best test results are indicated in bold, while suboptimal results are marked with an underline.

TABLE II
PERFORMANCE COMPARISON ON MELODY-SOLOS-DB DATASET. THE YEAR WHEN THE MODEL WAS PROPOSED IS LISTED IN PARENTHESES.

No.	Model	accuracy
1	SVM	0.44
2	RBM	0.47
3	XGBoost (2022)	0.52
4	Att-CRNN (2018) [36]	0.52
5	LSTM	0.54
6	SB-CNN (2017)	0.65
7	Openl3 (2019)	0.67
8	APNet (2021)	0.70
9	AttNN	0.73
10	Wavelet-CNN (2022)	0.73
11	ConvNet (2017	0.78
12	GAN-based CONV (2020)	0.78
13	GAN-based C-GRU (2022)	0.78
14	**LSTN (ours)**	**0.82**

Testing performance comparison on the Melody-solos-DB dataset is shown in Table 2. Table 2 shows that the proposed LSTN achieves the best test accuracy compared with other commonly used instrument recognition models. In this experiment, we list 13 commonly used models for comparison, No.1-3 models use traditional machine learning methods to manually extract features from the music audio [31], we should point out that different from reference [32], 3 and 8 models run on actual test data in this experiment. No.4-11 models use neural networks to build end-to-end instrument recognition approaches, in which No.6-8 models are commonly used for Melody-solos-DB (a convolutional neural network (SB-CNN) [33], an auto-encoder based audio prototype network (APNet) [32], a pre-trained embedding extractor model based on self-supervised learning (Openl3) [34]) and the results show that these neural networks are more effective compared with traditional models of No.1-3, No.9 model designs self-attention layer for music audio to improve the recognition ability, No.10 model decomposes music audio into time-frequency features by using Wavelet analysis to get higher recognition accuracy [35], and No.12-13 models use data augmentation to generate more effective neural networks. These illustrate that these augmentations are effective for instrument recognition. This experiment verifies that LSTN can be effectively generalized to unseen monophonic music audio for recognizing instruments.

To evaluate the ability of LSTN in detecting instruments with partially corrupted signals, simulating scenarios where the instrument signal undergoes partial data changes due to superposition, we introduce noise (masking noise and Gaussian noise) to the Melody-solos-DB dataset and measure the instrument recognition accuracy. To comprehensively analyze noise issues, the noise introduced in experiments includes both globally corrupted noise and partially corrupted noise. For the noise, we apply an introduction ratio of 0.2 to the dataset. For globally corrupted noise, the introduced variance of Gaussian noise is the 10% of the average variance of the current corrupted data. Regarding masking noise, the current corrupted data is masked at a ratio of 20%. Partially corrupted noise is introduced to time-frequency representations with a probability of 10% for each window. The size of each window randomly ranges from [10, 1] to [15, 5]. The test accuracies of LSTN, averaged over 10 trials, are presented in Fig 5. As depicted in Fig 5, both globally corrupted noise and partially corrupted noise affect all models, but LSTN outperforms other commonly used instrument recognition models under partially corrupted noise, and these results to some extent maintain consistency with the original intention of the model design.

Next, we valuate the proposed LSTN model on a more challenging IRMAS dataset. As this dataset is imbalanced, we first evaluate LSTN on each instrument class in Table 3 and then compare $F1_{macro}$ and $F1_{micro}$ scores of the whole

TABLE III
DETAILED PERFORMANCE COMPARISON ON IRMAS DATASET. THE YEAR WHEN THE MODEL WAS PROPOSED IS LISTED IN PARENTHESES.

