NOISE ROBUST MUSIC ARTIST RECOGNITION USING I-VECTOR FEATURES

Hamid Eghbal-zadeh **Gerhard Widmer**
Department of Computational Perception, Johannes Kepler University of Linz, Austria
hamid.eghbal-zadeh@jku.at

ABSTRACT

In music information retrieval (MIR), dealing with different types of noise is important and the MIR models are frequently used in noisy environments such as live performances. Recently, i-vector features have shown great promise for some major tasks in MIR, such as music similarity and artist recognition. In this paper, we introduce a novel noise-robust music artist recognition system using i-vector features. Our method uses a short sample of noise to learn the parameters of noise, then using a Maximum A Postriori (MAP) estimation it estimates clean i-vectors given noisy i-vectors. We examine the performance of multiple systems confronted with different kinds of additive noise in a clean training - noisy testing scenario. Using open-source tools, we have synthesized 12 different noisy versions from a standard 20-class music artist recognition dataset encountered with 4 different kinds of additive noise with 3 different Signal-to-Noise-Ratio (SNR). Using these datasets, we carried out music artist recognition experiments comparing the proposed method with the state-of-the-art. The results suggest that the proposed method outperforms the state-of-the-art.

1. INTRODUCTION

In MIR, the task of music artist recognition [1] is to recognize an artist, from a part of a song. In real life, MIR systems have to cope with different kinds of noise; example situations include music played in a public area such as a pub or in a live performance. MIR systems are usually trained with the high quality data from studio recordings or noise-free audios, yet they may be used in noisy environments.

In this paper, we are targeting a use-case, when an artist recognition mobile app is used in a noisy environment, while the artist recognition models are trained on clean data and are integrated inside the app. In such a use-case, the models can not be trained or adapted on the mobile phone, due to the limitation of computation. But using the app, a short example of the noise can be prepared to improve the performance of the system.

[1] We use the term music artist or artist to refer to the singer or the band of a song.

© Hamid Eghbal-zadeh Gerhard Widmer. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Hamid Eghbal-zadeh Gerhard Widmer. "noise robust music artist recognition using i-vector features", 17th International Society for Music Information Retrieval Conference, 2016.

I-vector extraction is an unsupervised high-level feature extraction technique that extracts excerpt-level features using frame-level features of an audio excerpt. I-vectors were proposed for the first time in the field of speaker verification [4], and then they were frequently used in other areas such as emotion [29], language [5], and accent recognition [1] and audio scene detection [9]. Recently, they were imported into the MIR domain, for singing language identification [17], music artist recognition [7] and music similarity [6]. I-vector features provide a fixed-length low-dimensional representation for songs which can capture specific variabilities from acoustic features using Factor Analysis (FA). In [7], i-vector systems used with noise-free data, and clean mp3 audio files were used in the artist recognition experiments.

I-vector based systems consist of 4 main modules: 1) frame-level feature extraction such as Mel-Frequency Cepstrum Coefficients (MFCC), 2) i-vector extraction, 3) inter-class compensation and finally, 4) i-vector scoring. Within-Class Covariance Normalization (WCCN) and Linear Discriminant Analysis (LDA) are examples of methods used in the inter-class compensation step. Cosine similarity and Probabilistic Linear Discriminant Analysis (PLDA) scoring are examples of methods used in the scoring step.

In this paper, we propose a noise-robust artist recognition system using i-vector features that can be adapted to different kinds of additive noise. We add an estimation step after i-vector extraction, which estimates clean i-vectors given noisy i-vectors in a clean training - noisy testing scenario. Our method is superior because it can be used to adapt i-vector based systems to different kinds of noise, without training the models with noisy data. This is done by learning the parameters of noise for different noisy environments given a short example of additive noise and estimating clean i-vectors that perform well with the models trained on clean data.

2. RELATED WORK

To make a MIR system robust to noise, a solution would be to include noise information inside the MIR models by adding noisy samples in training data. For example, in [18] a method called multi-style training is proposed for speaker verification in noisy environments. This method uses noisy data provided in the training set to extract noisy training i-vectors, with the i-vector extraction models trained on clean data. But it trains the inter-class compensation and scoring models with noisy training i-vectors.

Because MIR models are usually expensive to train, if no noisy data is available in training, the only option would be to use the models trained on clean data for testing the noisy data and try to reduce the effect of noise on the MIR models.

