INSTRUMENT IDENTIFICATION IN POLYPHONIC MUSIC: FEATURE WEIGHTING WITH MIXED SOUNDS, PITCH-DEPENDENT TIMBRE MODELING, AND USE OF MUSICAL CONTEXT

Tetsuro Kitahara,[†] Masataka Goto,[‡] Kazunori Komatani,[†] Tetsuya Ogata[†] and Hiroshi G. Okuno[†]

[†]Dept. of Intelligence Science and Technology
Graduate School of Informatics, Kyoto University
Sakyo-ku, Kyoto 606-8501, Japan
{kitahara, komatani, ogata, okuno}@kuis.kyoto-u.ac.jp

[‡]National Institute of Advanced Industrial
Science and Technology (AIST)
Tsukuba, Ibaraki 305-8568, Japan
m.goto@aist.go.jp

ABSTRACT

This paper addresses the problem of identifying musical instruments in polyphonic music. Musical instrument identification (MII) is an improtant task in music information retrieval because MII results make it possible to automatically retrieving certain types of music (*e.g.*, piano sonata, string quartet). Only a few studies, however, have dealt with MII in polyphonic music. In MII in polyphonic music, there are three issues: feature variations caused by sound mixtures, the pitch dependency of timbres, and the use of musical context. For the first issue, templates of feature vectors representing timbres are extracted from not only isolated sounds but also sound mixtures. Because some features are not robust in the mixtures, features are weighted according to their robustness by using linear discriminant analysis. For the second issue, we use an *F0-dependent multivariate normal distribution*, which approximates the pitch dependency as a function of fundamental frequency. For the third issue, when the instrument of each note is identified, the a priori probablity of the note is calculated from the a posteriori probabilities of temporally neighboring notes. Experimental results showed that recognition rates were improved from 60.8% to 85.8% for trio music and from 65.5% to 91.1% for duo music.

Keywords: Musical instrument identification, mixed-sound template, F0-dependent multivariate normal distribution, musical context, MPEG-7

1 INTRODUCTION

The increasing quantity of musical audio signals available in electric music distribution services and personal music storage has made users spend a longer time on finding musical pieces that they want. Efficient music information retrieval (MIR) technologies are indispensable to shorten the time to find musical pieces. In particular, automatic description of musical content in a universal framework is expected to become one of the most important key technologies for achieving sophisticated MIR. In fact, the ISO recently established a new standard called MPEG-7 [1], which provides a universal framework for describing multimedia content.

The names of musical instruments play an important role as music descriptors because musical pieces are sometimes characterized by what instruments are used. In fact, the names of some music genres are based on instrument names, such as "piano sonata" and "string quartet." In addition, when a user wants to search for certain types of musical pieces, such as piano solos or string quartets, a retrieval system can use the description of musical instrument names. Therefore, musical instrument identification (MII), which aims at determining what instruments are used in musical pieces, has been studied in recent years [2, 3, 4, 5, 6, 7, 8].

Identifying instruments in polyphonic music is more difficult than in monophonic music. In fact, most methods of identifying monophonic sounds [3, 4, 7, 9] often fail in dealing with polyphonic music. For example, our previous method [9], which identified an instrument by calculating the similarities between a feature vector of a given isolated sound and prestored feature vectors of instrument-labeled sounds (called *training data*), had difficulty dealing with polyphonic music because features extracted from simultaneously played instruments were different from those extracted from monophonic sounds.

To achieve highly accurate MII in polyphonic music, it is essential to resolve three issues: feature variations caused by sound mixtures, the pitch dependency of timbres, and the use of musical context. These issues, however, have not been fully dealt with in existing studies. Some techniques such as time-domain waveform template matching [5], feature adaptation [6] and the missing feature theory [2] have been proposed to address the first issue, but no attempts have been made to construct a template from polyphonic music although this is expected to contribute to improving MII. To address the second issue, most existing studies have used multiple templates covering the entire pitch range for each instrument, but they have not dealt with effective modeling of the pitch dependency of timbres. To address the third issue, Kashino *et al.* [5] introduced music stream networks and proposed a technique of propagating the a posteriori probabilities of musical notes in a network to one another based on the Bayesian network. To apply musical context to identification frameworks not based on the Bayesian network, however, we need an alternative solution.