No.	model	Cel	Cla	Flu	Gac	Gel	Org	Pia	Sax	Tru	Vio	Voi	F1
1	SVM	0.25	0.16	0.21	0.46	0.35	0.27	0.38	0.22	0.20	0.20	0.31	0.27
2	RBM	0.22	0.16	0.20	0.43	0.36	0.25	0.32	0.22	0.19	0.18	0.28	0.26
3	MTF-DNN (2018)	0.15	0.26	0.27	0.43	0.36	0.28	0.36	0.28	0.18	0.22	0.32	0.28
4	AttNN	0.45	0.30	0.46	0.53	0.52	0.46	0.50	0.52	0.55	0.50	0.60	0.49
5	ConvNet (2017)	0.55	0.18	0.43	**0.72**	0.69	0.45	0.67	0.61	0.44	0.48	0.85	0.55
6	Muti-task ConvNet(2020)	0.54	0.29	0.47	0.71	0.70	0.47	0.69	0.63	0.47	0.52	0.87	0.58
7	Kratimenos et al.(2021) [39]	0.52	0.22	0.45	0.70	0.70	0.46	0.66	0.63	0.45	0.55	0.81	0.56
8	GAN-ConvNet (2021)	0.55	0.36	0.55	0.63	0.67	0.55	0.62	0.58	0.65	0.68	0.73	0.60
9	GAN-Conv-LSTM (2021)	0.42	0.48	0.34	0.51	0.62	**0.66**	**0.78**	0.47	0.43	0.64	0.85	0.56
10	GAN-Fusion-C-GRU (2022)	**0.65**	0.19	0.59	0.66	0.67	0.61	0.74	0.65	0.56	0.64	0.91	0.62
11	Voting-Swin-T (2022)	0.61	**0.49**	**0.66**	0.59	0.66	0.55	0.63	0.56	0.65	0.65	0.78	0.62
12	ANP-VGG (2021) [38]	0.51	0.25	0.43	0.69	0.70	0.49	0.69	0.63	0.47	0.49	0.87	0.56
13	ANT-ConvNet (2020) [8]	0.47	0.29	0.44	0.68	0.68	0.46	0.66	0.69	0.55	0.46	0.85	0.57
14	**LSTN (ours)**	0.44	0.29	**0.66**	0.62	**0.71**	0.51	0.68	**0.77**	**0.66**	**0.72**	**0.94**	**0.64**

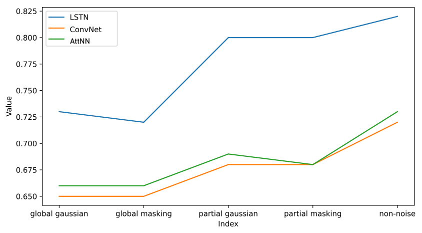


Fig. 5. The test accuracies of LSTN under noise with 10 averages.

classes in Table 4. For comparison, we list 13 machine learning models in Table 3 and 16 models in Table 4, in which No.1-2 models in Table 3 and No.1-3 models in Table 4 are traditional machine learning models. No.3-5 models in Table 3 and No.4-7 models in Table 4 use neural networks to build end-to-end instrument recognition approaches, among them, LSTM and Attention Neural Networks (AttNN) are designed based sequential representations, while MTF-DNN [37] and ConvNet [5] are built on time-frequency spectrograms. To provide more expressive data and labels for deep neural networks, scholars introduce additional methods to construct more effective instrument recognition models. No.6-11 models in Table 3 and No.8-14 models in Table 4 introduce data augmentation or label augmentation methods to deep neural networks, among them, Muti-task ConvNet [15] is a label augmentation based convolutional neural network, and the other data augmentation based deep neural networks apply WaveGAN to generate fake time-frequency representations and construct corresponding CNN (GAN-based ConvNet)[18], CNN-LSTM (GAN-based Conv-LSTM)[18], CNN-GRU (GAN-based Fusion-C-GRU) [18], or attention (Voting-Swin-T) neural networks [17] for instrument recognition. No.12-13 models in Table 3 (No. 15-16 models in Table 4) introduce adversarial noise learning methods to the deep neural networks to improve the their robustness [8, 38].