In recent years, research focused on studying the vulnerability of i-vectors to noise and providing methods for noise compensation. The de-noising techniques used with i-vectors can be categorized according to the level they work on (audio, frame-level features, i-vector extraction, or scoring) [22].

In [8, 25] spectral and wavelet based enhancement approaches on the signal level were examined with i-vectors. In [14], spectrum estimators were used for frame-level noise compensation, while in [19–21], multiple approaches were tested using vector Taylor series (VTS) in the cepstral domain for noise compensation on the feature level. Both signal-level and feature-level noise compensation techniques have shown inconsistencies because of their dependency on the type and level of noise. Also, they are mostly designed for speech enhancement and not for music. And in [18, 24] a method was proposed that improves the scoring but it uses noisy audios in model training.

In [16], a novel method called "i-MAP" is proposed for the purpose of speaker verification in noisy environments that uses an extra estimation step between i-vector extraction and inter-class compensation steps. In the estimation step, it estimates clean i-vectors given noisy i-vectors and further uses these estimations with inter-class compensation and scoring models that are trained on clean data. This method benefits from the Gaussian assumption of i-vectors and proposes a MAP estimation of a clean i-vector given a noisy i-vector. The i-MAP method can be used with different types and levels of additive noise with the models that are trained on clean data.

3. REVIEW OF I-VECTOR SYSTEMS

An i-vector refers to vectors in a low-dimensional space called I-vector Space. The i-vector space models variabilities encountered with both the artist and song [6] where, the song variability defines as the variability exhibited by a given artist from one song to another.

The i-vector space is created using a matrix $\mathbf{T}$ known as i-vector space matrix. This matrix is obtained by factor analysis, via a procedure described in details in [4]. In the resulting space, a given song is represented by an i-vector which indicates the directions that best separate different artists. This representation benefits from its low dimensionality and Gaussian distribution which enables us to use the properties of Gaussians in the i-vector space.

Conceptually, a Gaussian mixture model (GMM) mean supervector $\mathbf{M}$ adapted to a song from artist α can be decomposed as follows:

$$\mathbf{M} = m + \mathbf{T}.y \tag{1}$$

where m is the GMM mean supervector and $\mathbf{T}.y$ is an offset. The low-dimensional subspace vector y is a latent variable with the standard normal prior and the i-vector w is a MAP estimate of y. The UBM is a GMM that is trained unsupervised on acoustic features of sufficient amount of songs. Also, $\mathbf{M}$ is assumed to be normally distributed with mean vector m.

The obtained i-vector is an artist and song dependent vector. The low-rank rectangular matrix $\mathbf{T}$ (i-vector space matrix) is used to extract i-vectors from statistical supervectors of songs which are computed using UBM.

Using the UBM, we calculate statistical supervectors for a specific song s. These statistical supervectors are known as 0^{th} and 1^{st} order statistics (N_s and F_s) of song s:

$$(\ 0^{\text{th}} \text{ order statistics})\ \ N^s{}_c = \sum_{t=1}^{L} \gamma_t(c) \tag{2}$$

$$(\ 1^{\text{st}} \text{ order statistics})\ \ F^s{}_c = \sum_{t=1}^{L} \gamma_t(c) Y_t \tag{3}$$

where $\gamma_t(c)$ is the posterior probability of Gaussian component c of UBM for frame t and Y_t is the MFCC feature vector at frame t.

Using the statistical supervectors of songs in training set, we learn the $\mathbf{T}$ matrix via an Expectation Maximization (EM) algorithm: E-step, computes the probability of $P(w|X)$ where X is the given song and w is its i-vector. M-step, optimizes $\mathbf{T}$ by updating the following equation:

$$w = (\mathbf{I} + \mathbf{T}^t \mathbf{\Sigma}^{-1} N(s) \mathbf{T})^{-1} \cdot \mathbf{T}^t \mathbf{\Sigma}^{-1} F(s) \tag{4}$$

where $N(s)$ and $F(s)$ are diagonal matrices with $N^s{}_c.I$ and $F^s{}_c.I$ on diameter and N_s and F_s are 0^{th} and 1^{st} order statistical supervectors of song s. $\mathbf{\Sigma}$ is the diagonal covariance matrix, estimated during factor analysis training. The actual computation of an i-vector w for a given song s can be done using (4) after training $\mathbf{T}$. More information about the training procedure of $\mathbf{T}$ can be found in [4, 15].