In this paper, to address the first issue, we construct a *feature vector template* (*i.e.*, a set of training data) from polyphonic sound mixtures. Because features tend to vary

in similar ways during both training and identification, this method can improve MII. Furthermore, because the robustness of features to the mixtures can be analyzed through their variances in the mixture-based templates, features are weighted according to their robustness by using linear discriminant analysis. To address the second issue, we use a pitch-dependent timbre model, called *F0-dependent multivariate normal distribution*, which we proposed in our previous paper [9] for isolated sounds. This model represents the pitch dependency of each feature of instrument timbres as a function of fundamental frequency (F0) and is expected to remain effective for polyphonic mixtures. To address the third issue, we calculate the a priori probability of each note from the a posteriori probabilities of its temporally neighboring notes. This method aims at avoiding musically unnatural errors by considering temporal continuity of melodies; for example, if the identified instrument names of a successive note sequence are all "flute" except for one "clarinet," this exception can be considered an error and corrected.

The rest of this paper is organized as follows: Section 2 discusses the three issues involved in applying MII to polyphonic music. Sections 3, 4 and 5 describe our solutions to the three issues. Section 6 reports the results of our experiments and Section 7 concludes the paper.

2 INSTRUMENT IDENTIFICATION IN POLYPHONIC MUSIC

The aim of our study is to identify, when an audio signal of polyphonic music is given, which musical instruments are being used to play this musical piece. In a typical framework, an MII system first detects musical notes and then identifies the name of the instrument for each note, because the musical audio signals handled here contain many musical notes, which are often simultaneously played. Once a musical note is detected, a feature vector $\boldsymbol{x}$ concerning the note is extracted. Then, the a posteriori probability given by $p(\omega_i|\boldsymbol{x}) = p(\boldsymbol{x}|\omega_i)p(\omega_i)/p(\boldsymbol{x})$ is calculated, where ω_i denotes an instrument ID, $p(\boldsymbol{x}|\omega_i)$ and $p(\omega_i)$ are a probability density function (PDF) and the a priori probability of the instrument ω_i. Finally, the instrument maximizing $p(\omega_i|\boldsymbol{x})$ is determined as an MII result.

As previously mentioned, dealing with polyphonic music is much more difficult than with monophonic music. The main issues in dealing with polyphonic music can be summarized as follows:

Issue 1 *Feature variations caused by sound mixtures*
The main difficulty of dealing with polyphonic music lies in the fact that it is impossible to extract acoustical features of each instrument without blurring because of the overlapping of frequency components. If a clear sound for each instrument could be obtained with sound separation technology, the identification of polyphonic music might result in identifying monophonic sounds. In practice, however, it is very difficult to separate a mixture of sounds without distortion occuring. To achieve highly accurate identification, it is necessary to deal with feature variations caused by the overlapping of frequency components.

Issue 2 *Pitch dependency of timbres*
The pitch dependency of timbres also makes MII difficult. In contrast to other sound sources including human voices, musical instruments have wide pitch ranges. For example, the pitch range of pianos covers over seven octaves. Such a wide pitch range makes timbres quite different from pitch to pitch. It is therefore necessary to deal with this pitch dependency to attain accurate MII.

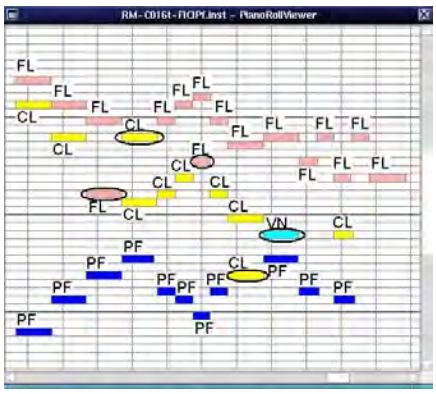

Figure 1: Example of musically unnatural errors. This example is an excerpt from results of identifying each note individually in a piece of trio music. The marked notes are musically unnatural errors, which can be avoided by using musical context. PF, VN, CL and FL represent piano, violin, clarinet and flute.

Issue 3 *Musical context*
When identifying the instrument playing a musical note, a system should take identification results of temporally neighboring notes into consideration due to the time continuity of melodies. Individually identifying the instrument of each note sometimes causes musically unnatural errors as can be seen in Figure 1 (*e.g.*, only one clarinet note in a melody played on a flute). To avoid such musically unnatural errors, it is important to exploit musical context.