First, we show the test F1 scores on each type of instrument compared with some typical temporal instrument recognition models and CNN-based instrument recognition models in Fig. 4. In Fig. 4, the models are categorized into three groups. The left sub-figure illustrates that the proposed LSTN outperforms the models without augmentation methods (including noise learning models), demonstrating superior test results for most instrument types. The middle sub-figure compares the proposed LSTN with convolution-based models employing data or label augmentation, indicating its consistently exceptional performance. The right sub-figure showcases various state-of-the-art instrument recognition models incorporating data augmentation or (and) label augmentation techniques into sequential models. The results indicate that LSTN achieves superior performance on the majority of instrument types. Next, we list the entire experimental results compared with commonly used instrument recognition models. In Table 3, ConvNet is specially designed for instrument recognition neural network, and the results show that it performs better than the other neural networks and traditional models of No.1-5. This means the specially designed generalization strategy of ConvNet for the testing data in IRMAS dataset is effective. No.6 model introduces a multi-task learning method to the ConvNet by using an auxiliary classification designed on additional labels of the onset groups and instrument families. As Table 3 shows, this Muti-task ConvNet achieves better $F1$ scores compared with other methods in No.1-6 by generating more expressive labels. No.7-11 models aim to generate fake audio to provide more expressive data for deep neural networks, and they achieve better results than general neural networks. Results compared with the No.12-13 models verify that LSTN performs better than commonly used deep noise learning neural networks. The improved results of LSTN mean that the extracted instrument label-specific features can be effectively applied to recognize superimposed instruments in polyphonic music of testing data. Moreover, we also list the resulting $F1_{macro}$ and $F1_{micro}$ scores of LSTN for the whole classes in Table 4. As it shows, LSTN achieves the best $F1_{macro}$ score and $F1_{micro}$ score compared with the other 14 machine learning models, among them, No.11-14 models in Table 4 use general data augmentation method for generating more balanced training data to obtain better adaptability.

TABLE IV
PERFORMANCE COMPARISON ON IRMAS DATASET. THE YEAR WHEN THE MODEL WAS PROPOSED IS LISTED IN PARENTHESES.

No.	Model	F1 macro	F1 micro
1	RBM	0.26	0.32
2	SVM	0.27	0.36
3	Bosch et al. [14]	0.43	0.50
4	LSTM	0.44	0.51
5	MTF-DNN (2018)	0.28	0.32
6	AttNN	0.49	0.55
7	ConvNet (2017)	0.52	0.62
8	Muti-task ConvNet (2020)	0.58	0.68
9	Kratimenos et al. (2021) [39]	0.55	0.65
10	Pons et al. (2017) [40]	0.52	0.65
11	GAN-based ConvNet (2021)	0.60	0.65
12	GAN-based Conv-LSTM (2021)	0.56	0.65
13	GAN-based Fusion-C-GRU (2022)	<u>0.62</u>	<u>0.69</u>
14	Voting-Swin-T (2022)	<u>0.62</u>	0.66
15	ANP-VGG (2021) [38]	0.56	0.62
16	ANT-ConvNet (2020) [8]	0.57	0.62
17	**LSTN (ours)**	**0.64**	**0.71**

In the above experiments, we validate that the proposed LSTN can effectively generalize the extracted time-frequency features to unseen monophonic and polyphonic music audio for instrument recognition. Next, we show the selection process of hyper-parameters. Hyper-parameters in LSTN contain two parts, TNN hyper-parameters and LSN hyper-parameters. First, we select hyper-parameters of TNN from its candidate hyper-parameters. Then, we fix the optimal hyper-parameters of TNN and select hyper-parameters of LSN. The optimal hyper-parameters of LSTN are shown in Table 5.

TABLE V
HYPER-PARAMETERS OF LSTN.

Model	batch size	kernel-h	kernel-w	lr	hidden-channel
TNN	64	3	3	2e-6	52
Model	batch size	auxiliary	window-size	lr	-
LSN	64	3	2, 4, 6	1e-3	-

In Table 5, kernel-h and kernel-w are the height and width of traditional convolution kernels, lr is the initial value of the learning rate for Adam optimizer, hidden-channel is the channel of time-frequency features in TNN, and auxiliary is the number of classes in the auxiliary classifier (based on Onset types). For the test dataset, LSTN uses the same max-pooling strategy with Muti-task ConvNet, and window-size is the size of the max-pooling window, moreover, the theta is selected from [0, 0.1, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6], it should be noted that both method 1 and method 2 in the max-pooling strategy are effective on IRMAS dataset. Some comparisons of models with different hyper-parameters are shown in ablation experiments.