4. MAP ESTIMATION OF A CLEAN I-VECTOR

In cases where only clean songs are available for training but at testing the observations are noisy, the best way to improve the performance of clean models encountered with noisy data would be to have an estimation of how clean data looks like. In an i-vector based approach, this estimation is done in the i-vector space by estimating clean i-vectors given noisy i-vectors.

This section describes a solution for music artist recognition in noisy environments which uses a state-of-the-art i-vector based system with an extra estimation step. Our method benefits from a MAP estimation of clean i-vectors given noisy i-vectors that was proposed in i-MAP method [16] for speaker verification applications. The estimation step is applied after i-vector extraction, as it is

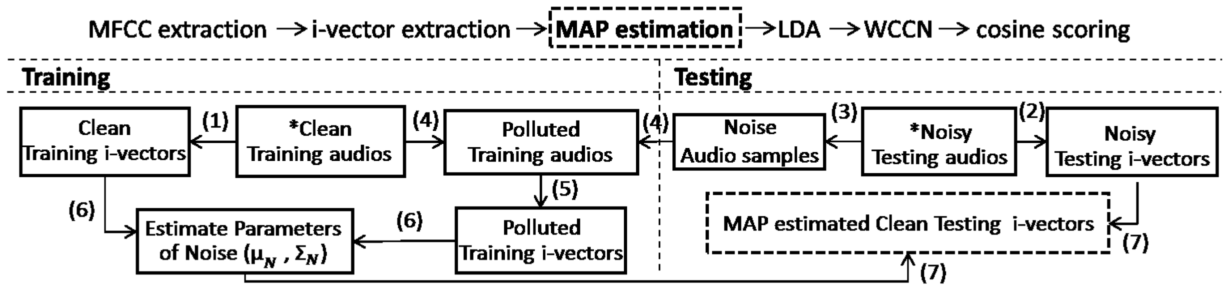

Figure 1. Block diagram of the estimation step in the proposed artist recognition method. The blocks with an asterisk ($*$) indicate the training and testing audio data as the starting point of the diagram.

shown on the top of Figure 1. The resulting estimated i-vectors are used for inter-class compensation and scoring. The estimation step in i-MAP consists of: a) detecting noise using a Voice Activity Detector (VAD), b) synthesizing polluted data, c) estimating the mean and covariance of noise in i-vector space, and finally d) estimating clean i-vectors given noisy i-vectors. To estimate a clean i-vector given a noisy i-vector, i-MAP uses an estimation of *the mean and covariance of noise* in i-vector space by using noise audio samples detected in the noisy testing audios.

Based on i-MAP, the estimation step used in our proposed artist recognition method is described as follows.

Considering two sets of clean and noisy i-vectors, we define the following random variables:

- X: corresponding to the clean i-vectors
- Y: corresponding to the noisy i-vectors

And as i-vectors are normally distributed, we define:

$$X \sim \mathcal{N}(\mu_X, \Sigma_X) \tag{5}$$

$$Y \sim \mathcal{N}(\mu_Y, \Sigma_Y) \tag{6}$$

We denote the probability density functions (PDF) of X and Y by $f(X)$ and $f(Y)$ with the parameters of (μ_X, Σ_X) and (μ_Y, Σ_Y) as mean and covariance of clean and noisy i-vectors, respectively.

We now consider $f(X|Y_0)$ as the conditional PDF of clean i-vectors given a noisy i-vector Y_0. By Bayes' rule,

$$f(X|Y_0) = \frac{f(Y_0|X)f(X)}{f(Y_0)} \tag{7}$$

Using a MAP estimator, a clean i-vector $\widehat{X_0}$ can be estimated by maximizing $f(X|Y_0)$:

$$\widehat{X_0} = \underset{X}{\operatorname{argmax}}\{f(X|Y_0)\} \tag{8}$$

using (7) and taking ln we solve:

$$\frac{\partial}{\partial X}\{\ln f(Y_0|X) + \ln f(X)\} = 0 \tag{9}$$

The solution of (9) would provide an estimation of clean i-vector $\hat{X}_0$ from noisy i-vector Y_0.

4.1 Clean i-vector estimation

In i-MAP [2] it is assumed that when the noise is additive in the speech signal, the noise will be also additive in i-vector space. We keep this assumption and thus, the noise model defines as:

$$Y = X + N \tag{10}$$

where N is a random variable corresponds to noise in i-vector space:

$$N \sim \mathcal{N}(\mu_N, \Sigma_N) \tag{11}$$

where (μ_N, Σ_N) are parameters of noise in i-vector space.