We resolve these issues with the following approaches:

Solution 1 *Mixed-sound template*
We construct a feature vector template from polyphonic sound mixtures (called a *mixed-sound template*). It would be effective, not only because it is obtained from features that have already been affected by other instruments playing simultaneously, but also because it facilitates to weight features based on their robustness by applying *linear discriminant analysis* (LDA), which maximizes the ratio of the between-class covariance to the with-in class covariance; features that vary because of the overlapping of featurency components have high variances within the class (instrument), which are given low weights by LDA.

Solution 2 *Pitch-dependent timbre model*
We use a pitch-dependent timbre model, called an *F0-dependent multivariate normal distribution* [9], to solve the pitch dependency problem. It approximates the pitch dependency of features representing the timbres of musical instruments as a function of F0. Although our previous experiments [9] showed the effectiveness of this method for solo musical sounds, we have not yet confirmed it for polyphonic music.

Solution 3 *Musical-context-based a priori probabilities*
Our key idea in using musical context is to apply, when calculating the a postriori probability given by $p(\omega_i|\boldsymbol{x}_k) = p(\boldsymbol{x}_k|\omega_i)p(\omega_i)/p(\boldsymbol{x})$ of a musical note n_k, the a posteriori probabilities of temporally neighboring notes to the a priori probabilities.

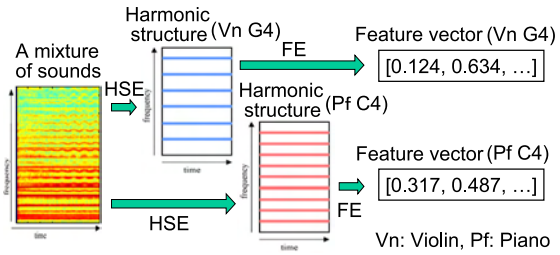

Figure 2: Overview of the process of constructing a mixed-sound template. HSE and FE represent harmonic structure extraction and feature extraction, respectively.

3 MIXED-SOUND TEMPLATE

As previously described, we construct a feature vector template, called a *mixed-sound template*, from polyphonic sound mixtures. Figure 2 has an overview of the process of constructing the mixed-sound template. The sound of each note used in making the template has been beforehand labeled with the instrument name, the pitch, the onset time, and the duration. By using these labels, we extract the harmonic structure corresponding to each note from the power spectrum. We then extract acoustic features from the harmonic structure. We thus obtain a set of many feature vectors extracted from sound mixtures.

The main issue in constructing the mixed-sound template is to design an appropriate subset of sound mixtures because there are an infinite number of possible combinations of musical sounds. These combinations mainly consist of *note combinations*, related to which harmonic components could be affected by other sounds, and *instrument combinations*, related to how much each frequency component is affected (the amount depends on the instrument). In particular, the former causes a serious problem because musical instruments have wide pitch ranges.

To solve this problem, we focus on the fact that not all combinations usually appear in music. Because the complete set of combinations contains many disharmonious ones (*e.g.*, simultaneously playing three notes of C4, C#4 and D4) that are rarely used in music, it is not necessary to cover all possible combinations. To obtain only combinations that actually appear in music, the template is made from (*i.e.*, is trained on) sound mixtures performed based on the scores of musical pieces. Note that this training could be done on musical pieces that are not used for identification. Because our method considers only relative-pitch relationship for feature extraction, once a note combination has been trained, other transposed note combinations are not generally necessary. We can thus expect that a sufficient number of sound combinations can be trained on a small set of musical pieces.

4 F0-DEPENDENT MULTIVARIATE NORMAL DISTRIBUTION

The key idea behind our method is to approximate the pitch dependency of each feature representing the timbres of musical instrument sounds as a function of F0. An F0-dependent multivariate normal distribution [9] has two parameters: an *F0-dependent mean function* and an *F0-normalized covariance*. The former represents the pitch dependency of features and the latter represents the non-pitch dependency. The reason why the mean of a distribution of tone features is approximated as a function of F0 is that tone features at different pitches have different positions (means) of distributions in the feature space. Approximating the mean of the distribution as a function of F0 makes it possible to model how the features will vary according to the pitch with a small set of parameters.

4.1 Parameters of F0-dependent multivariate normal distribution

The following two parameters of the F0-dependent multivariate normal distribution $\mathcal{N}_{\mathrm{F0}}(\boldsymbol{\mu}_i(f), \Sigma_i)$ are estimated for each instrument ω_i.