Results in Table 4 verify that for most instruments, LSTN performs better than the other commonly used models. We consider that these improvements mainly come from specially extracted time-frequency features and the constructed label-specific features. To valuate the contribution of these two features in the recognition process, we design an ablation experiment on the IRMAS dataset. It should be emphasized that although the test dataset is different from the training dataset of IRMAS, the data distributions of the two sub datasets are consistent. However, test data may come from classes that are unknown during training [41, 42]. In future work, we will investigate instrument recognition in such scenarios.

*3) **Ablation study for LSTN:*** Next, we verify that the time-frequency features extracted from TNN and instrument label-specific features extracted from LSN in LSTN are both effective for instrument recognition in this ablation experiment. Results are shown in Table 6.

TABLE VI
ABLATION EXPERIMENT FOR LSTN.

No.	Model	F1 macro	F1 micro
1	RBM	0.26	0.32
2	LSN Classifier	0.32	0.39
3	<u>backbone</u>	<u>0.58</u>	<u>0.68</u>
4	<u>TNN_3+backbone</u>	<u>0.60</u>	<u>0.68</u>
5	<u>TNN_4+backbone</u>	<u>0.60</u>	<u>0.67</u>
6	<u>TNN_5+backbone</u>	<u>0.58</u>	<u>0.68</u>
7	*TNN+Flatten+LSN*	*0.41*	*0.46*
8	*TNN+ConvDBN+LSN*	*0.51*	*0.52*
9	*TNN+ResNet50+LSN*	*0.60*	*0.68*
10	*TNN+ResNet101+LSN*	*0.60*	*0.68*
11	*TNN+ConvAtt+LSN*	*0.63*	*0.70*
12	*backbone+LSN*	*0.61*	*0.70*
13	TNN+ConvNet+LSN-Single	0.64	0.70
14	TNN_without_c+backbone+LSN	0.63	0.70
15	TNN+backbone+LSN(LSTN)	0.64	0.71
16	TNN+backbone+VGGish+LSN	0.64	0.71

In Table 6, models indicated by underlined names represent recognition neural networks with convolution kernels of varying sizes (from 0 to 5, and No.3 model can be regarded as a model without kernels in TNN). Models labeled by italic names are variants with different backbone architectures (we conduct these experiments to demonstrate the compatibility of LSTN with different backbone models). As Table 6 shows, the classifier of LSN outperforms RBM in terms of $F1_{macro}$ and $F1_{micro}$. This outcome indicates that the classifier of LSN extracts more effective features for instrument recognition compared to traditional RBM. In our previous experiments, the backbone neural network is Muti-task ConvNet. TNN+backbone refers to a Muti-task ConvNet that incorporates time-frequency features extracted from TNN, while backbone+LSN combines Muti-task ConvNet with the LSN classifier. As demonstrated by this ablation experiment, backbone+LSN achieves better results on $F1_{macro}$ compared

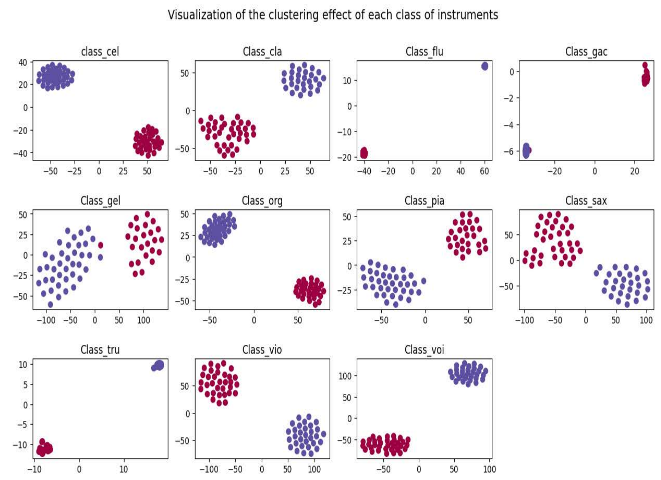


Fig. 6. Visualization of the label-specific feature clusters for each class.