Also, the Gaussian conditional PDF $f(Y_0|X)$ is defined as:

$$f(Y_0|X) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma_N|^{\frac{1}{2}}} e^{-\frac{1}{2}(Y_0-X-\mu_N)^t{\Sigma_N}^{-1}(Y_0-X-\mu_N)} \tag{12}$$

and Gaussian PDF $f(X)$ is:

$$f(X) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma_X|^{\frac{1}{2}}} e^{-\frac{1}{2}(X-\mu_X)^t{\Sigma_X}^{-1}(X-\mu_X)} \tag{13}$$

where (μ_X, Σ_X) are parameters of clean i-vectors, and (μ_N, Σ_N) are parameters of noise in i-vector space. Since $f(Y_0|X)$ and $f(X)$ are Gaussian, the resulting estimate of $f(X|Y_0)$ is also a valid Gaussian PDF as discussed in [22].

Now, by replacing $f(Y_0|X)$ and $f(X)$ by (12) and (13) in (9) and solving it we will have:

$$\widehat{X_0} = ({\Sigma_N}^{-1} + \Sigma_X^{-1})^{-1}(\Sigma_N^{-1}(Y_0 - \mu_N) + \Sigma_X^{-1}\mu_X) \tag{14}$$

$\widehat{X_0}$ is the MAP estimation of a clean i-vector given a noisy i-vector Y_0.

4.2 Parameters of Noise in I-vector Space

To use the MAP estimation provided in (14), we need an estimation of the parameters of clean i-vectors μ_X, Σ_X (which can be estimated from clean training i-vectors [2]) and parameters of noise in i-vector space μ_N, Σ_N.

[2] We use the term clean training audios (e.g. clean training songs) to address the audio data in training set that does not contain any noise. Noisy training audios (e.g. noisy testing songs) indicates audio data in testing set confronted with noise. The word polluted training audios (e.g. polluted training songs) indicates the data that are synthesized from clean

The parameters of clean training i-vectors (μ_X, Σ_X) can be learned by computing the mean and covariance matrix of clean training i-vectors. To learn the parameters of noise in i-vector space, we follow a similar procedure suggested in i-MAP (step numbers can be found in Figure 1): a) we extract clean training i-vectors from clean training songs and noisy testing i-vectors from noisy testing songs (steps 1 and 2). b) we detect noise audio samples in noisy testing songs (step 3). c) we use the noise audio samples detected in step 3 to synthesize polluted training songs from clean training songs (step 4). d) from polluted training songs, we extract a set of i-vectors known as polluted training i-vectors (step 5). e) using clean training i-vectors and polluted training i-vectors, we estimate the parameters of noise in i-vector space (step 6). f) now given noisy testing i-vectors, by having the parameters of noise in i-vector space, we estimate clean testing i-vectors via MAP estimation (step 7). The clean testing i-vectors estimated in step 7 are used for testing experiments in the proposed method.

In step 3, we use a noise detector which has the following steps: We first apply a windowing of 32 ms on the song's audio signal. Then for each window, we calculate the energy of the signal. Using a fixed threshold in energy of each window, we look at the beginning and ending area of each song to detect the areas with lower energy in noisy testing songs. We keep these areas as a set of noise audio samples. These samples are low-activity and assumed to be the examples of the additive noise that we are dealing with. This step provides the short sample of noise (a couple of seconds) in the use-case example described in Section 1.

Further, for each noise area detected in a noisy testing song, we estimate the SNR of the song and the noise sample. We select a limited number of noise areas with longer durations [3]. By adding the selected noise areas (noise audio samples) to clean training songs with the estimated SNR, we synthesize another audio set from our clean training songs which we know as polluted training songs.

Estimating the parameters of noise in step 6 is as follows: After extracting i-vectors from polluted training songs, we compute noise in i-vector space for each song by subtracting a clean training i-vector X_i from polluted training i-vector of that specific song Y_i as follows:

$$N'_i = Y'_i - X_i \tag{15}$$

where Y'_i is the polluted training i-vector extracted from polluted training song $S^p{}_i$ and X_i is the clean training i-vector extracted from clean training song $S^c{}_i$ and N'_i is the noise related to Y'_i in i-vector space. Now the parameters of noise in i-vector space (μ_N, Σ_N) can be calculated by:

$$\mu_N = mean(N') \tag{16}$$

$$\Sigma_N = cov(N') \tag{17}$$

where

$$N' = \{N'_i \mid i = 1, ..., n\} \tag{18}$$

and n is the number of training songs.