- **F0-dependent mean function $\boldsymbol{\mu}_i(f)$**
  For each element of the feature vector, the pitch dependency of the distribution is approximated as a function (cubic polynomial) of F0 using the least square method.
- **F0-normalized covariance Σ_i**
  The F0-normalized covariance is calculated with the following equation:
  $$\Sigma_i = \frac{1}{n_i} \sum_{\boldsymbol{x} \in \chi_i} (\boldsymbol{x} - \boldsymbol{\mu}_i(f_{\boldsymbol{x}}))(\boldsymbol{x} - \boldsymbol{\mu}_i(f_{\boldsymbol{x}}))',$$
  where χ_i is the set of the training data of the instrument ω_i and n_i is the total number. $f_{\boldsymbol{x}}$ denotes the F0 of the feature vector $\boldsymbol{x}$.

4.2 Bayes decision rule for F0-dependent multivariate normal distribution

Once the parameters of the F0-dependent multivariate normal distribution have been estimated, the Bayes decision rule is applied to identify the name of the instrument. The Bayes decision rule for the F0-dependent mutltivariate normal distribution is given by the following equation [9]:

$$\hat{\omega} = \operatorname*{argmax}_{\omega_i} \Big\{ -\frac{1}{2} D^2_{\mathrm{M}}(\boldsymbol{x}, \boldsymbol{\mu}_i(f)) - \frac{1}{2} \log |\Sigma_i| + \log p(\omega_i) \Big\},$$

where D^2_{M} is the squared Mahalanobis distance defined by

$$D^2_{\mathrm{M}}(\boldsymbol{x}, \boldsymbol{\mu}_i(f)) = (\boldsymbol{x} - \boldsymbol{\mu}_i(f))' \Sigma_i^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_i(f)).$$

The a priori probability $p(\omega_i)$ is determined based on the a posteriori probabilities of temporally neighboring notes as described in the next section.

5 USING MUSICAL CONTEXT

As previously mentioned, the key idea behind using musical context is to apply, when calculating the a posteriori probability given by $p(\omega_i|\boldsymbol{x}_k) = p(\boldsymbol{x}_k|\omega_i)p(\omega_i)/p(\boldsymbol{x})$ of a musical note n_k, a posteriori probabilities of temporally neighboring notes to the a priori probability $p(\omega_i)$ of the note n_k (Figure 3). To achieve this calculation, we have to resolve the following two issues:

Issue 1 *How to find notes that are played on the same instrument as the note n_k from neighboring notes.*
Because various instruments as well as that for the note n_k are played at the same time, an identification system has to find notes that are played on the same instrument as the note n_k from notes on the various instruments. This is not easy because it is mutually dependent on musical instrument identification.

Issue 2 *How to calculate a posteriori probabilities of neighboring notes.*
Calculating the a posteriori probabilities of temporally

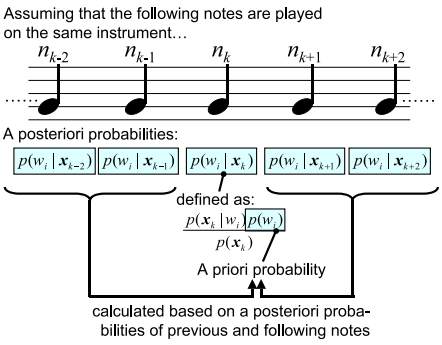

Figure 3: Basic idea for using musical context. To calculate the posteriori probability of the note n_k, the a posteriori probabilities of temporally neighboring notes of n_k are used.

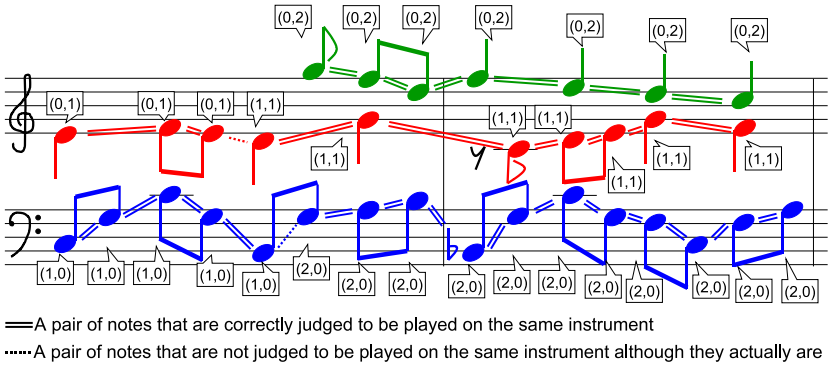

Figure 4: An example of judgment of whether notes are played on the same instrument or not. The tuple "(a, b)" in the figure represents $s_{\mathrm{h}}(n_k) = \mathsf{a}$ and $s_{\mathrm{l}}(n_k) = \mathsf{b}$.