to backbone alone. Similarly, TNN+backbone also shows an improvement in $F1_{macro}$ compared to backbone alone. Finally, TNN+backbone+LSN (LSTN) achieves the best F1 scores, confirming the effectiveness of both time-frequency features and label-specific features. In this ablation experiment, we also test the effect of different convolution kernels, as shown in Table 6, the size of convolution kernels in TNN has a slight impact on the recognition results, we choose convolution kernels in the commonly used size of 3 in LSTN. Moreover, we also evaluate the instrument recognition ability of LSTN using different backbone neural networks. Among them, TNN+Flatten+LSN is an LSTN without a backbone, as it only has a single hidden layer. TNN+ConvDBN+LSN employs a three-layer Convolutional Deep Belief Net as the backbone, while TNN+ConvNet+LSN-Single is based on a ConvNet without multi-task learning. TNN+ResNet+LSN uses ResNet as the backbone and TNN+ConvAtt+LSN uses a convolutional attention neural network as the backbone, these models are used to test the effectiveness and compatibility of different neural networks for LSTN. The experimental results demonstrate that the extracted time-frequency features and the label-specific recognition approach are useful for different backbones, but the VGG net obtains the best recognition results. The important point to emphasize is that the performance of the model in the predominant instrument recognition task does not necessarily improve as the neural network becomes deeper. Using ResNet_50 and ResNet_101 as backbones cannot bring better recognition results. Furthermore, TNN_without_c+backbone+LSN utilizes only the extracted time-frequency features (h_1, h_2), excluding the time-domain feature c_t and frequency-domain feature c_f. The experiment comparing LSTN and TNN_without_c+backbone+LSN confirms the effectiveness of the specially designed time-domain feature c_t and frequency-domain feature c_f for instrument recognition. Additionally, pre-trained neural networks based on large audio datasets always be helpful for downstream recognition tasks, such as VGGish for audio retrieval [43], but VGGish cannot provide effective improvement for predominant instrument recognition task as TNN+backbone+VGGish+LSN model shows in Table 6.

Next, we discuss the training effectiveness of LSTN under different training frameworks. As mentioned above, the TNN in LSTN can be trained by contrast divergence or score matching, and the LSN can be trained as a joint energy-based model for each type of instrument or a contrast divergence-based model with several classifiers for different instruments. Specifically, we denote the LSTN with a score matching trained TNN as LSTN_scoreTNN, the LSTN with LSNs trained as joint energy-based models as LSTN_JEMLSN, and the LSTN trained as a joint energy-based model as LSTN_JEM. Table 7 displays the F1 scores obtained by testing these models.

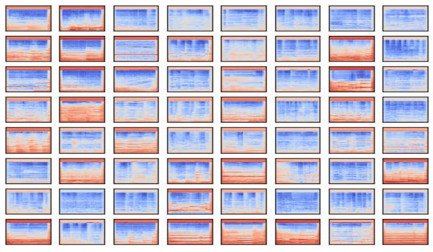
TABLE VII
ABLATION EXPERIMENT FOR LSTN WITH DIFFERENT TRAINING METHODS.

No.	Model	F1 macro	F1 micro
1	LSTN_scoreTNN	0.64	0.71
2	LSTN_JEMLSN	0.64	0.70
3	LSTN_JEM	0.63	0.69
4	LSTN	0.64	0.71

As indicated in Table 7, the LSTN models trained using different frameworks yield comparable test F1 scores. Notably, the LSTN model trained with $Encoder$ and multiple classifiers constructed using contrast divergence for different instruments exhibits slightly superior performance compared to other models, which employ distinct deep neural networks for each instrument type. These results suggest that both the score matching method and contrast divergence demonstrate similar training effectiveness.

LSTN has the characteristic of explicitly modeling time-domain features and frequency-domain features. In the next section, we further analyze these explicit features.