To use the proposed method in an adaptive way, we only need new audio samples of noise (which in our use-case we assumed the mobile app can provide, also described a feasible solution in step 3 about how to prepare them) to create a new set of polluted i-vectors to update the parameters of noise in the i-vector space. When the noise is changed, our parameters (μ_N, Σ_N) can also be updated to that noise.

5. EXPERIMENTS

5.1 I-vector Extractor

Our i-vector extractor consists of a UBM with 1024 Gaussian components. This UBM is trained on all the MFCCs of the clean training songs in each fold. The 0^{th} and 1^{st} order statistics (also known as statistical supervectors) are calculated for each song from MFCC features of the song using the UBM. The i-vector space matrix ($\mathbf{T}$) is learned from the statistical supervectors, via an Expectation Maximization (EM) algorithm described in [4, 15] where $\mathbf{T}$ is initialized from random and i-vector space dimensionality is set to 400. Using $\mathbf{T}$ matrix, 400 dimensional i-vectors are extracted for both training and testing set. All the i-vector extraction procedure is done unsupervised. We use 20-dimensional MFCCs in all of our i-vector based systems, extracted with RASTAMAT [10] toolbox with the same configuration as used in [7, 11]. The i-vector space matrix ($\mathbf{T}$) is trained using MSR identity toolbox [26].

5.2 Inter-class Compensation and Scoring

As we described in Section 3, i-vectors contain both artist and song variability. To reduce the song variability in i-vector space, multiple inter-class compensation methods such as length normalization, LDA and WCCN are found effective [4, 12]. Our i-vector inter-class compensation consists of 3 modules: 1) length-normalization, 2) LDA and 3) WCCN. For the scoring, we use a simple cosine scoring approach as detailed in [3].

Length of i-vectors causes negative effects in i-vector space [3, 12]. To discard these effects, we normalize the length of i-vectors by dividing each i-vector by its length. Thus, both training and testing i-vectors are first length normalized [12]. Using the resulting clean training i-vectors, a LDA projection matrix $\mathbf{V}$ is trained and both training and testing i-vectors are projected using $\mathbf{V}$. Then the resulting clean training i-vectors are length normalized again and then used to train a WCCN projection matrix $\mathbf{B}$. The WCCN matrix is used to project both training and testing i-vectors resulted from the LDA step. The final WCCN-projected i-vectors are used for cosine scoring. For each testing i-vector, a similarity score is calculated for each class separately. These scores are calculated given

training audios in training set, using audio samples of noise detected in noisy testing set. Noisy testing i-vectors are i-vectors extracted from noisy audios in testing set, polluted training i-vectors are i-vectors extracted from polluted training audios, and clean training i-vectors are i-vectors extracted from clean training audios. Clean testing i-vectors are estimated via MAP from noisy testing i-vectors.

[3] These noise areas are usually very short in time (a couple of seconds).

the model i-vectors that are computed by averaging LDA-WCCN projected clean training i-vectors for each class. And finally, the class with the maximum score is chosen as the predicted label for the testing i-vector.

5.3 Within-Class Covariance Normalization (WCCN)

Within-Class Covariance Normalization (WCCN) provides an effective compensation that can be used with cosine scoring. After i-vectors are length normalized and projected by LDA, they are encountered with WCCN projection matrix. WCCN scales the i-vector space in the opposite direction of its inter-class covariance matrix, so that directions of intra-artist variability are improved for i-vector scoring. The within-class covariance is estimated using clean training i-vectors as follows:

$$\mathbf{W} = \frac{1}{A}\sum_{a=1}^{\alpha}\frac{1}{n_a}\sum_{i=1}^{n_a}(w^a{}_i - \overline{w^a})(w^a{}_i - \overline{w^a})^t \qquad (19)$$

where $\overline{w^a} = \frac{1}{n_a}\sum_{i=1}^{n_a} w^a{}_i$ is the mean of the LDA projected i-vectors for each artist α. A is the total number of artists, and n_a is the number of songs of each artist α in training set. We use the inverse of the $\mathbf{W}$ matrix to normalize the direction of the projected i-vectors. WCCN projection matrix $\mathbf{B}$ can be estimated such that:

$$\mathbf{B}\mathbf{B}^t = \mathbf{W}^{-1} \qquad (20)$$

5.4 Cosine Scoring

In the i-vector space, a simple cosine scoring has been successfully used to compare two i-vectors [3]. Given a length normalized, LDA and WCCN projected i-vector w_i from an unknown artist, cosine score for artist α is defined as:

$$score(\overline{w_\alpha}, w_i) = \frac{(\overline{w_\alpha})^t.w_i}{\|\overline{w_\alpha}\| \,.\, \|w_i\|} \qquad (21)$$

where $\overline{w_\alpha}$ represents the mean i-vector of artist α, calculated by averaging all the length normalized, LDA and WCCN projected clean training i-vectors extracted from all the songs of artist α. $score(\overline{w_\alpha}, w_i)$ represents the cosine score of testing i-vector w_i for artist α. To predict the artist label for i-vector w_i, the artist with the highest cosine score is chosen as the label for w_i.

5.5 Dataset

In our experiments, we used Artist20 dataset [11], a freely available 20-class artist recognition corpus which consists of 1413 mp3 songs from 20 different artists mostly in pop and rock genres. The dataset is composed of six albums from each of 20 artists. A 6-fold cross validation is also provided with the dataset which is used in all of our experiments. In each fold, 5 out of 6 albums from all the artists were used for training and the rest were used for testing.

For our experiments with noisy data, we synthesized 12 (4×3) different noisy sets from Artist20 dataset with 4 different kinds of additive noise (festival, humming, pink, pub environment) of 3 different SNRs (5db, 10 db and 20 db), by applying the noise to all the songs. For applying the noise, the open-source Audio Degradation Toolbox (ADT) [23] is used.

For all the experiments (except IVEC-CLN and EBLF-CLN), the models are trained on training folds of clean dataset and tested on the testing fold of noisy dataset. For IVEC-CLN and EBLF-CLN experiments, models are trained on training folds of clean dataset and tested on testing fold of clean dataset.

We found noise samples of festival noise in the FreeSound repository [4]. The festival noise sample is an audio recording from a live performance during a festival with a lot of cheering sounds and human speaking in loudspeaker. This noise example is available upon request. For the other additive noises (pub environment, pink-noise, humming) the noise samples provided in ADT are used. The pub environment noise sample, recorded in a crowded pub, and the humming noise is recorded from a damaged speaker.

5.6 Evaluation

To evaluate the performance of different methods dealing with different kinds of noise, the averaged Fmeasure over all the classes is used [5]. The reported results in Table 1 show the mean of the averaged Fmeasures over 6-folds of our cross-validation, $\pm$ the standard deviation (std) of the averaged Fmeasures over folds. To examine the statistical significance, a t-test is applied for each sets of experiments separately, comparing the Fmeasures of 6 folds between the proposed method and each of the baselines (for example: festival noise with 3 db SNR, comparing IVEC-NSY and EBLF-NSY). Each set of experiments for a specific kind of noise with a certain SNR is done independently.

5.7 Baseline Methods

We compare the performance of our proposed method with two baselines. The first baseline is a state-of-the-art standard i-vector based artist recognition system known as *IVEC-NSY*. The reason we chose this baseline is to show the improvements by adding the estimation step to this baseline. We extract 400 dimensional i-vectors using 20-dimensional MFCCs and a 1024 components GMM as UBM. Then we apply length normalization, LDA and WCCN and further apply the cosine scoring to predict the labels.

The second baseline (*EBLF-NSY*) uses an extended version of Block-Level features [28] (EBLF) used in [27]. EBLF are the winner of multiple tasks in MIREX challenge [6] such as music similarity and genre classification and provide a set of 8 song-level descriptors which represent different characteristics for a song. These 8 descriptors contain a good variety of features such as rhythm and

[4] http://freesond.org

[5] Since the number of songs from each artist are more or less the same (6 albums), the averaged Fmeasure seems to be a good measurement for our multi-class artist recognition task.

[6] Annual Music Information Retrieval eXchange (MIREX). More information is available at: http://www.music-ir.org

timbre in a song. Since these features are frequently used for multiple purposes in MIR, we chose them as our second baseline to show how they perform in a noisy environment. For classification, a WEKA [13] implementation of SMO support vector machine is used as described in [27]. The SVM model in this baseline is trained on features of clean training songs and tested on features of noisy testing songs.

To demonstrate the maximum power of our baselines on clean data, we also provided the performance of these methods, dealing with only clean data. In these specific experiments, we train and test the baselines using clean data and the results can be found in the description of Table 1 as *IVEC-CLN* and *EBLF-CLN*.