neighboring notes also suffers from the mutual dependency problem, that is, $p(\omega_i|\boldsymbol{x}_k)$ is calculated from $p(\omega_i|\boldsymbol{x}_{k-1})$ etc., while $p(\omega_i|\boldsymbol{x}_{k-1})$ is calculated from $p(\omega_i|\boldsymbol{x}_k)$ etc.

We resolve these issues as follows:

Solution 1 *Use of musical role consistency*
To solve **Issue 1**, we exploit *musical role consistency*, which is musical heuristics that means each instrument has a single musical role (*e.g.*, a principal melody or bass line) from the beginning to the end of a musical piece. Kashino *et al.* [5] also used musical role consistency to generate music streams. They designed two kinds of musical roles: the highest and lowest notes (usually corresponding to the principal melody and bass lines). This method, however, had problems in that it could cause ambiguity when applied to a piece where four or more instruments are being played simultaneously and that it could mistakenly determine, when a principal melody was temporarily absent, the highest note to be a principal melody. To help solve these problems, in this paper, we define a musical role as being based on how many simultanously played notes there are in the higher or lower pitch range. Let $s_{\mathrm{h}}(n_k)$ and $s_{\mathrm{l}}(n_k)$ be the maximum number of simultaneously played notes in the higher and lower pitch ranges when the note n_k is being played, respectively. Then, the two notes, n_k and n_j, are considered to be played on the same instrument if and only if $s_{\mathrm{h}}(n_k) = s_{\mathrm{h}}(n_j)$ and $s_{\mathrm{l}}(n_k) = s_{\mathrm{l}}(n_j)$ (Figure 4).

Solution 2 *Two-pass calculation*
To solve **Issue 2**, we pre-calculate a posteriori probabilities without musical context. After this calculation, we calculate them again using the a posteriori probabilities of temporally neighboring notes.

[1st pass] Pre-calculation of a posteriori probabilities

For each note n_k, the a posteriori probability $p(\omega_i|\boldsymbol{x}_k)$ is calculated by considering the a priori probability $p(\omega_i)$ to be a constant, because the a priori probability, which depends on the a posteriori probabilities of temporally neighboring notes, cannot be determined in this step.

[2nd pass] Re-calculation of a posteriori probabilities

This pass consists of three steps:

(1) *Finding notes played on same instrument*
Notes that satisfy $\{n_j \mid s_{\mathrm{h}}(n_k) = s_{\mathrm{h}}(n_j) \cap s_{\mathrm{l}}(n_k) = s_{\mathrm{l}}(n_j)\}$ are extracted from temporally neighboring notes of n_k. This extraction is performed from the nearest notes to the farthest notes, and stops when c notes are extracted. Let $\mathcal{N}$ be the set of the extracted notes.

(2) *Calculating a priori probability*
The a priori probability of the note n_k is calculated based on the a posteriori probabilities of the notes extracted in the previous step. Let Z_{n_k} be a random variable that represents the instrument for the note n_k. Then, the probability $p(Z_{n_k} = \omega_i)$ that the instrument for the note n_k will be ω_i is the a priori probability to be calculated and can be expanded as follows:

$$p(Z_{n_k} = \omega_i) = p(Z_{n_k} = \omega_i \mid \forall n_j \in \mathcal{N} : Z_{n_j} = \omega_i) \times \prod_{n_j \in \mathcal{N}} p(Z_{n_j} = \omega_i)$$

The first factor of the right side of this equation represents the probability that the note n_k will be played on the instrument ω_i when all the extracted neighboring notes of n_k are played on ω_i. Although this can be acquired through statistical analysis, we use $1 - (1/2)^{2c}$ for simplicity. This is based on the heuristics that, as more notes are used to represent a context, the context information is more reliable. The a posteriori probability calculated in the first pass is used to calculate $p(Z_{n_j} = \omega_i)$.