*4) **Analysis of explicit features in LSTN**:* The purpose of this section is to visually demonstrate the consistency between the extracted explicit features and their intended purpose. Here, we utilize the validation dataset for visualization. We analyze the separation effect of label-specific features using the t-distributed stochastic neighbor embedding (t-SNE) algorithm [44]. The parameter n-components of t-SNE is set to 2 in order to project the output feature maps from network layers into two dimensions, and PCA is employed for initialization. Fig.6 displays the results of a mini-batch, illustrating LSTN's ability to distinguish label-specific features from other features. As an energy-based model, effective features play a crucial role in both data reconstruction and recognition tasks [32]. Subsequently, we demonstrate that the extracted time-frequency features can be used to reconstruct time-frequency representations, as depicted in Fig.7. Furthermore, we visualize the extracted time-domain feature c_t and frequency-domain feature c_f in Fig.8. From the figures, it can be observed that the time-domain features c_f in the mini-batch exhibit a higher activation frequency, indicating their importance in capturing temporal information. On the other hand, the activations of frequency-domain features c_t depend more on their corresponding input representations.


(a) Real_input

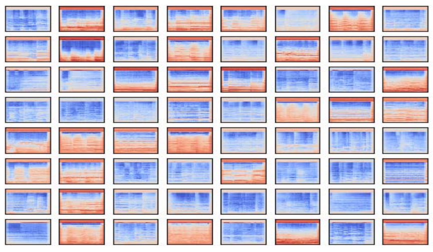

(b) Reconstruction

Fig. 7. The input time-frequency representations and their reconstruction.

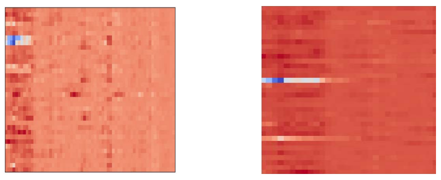

(a) c_t (b) c_f

Fig. 8. The extracted long-term feature c_t and long-frequency feature c_f.

V. Conclusion

This paper proposes a deep energy-based neural network named LSTN for instrument recognition. Experimental results verify that the instrument label-specific features extracted by LSTN can be effectively generalized to unseen monophonic music and polyphonic music for instrument recognition. Furthermore, for the monophonic Melody-solos-DB dataset and polyphonic IRMAS dataset, LSTN achieves comparable or better recognition results compared with other commonly used instrument recognition models. While LSTN explores a predominant instrument recognition method based on label-specific features, there are still some related aspects that require further strengthening. This paper primarily detects the instruments of interest from the perspective of label specific features. However, there exists correlation between the data and the labels, as well as correlation among the label-specific features. Exploring how to further model the label correlation and the correlation among label-specific features to improve instrument recognition results is a direction we intend to investigate in the future. Additionally, the differences in data quality may have a significant impact on the effectiveness of modeling instrument features. Although there are relevant methods that aim to improve instrument recognition by generating samples, these methods mainly focus on addressing intra-class and inter-class imbalance issues. However, samples generated by such methods still exhibit imbalances in terms of data quality, thereby limiting the improvements they provide. Therefore, a focus of our future work will be on developing suitable data augmentation methods that address both the imbalance in data quality and the imbalance in sample quantity.

References

[1] J. Wu, C. Hu, Y. Wang, et al. *A hierarchical recurrent neural network for symbolic melody generation*. IEEE Trans. Cybernetics, vol. 50, no. 6, pp. 2749C2757, 2019.

[2] Y. Wang, K. Han, D. Wang, *Exploring monaural features for classification-based speech segregation*, IEEE Trans. Audio, Speech, Lang. Process., vol. 21, no. 2, pp. 270C279, Feb. 2013

[3] S. Essid, G. Richard, B. David. *Musical instrument recognition by pairwise classification strategies*. IEEE Trans. Audio, Speech, and Language Processing, vol. 14, no.4, pp. 1401-1412, 2006.

[4] S. Rajesh, N.-J. Nalini. *Recognition of musical instrument using deep learning techniques*. International Journal of Information Retrieval Research, vol. 11, no.4, pp. 41-60, 2021.

[5] Y. Han, J. Kim, and K. Lee, *Deep convolutional neural networks for predominant instrument recognition in polyphonic music*, IEEE/ACMTrans. Audio, Speech Lang. Process., vol. 25, no. 1, pp. 208C221, May 2016.