The performance of *EBLF-CLN* is comparable with the baselines in [7] which compares different approaches for music artist recognition and therefore is a competitive method to be used as a baseline. Also, the performance of *IVEC-CLN* is comparable with the the best results achieved using a state-of-the-art i-vector based artist recognition system reported in [7] (which are known to be the best artist recognition results on Artist20 dataset published so far). Both *IVEC-CLN* and [7] use similar i-vector extractors. The difference between *IVEC-CLN* and [7] is in the inter-class compensation and scoring. We used LDA followed by WCCN as inter-class compensation and a simple cosine scoring, while in [7] only LDA is used as inter-class compensation and the best performance achieved with a Discriminant Analysis classifier.

5.8 The Proposed Method

Our proposed method is shown in Table 1 as *IVEC-MAP*. All the i-vector extraction (UBM,$\mathbf{T}$), inter-class compensation (LDA,WCCN) and scoring models are only trained with clean training data and the only extra information used in the proposed method compared to the baselines, is the mean and covariance of noise in i-vector space. The number of at least 500 i-vectors are needed to estimate the parameters of noise as reported in [16] in a speaker verification scenario. We used all the polluted training i-vectors for parameter estimation, since our training set is not very big ($\sim$ 1100 songs).

5.9 Results and Discussion

By looking at Table 1 it can be seen that the proposed method outperformed the baselines in all 12 cases of 4 different additive noises with 3 different SNRs. Having a closer look at the results suggests the IVEC-NSY baseline performed much better and more robust than EBLF-NSY.By looking at the results dealing with noises of different SNR levels, it can be seen that as expected the lower the SNR (more noise), the lower the performance of our baselines are. Unlike the baselines, the change in SNR does not affect the performance of our proposed method significantly in festival and humming noises. Considering the standard deviation of the averaged Fmeasures for all the 6-folds, results suggest that the proposed method is always higher than 1 std from the performance of EBLF-NSY in all the noises with all different SNRs. When the noise

		Averaged Fmeasure (%)		
snr	nois.	IVEC-NSY	EBLF-NSY	IVEC-MAP
5 db	fest.	68.22±8.85	19.01±23.31	**81.08±7.68**
	hum.	75.28±8.14	5.21±5.15	**82.56±6.82**
	pink	60.44±10.27	6.64±11.58	**74.88±6.67**
	pub	44.11±8.13	14.01±22.27	**71.12±8.09**
10 db	fest.	74.27±8.97	24.32±27.39	**81.91±7.55**
	hum.	77.15±8.97	25.71±26.5	**82.64±7.24**
	pink	68.58±8.22	11.89±16.38	**78.05±7.2**
	pub	66.15±9.04	22.7±25.91	**79.87±7.31**
20 db	fest.	77.28±7.6	36.12±26.76	**81.89±7.47**
	hum.	77.32±7.43	36.89±25.3	**82.86±7.1**
	pink	74.57±7.63	13.93±21.31	**80.54±7.53**
	pub	76.74±7.8	26.17±29.05	**82.63±7.2**

Table 1. Comparison of artist recognition performance of different methods on Artist20 dataset dealing with different kinds and levels of additive noise. The numbers indicate the averaged Fmeasure (as described in Section 5.6) with the standard deviation over all the folds. The performance of the baselines IVEC-CLN and EBLF-CLN on clean data are 83.73±7.58 and 72.26±7.42 respectively.

is in its highest level (SNR=5 db) the proposed method's Fmeasure is higher than IVEC-NSY by 1 std. On average, the proposed method achieved the relative averaged Fmeasure of 28.41, 12.99 and 7.21 percentage points higher than IVEC-NSY encountering additive noises with 5, 10 and 20 db SNR, respectively. Applying a *t-test hypothesis testing* on the averaged Fmeasures for different folds to examine the performance between the proposed method and the baselines (IVEC-NSY and EBLF-NSY), shows that the test rejects the null hypothesis at 5% significance level for all the experiments and our results are statistically significant.

6. CONCLUSION

In this paper, we proposed a noise-robust artist recognition system using i-vector features and a MAP estimation of clean i-vectors given noisy i-vectors. Our method outperformed the state-of-the-art standard i-vector system and EBLF, also showed a stable performance dealing with multiple kinds of additive noise with different SNRs. We showed that by adding an estimation step to a standard i-vector based artist recognition system, the performance in noisy environments can be significantly improved.

7. ACKNOWLEDGMENTS

We would like to acknowledge the help by Mohamad Hasan Bahari of KU Leuven University to this work. We also appreciate helpful suggestions of Waad Ben-Kheder from University of Avignon. This work was supported by the Austrian Science Fund (FWF) under grant no. Z159 (Wittgenstein Award).