(3) *Updating a posteriori probability*
The a posteriori probability is re-calculated using the a priori probability calculated in the previous step.

6 IMPLEMENTATION

6.1 Overview

Figure 5 has an overview of our MII system. Given an audio signal of polyphonic music, the system first calculates a spectrogram using the short-time Fourier transform (STFT) and then obtains the pitch, the onset time and the duration of each note. Because our focus was solely on evaluating the performance of MII by itself, we manually fed correct note data into the system. Then, it identifies the instrument of each note through four steps of feature

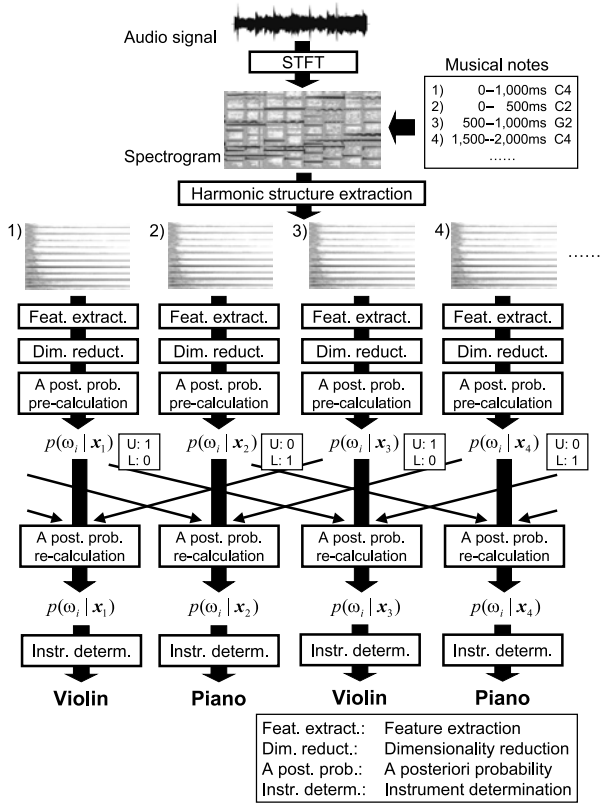

Figure 5: Overview of our musical instrument identification system.

extraction, dimensionality reduction, a posteriori probability calculation, and instrument determination.

6.2 Short-time Fourier transform

The spectrogram of the given audio signal is calculated by STFT shifted by 10 ms (441 points at 44.1 kHz sampling) with an 8192-point Hamming window.

6.3 Harmonic structure extraction

The harmonic structure of each note is extracted according to the manually fed note data. Spectral peaks corresponding to the first 10 harmonics are extracted from the onset time to the offset time. Then, the frequency of the spectral peaks are normalized so that the temporal mean of F0 is 1.

Then, the harmonic structure is trimmed because training and identification need notes with a fixed-length duration. Because a template with a long duration is more stable and robust than a template with a short one, it is better to trim a note as long as possible. We therefore prepare three templates with different durations (300, 450, and 600 ms), and the longest within the actual duration of each note is automatically selected and used for training and identification. Notes shorter than 300 ms are excluded from identification.

6.4 Feature extraction

Features that are useful for identification are extracted from the harmonic structure of each note. From a feature set that we previously proposed [9], we selected 43 features (for the 600-ms template), summarized in Table 1, that we expected to be robust with respect to sound mixtures. We use 37 and 31 feautres for the 450- and 300-ms template, respectively, because some features are excluded due to limitations with note durations.

Table 1: Overview of 43 features

Spectral features	
1	Spectral centroid
2	Relative power of fundamental component
3–10	Relative cumulative power from fundamental to i-th components ($i = 2, 3, \cdots, 9$)
11	Relative power in odd and even components
12–20	Number of components whose duration is $p\%$ longer than the longest duration ($p = 10, 20, \cdots, 90$)
Temporal features	
21	Gradient of straight line approximating power envelope
22–30	Average differential of power envelope during t-sec interval from onset time ($t = 0.15, 0.20, 0.25, \cdots, 0.55$[s])
31–39	Ratio of power at t-sec after onset time
Modulation features	
40, 41	Amplitude and Frequency of AM
42, 43	Amplitude and Frequency of FM

6.5 Dimensionality reduction

The dimensionality of the 43-, 37- or 31-dimensional feature space is reduced through two successive processing steps: it is first reduced to 20 dimensions by applying PCA with a proportion value of 99%, and then further reduced by applying LDA. The feature space is finally reduced to a 3-dimensional space when we deal with four instruments. Because LDA is a dimensionality reduction technique that maximizes the ratio of the between-class covariance to the within-class covariance, it enables us to set high weights for robust features with respect to sound mixtures.