[6] N. Narodytska, S.-P. Kasiviswanathan, *Simple Black-Box Adversarial Attacks on Deep Neural Networks*. In CVPR Workshops, Vol. 2, pp. 2, 2017.

[7] N. Levy, G. Katz, *Roma: a method for neural network robustness measurement and assessment*. International Conference on Neural Information Processing, Singapore, pp.92C105, 2022.

[8] E. Rusak, L. Schott, R. Zimmermann, et al. *A simple way to make neural networks robust against diverse image corruptions*. In Proc. Int. 16th European Conference, Glasgow, pp. 23C28, 2020.

[9] J. Engel, C. Resnick, A. Roberts, et al. *Neural audio synthesis of musical notes with wavenet autoencoders*. International Conference on Machine Learning, pp. 1068C1077, 2017.

[10] W. Qian, J. Huang, F. Xu, et al. *A survey on multi-label feature selection from perspectives of label fusion*. Information Fusion, no. 100, pp. 101948, 2023.

[11] M.-L. Zhang, L. Wu, *LIFT: Multi-Label Learning with Label-Specific Features*, IEEE Trans. Pattern Analysis, Machine Intelligence. Process., vol. 37, no. 1, pp. 107C120, 2015.

[12] X. Wang, F. Wu, Y.-H.-L. Li, *Listening training and subjective evaluation of sound quality (translated)*, Communication University of China Press, 2016.

[13] Y. Du, C. Durkan, R. Strudel, et al. *Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc*. in Proc. Int. machine learning Conf., pp. 8489C8510, 2023.

[14] J. Bosch, J. Janer, F. Fuhrmann, et al., *A comparison of sound segregation techniques for predominant instrument recognition in musical audio signals*, in Proc. Int. Symp./Conf. Music Inf. Retrieval, pp. 559C564, 2012.

[15] D. Yu, H. Duan, J. Fang, et al., *Predominant Instrument Recognition Based on Deep Neural Network With Auxiliary Classification*, IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 28, no.2, pp.852-861, 2020.

[16] B. Kilambi, A. Parankusham, S. Tadepalli, *Instrument Recognition in Polyphonic Music Using Convolutional Recurrent Neural Networks*, in Proc. Int. Intelligent Computing, Information,Control Systems Conf., pp. 449C460, 2021.

[17] L.-C. Reghunath, R. Rajan, *Transformer-based ensemble method for multiple predominant instruments recognition in polyphonic music*. EURASIP Journal on Audio, Speech, and Music Processing, vol. 1, pp. 11, 2022.

[18] C.-R. Lekshmi, R. Rajeev, *Multiple Predominant Instruments Recognition in Polyphonic Music Using Spectro/Modgd-gram Fusion*, Circuits, Systems, and Signal Processing, pp. 1C21, 2023.

[19] M.-L. Zhang, Y.-K. Li, H. Yang, et al., *Towards Class-Imbalance Aware Multi-Label Learning*. IEEE Trans. Cybernetics, Process., vol. 99, pp. 1C13, 2020.

[20] S. Gururani, M. Sharma, A. Lerch, *An Attention Mechanism for Musical Instrument Recognition*, arXiv preprint arXiv:1907.04294, 2019.

[21] P. Hamel, D. Eck, *Learning Features from Music Audio with Deep Belief Networks*, in Proc. Int. Society for Music Information Retrieval Conf., pp. 9C13, 2010.

[22] Y. Song, D. Kingma. *How to train your energy-based models*. arXiv preprint arXiv:2101.03288, 2021.

[23] A. Courville, G. Desjardins, J. Bergstra, et al., *The spike-and-slab RBM and extensions to discrete and sparse data distributions*, IEEE trans. pattern analysis, machine intelligence, vol.36, no. 9, pp. 1874C1887, 2013.
[24] J.-Y. Hang, M.-L. Zhang. *Collaborative learning of label semantics and deep label-specific features for multi-label classification*. IEEE trans. pattern analysis, machine intelligence, 2021, 44(12): 9860-9871.
[25] J. Ma, H. Zhang, T.-W.-S. Chow. *Multilabel classification with label-specific features and classifiers: A coarse-and fine-tuned framework*. IEEE trans. cybernetics, vol. 51, no.2, pp. 1028-1042, 2019.
[26] W. Grathwohl, K.-C. Wang, J.-H. Jacobsen, et al. *Your classifier is secretly an energy based model and you should treat it like one*, in Proc. Int. Learning Representations Conf, 2019.
[27] Y. Liu, Y. Wang, L. Deng, et al., *Visual Images Memory Recall Based on ClassRBM and Free Energy Minimization*, in Proc. IEEE Int. Signal Processing Conf., vol. 1, pp. 262C267, 2020.
[28] A.-N. Gorban, D. Ii, *The general approximation theorem*, in Proc. IEEE, World Congress, Neural Networks, Joint Conf., vol.2, pp. 1271C1274, 1998.
[29] H. Larochelle, Y. Bengio. *Classification using discriminative restricted Boltzmann machines*. in Proc. 25th international conference on Machine learning, pp. 536C543, 2008.
[30] R. Bittner, J. Salamon, M. Tierney, et al., *MedleyDB: A Multitrack Dataset for Annotation-Intensive MIR Research*. in Proc. International Society for Music Information Retrieval Conf., pp. 27C31, October 2014.
[31] C. Joder, S. Essid, G. Richard, *Temporal Integration for Audio Classification With Application to Musical Instrument Classification*, IEEE Trans. Audio Speech Lang. Process. vol. 17, pp. 174C186, 2009.
[32] P. Zinemanas, M. Rocamora, M. Miron, et al., *An Interpretable Deep Learning Model for Automatic Sound Classification*, Electronics, vol. 10, no. 7, pp. 850, 2021.
[33] J. Salamon, J.-P. Bello, *Deep Convolutional Neural Networks and Data Augmentation For Environmental Sound Classification*, IEEE Signal Process. Lett., vol. 24, pp. 279C283, 2017.
[34] J. Cramer, H.-H. Wu, J. Salamon, et al., *Look, Listen and Learn More: Design Choices for Deep Audio Embeddings*, in Proc. IEEE Acoustics, Speech, Signal Processing Conf., pp. 3852C3856, May 2019.
[35] H. Hacihabiboglu, N. Canagarajah, *Musical instrument recognition with wavelet envelopes*, in Proc. Forum Acusticum, 2002.
[36] D.-C. De Andrade, S. Leo, M.-L.-D.-S. Viana, et al., *A neural attention model for speech command recognition*. arxiv preprint arxiv:1808.08929, 2018.
[37] K. Racharla, V. Kumar, C.-B. Jayant, et al., *Predominant musical instrument classification based on spectral features*. in Proc. Signal Processing and Integrated Networks (SPIN), pp. 617-622, 2020.
[38] A. Liu, X. Liu, H. Yu, et al. *Training robust deep neural networks via adversarial noise propagation*. IEEE Trans. Image Processing, vol. 30, pp. 5769C5781, 2021.
[39] A. Kratimenos, K. Avramidis, C. Garoufis, et al., *Augmentation methods on monophonic audio for instrument classification in polyphonic music*, in Proc. European Signal Processing Conf., pp. 156C160, 2021.
[40] J. Pons, O. Slizovskaia, R. Gong, *Timbre analysis of music audio signals with convolutional neural networks*, in Proc. European Signal Processing Conf., pp. 2744C2748, 2017.
[41] Z. Fang, Y. Li, J. Lu, et al. Is out-of-distribution detection learnable?. in Proc. Advances in Neural Information Processing Systems, vol. 35, pp. 37199C37213, 2022.
[42] R. Dai, Y. Zhang, Z. Fang, et al. Moderately Distributional Exploration for Domain Generalization. arXiv preprint arXiv:2304.13976, 2023.
[43] J.-F. Gemmeke, D.-P.-W. Ellis, D. Freedman, et al. Audio set: An ontology and human-labeled dataset for audio events, in Proc. acoustics, speech and signal processing Conf. pp. 776-780, 2017.
[44] L. van der. Matten, G. Hinton, *Visualizing high-dimensional data using t-SNE*, J. Mach. Learn. Res., vol. 9, pp. 2579C2605, 2008.