6.6 A posteriori probability calculation

For each note n_k, the a posteriori probability $p(\omega_i|\boldsymbol{x}_k)$ is calculated. This is based on the two-pass method, described in Section 5, with the F0-dependent multivariate normal distribution, described in Section 4.

6.7 Instrument Determination

The instrument maximizing the a posteriori probability $p(\omega_i|\boldsymbol{x}_k)$ is determined as the result for the note n_k.

7 EXPERIMENTS

7.1 Data for experiments

We used two kinds of musical audio data: isolated notes of solo instruments, and polyphonic music. The data on the isolated notes were excerpted from RWC-MDB-I-2001 [10], and were used to create the audio data for the polyphonic music, as well as to obtain the solo-sound template. Details on the solo-instrument audio data are listed in Table 2. All data were sampled at 44.1 kHz with 16 bits.

Both trio and duo music were used as polyphonic music, and their audio data were generated by mixing the audio data listed in Table 2 according to standard MIDI files (SMFs) on a computer. The SMFs we used in the experiments were three pieces taken from RWC-MDB-C-2001 (Piece Nos. 13, 16 and 17) [11]. We chose three or

Table 2: Audio data on solo instruments

Instr. No.	Name	Pitch Range	Variation	Dynamics	Articulation	# of data
01	Piano (PF)	A0–C8	1, 2, 3	Forte,		792
15	Violin (VN)	G3–E7	〃	Mezzo	Normal	576
31	Clarinet (CL)	D3–F6	〃	&	only	360
33	Flute (FL)	C4–C7	1, 2	Piano		221

Table 3: Experimental results

		(a)	(b)	(c)	(d)	(e)
MS Templ.		—	—	√	√	√
F0-dpt.		—	√	—	√	√
Context		—	√	√	—	√
	PF	*83.4%*	*88.6%*	*95.1%*	*83.8%*	*91.6%*
Trio	VN	70.8%	*86.9%*	*83.8%*	74.6%	*86.8%*
No.13	CL	45.0%	34.8%	78.4%	77.3%	*85.5%*
	FL	57.8%	60.1%	*81.2%*	77.2%	*81.8%*
	PF	*86.7%*	*95.6%*	*97.4%*	*91.2%*	*97.7%*
Trio	VN	60.7%	73.4%	*84.5%*	60.9%	*82.9%*
No.16	CL	52.9%	29.0%	*89.1%*	78.0%	*89.8%*
	FL	50.1%	65.7%	79.0%	71.9%	*80.9%*
	PF	80.0%	*87.5%*	*87.5%*	*83.9%*	*91.1%*
Trio	VN	56.5%	71.0%	71.0%	66.8%	*84.8%*
No.17	CL	33.8%	19.4%	74.5%	66.0%	78.6%
	FL	52.3%	51.1%	77.0%	75.0%	78.6%
Average		60.8%	63.6%	83.2%	75.6%	85.8%
	PF	*92.3%*	*96.9%*	*98.8%*	*91.4%*	*97.4%*
Duo	VN	71.1%	*86.6%*	*85.1%*	71.1%	*90.2%*
No.13	CL	58.5%	52.1%	*93.6%*	*83.0%*	*93.6%*
	FL	56.4%	54.5%	*86.1%*	76.2%	*91.1%*
	PF	*93.6%*	*98.6%*	*99.0%*	*95.3%*	*98.7%*
Duo	VN	64.0%	74.0%	*86.0%*	58.9%	78.4%
No.16	CL	63.4%	37.3%	*95.4%*	*83.7%*	*94.1%*
	FL	47.5%	61.7%	*83.0%*	69.5%	*86.5%*
	PF	*89.0%*	*94.9%*	*94.5%*	*92.3%*	*96.7%*
Duo	VN	60.3%	*80.3%*	74.7%	69.1%	*91.9%*
No.17	CL	41.8%	30.7%	*92.8%*	74.5%	*91.5%*
	FL	48.5%	50.9%	77.8%	74.9%	*82.6%*
Average		65.5%	68.2%	88.9%	78.3%	91.1%

MS. Templ.: Mixed-sound template.
Single- and double-underlined numbers denote recognition rates of more than 80% and 90%, respectively.

two simultaneous voices from each piece to generate trio or duo music from these SMFs. To avoid using the same audio data for training and testing, we used 011PFNOM, 151VNNOM, 311CLNOM, and 331FLNOM for the test data and the rest in Table 2 for the training data.

7.2 Experimental results

Table 3 lists results of experiments conducted with the leave-one-out cross-validation method. Using a mixed-sound template, an F0-dependent multivariate normal distribution and musical context, we improved the recognition rate of MII from 60.8% to 85.8% for trio music and from 65.5% to 91.1% for duo music, on average. The observations of the results can be summarized as follows:

- **Effectiveness of the mixed-sound template**
  When we compared the case (e) with the case (b), the recognition rate was improved by more than 20% on average. In particular, those for CL and FL were significantly improved: from 20–65% to 78–94%.
- **Effectiveness of pitch-dependent model**
  F0-dependent multivariate normal distribution improved the recognition rates, on average, from 83.2% to 85.8% for trio music and from 88.9% to 91.1% for duo music. This improvement, however, only occurred when the mixed-sound template was used.
- **Effectiveness of musical context**
  Using musical context also improved the recognition rates, on average, from 75.6% to 85.8% for trio music and from 78.3% to 91.1% for duo music. This was because, in the musical pieces used in our experiments, pitches rarely cross among the melodies of simultaneous voices.

8 CONCLUSION

We have described three methods that work in combination to automatically generate the description of musical instrument names for music information retrieval. To identify the name of the musical instrument performing each note in polyphonic sound mixtures of musical pieces, our methods solve three problems: feature variations caused by sound mixtures, the pitch dependency of timbres, and the use of musical context. In our experiments with three musical pieces including four musical instruments, we found that our methods achieved recognition rates of 85.8% for trio music and 91.1% for duo music on average and confirmed the robustness and effectiveness of those methods. Future work will include to integrate our methods with a musical note estimation method because the pitch and onset time of each note are manually given in our experiments to reveal the performance of instrument identificatin.

Acknowledgements: This research was partially supported by the Ministry of Education, Culture, Sports, Science and Technology (MEXT), Grant-in-Aid for Scientific Research (A), No.15200015, and Informatics Research Center for Development of Knowledge Society Infrastructure (COE program of MEXT, Japan). We thank everyone who has contributed to building and distributing the RWC Music Database [10, 11].

References

[1] B. S. Manjunath, P. Salembier, and T. Sikora. *Introduction of MPEG-7*. John Wiley & Sons Ltd., 2002.
[2] J. Eggink and G. J. Brown. Application of missing feature approach to the recognition of musical instruments in polyphonic audio. *Proc. ISMIR*, 2003.
[3] A. Eronen and A. Klapuri. Musical instrument recognition using cepstral coefficients and temporal features. *Proc. ICASSP*, pages 735–756, 2000.
[4] I. Fujinaga and K. MacMillan. Realtime recognition of orchestral instruments. *Proc. ICMC*, pages 141–143, 2000.
[5] K. Kashino and H. Murase. A sound source identification system for ensemble music based on template adaptation and music stream extraction. *Speech Communication*, 27: 337–349, 1999.
[6] T. Kinoshita, S. Sakai, and H. Tanaka. Musical sound source identification based on frequency component adaptation. *Proc. IJCAI CASA Workshop*, pages 18–24, 1999.
[7] K. D. Martin. *Sound-Source Recognition: A Theory and Computational Model*. PhD thesis, MIT, 1999.
[8] E. Vincent and X. Rodet. Instrument identification in solo and ensemble music using independent subspace analysis. *Proc. ISMIR*, pages 576–581, 2004.
[9] T. Kitahara, M. Goto, and H. G. Okuno. Musical instrument identification based on F0-dependent multivariate normal distribution. *Proc. ICASSP*, volume V, pages 421–424, 2003.
[10] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Music genre database and musical instrument sound database. *Proc. ISMIR*, pages 229–230, 2003.
[11] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Popular, classical, and jazz music databases. *Proc. ISMIR*, pages 287–288, 2002.