8. REFERENCES

[1] Mohamad Hasan Bahari, Rahim Saeidi, David Van Leeuwen, et al. Accent recognition using i-vector, gaussian mean supervector and gaussian posterior probability supervector for spontaneous telephone speech. In *ICASSP*. IEEE, 2013.

[2] Waad Ben Kheder, Driss Matrouf, Jean-François Bonastre, Moez Ajili, and Pierre-Michel Bousquet. Additive noise compensation in the i-vector space for speaker recognition. In *ICASSP*. IEEE, 2015.

[3] Najim Dehak, Reda Dehak, James R Glass, Douglas A Reynolds, and Patrick Kenny. Cosine similarity scoring without score normalization techniques. In *Odyssey*, page 15, 2010.

[4] Najim Dehak, Patrick Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 2011.

[5] Najim Dehak, Pedro A Torres-Carrasquillo, Douglas A Reynolds, and Reda Dehak. Language recognition via i-vectors and dimensionality reduction. In *INTERSPEECH*. Citeseer, 2011.

[6] Hamid Eghbal-zadeh, Bernhard Lehner, Markus Schedl, and Gerhard Widmer. I-vectors for timbre-based music similarity and music artist classification. In *ISMIR*, 2015.

[7] Hamid Eghbal-zadeh, Markus Schedl, and Gerhard Widmer. Timbral modeling for music artist recognition using i-vectors. In *EUSIPCO*, 2015.

[8] A El-Solh, A Cuhadar, and RA Goubran. Evaluation of speech enhancement techniques for speaker identification in noisy environments. In *ISMW*. IEEE, 2007.

[9] Benjamin Elizalde, Howard Lei, and Gerald Friedland. An i-vector representation of acoustic environments for audio-based video event detection on user generated content. In *ISM*. IEEE, 2013.

[10] Daniel PW Ellis. PLP and RASTA (and MFCC, and inversion) in Matlab, 2005. online web resource.

[11] Daniel PW Ellis. Classifying music audio with timbral and chroma features. In *ISMIR*, 2007.

[12] Daniel Garcia-Romero and Carol Y Espy-Wilson. Analysis of i-vector length normalization in speaker recognition systems. In *INTERSPEECH*, 2011.

[13] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 2009.

[14] Cemal Hanilçi, Tomi Kinnunen, Rahim Saeidi, Jouni Pohjalainen, Paavo Alku, F Ertaş, Johan Sandberg, and Maria Hansson-Sandsten. Comparing spectrum estimators in speaker verification under additive noise degradation. In *ICASSP*. IEEE, 2012.

[15] Patrick Kenny. Joint factor analysis of speaker and session variability: Theory and algorithms. *CRIM, Montreal,(Report) CRIM-06/08-13*, 2005.

[16] Waad Ben Kheder, Driss Matrouf, Pierre-Michel Bousquet, Jean-François Bonastre, and Moez Ajili. Robust speaker recognition using map estimation of additive noise in i-vectors space. In *Statistical Language and Speech Processing*. Springer, 2014.

[17] Anna M Kruspe. Improving singing language identification through i-vector extraction. In *DAFx*, 2011.

[18] Yun Lei, Lukas Burget, Luciana Ferrer, Martin Graciarena, and Nicolas Scheffer. Towards noise-robust speaker recognition using probabilistic linear discriminant analysis. In *ICASSP*. IEEE, 2012.

[19] Yun Lei, Lukas Burget, and Nicolas Scheffer. A noise robust i-vector extractor using vector taylor series for speaker recognition. In *ICASSP*. IEEE, 2013.

[20] Yun Lei, Moray McLaren, Luciana Ferrer, and Nicolas Scheffer. Simplified vts-based i-vector extraction in noise-robust speaker recognition. In *ICASSP*. IEEE, 2014.

[21] D Martinez, Lukas Burget, Themos Stafylakis, Yun Lei, Patrick Kenny, and Eduardo Lleida. Unscented transform for ivector-based noisy speaker recognition. In *ICASSP*. IEEE, 2014.

[22] Driss Matrouf, Waad Ben Kheder, Jean-François Bonastre, Moez Ajili, and Pierre-Michel Bousquet. Dealing with additive noise in speaker recognition systems based on i-vector approach. In *EUSIPCO*. IEEE, 2015.

[23] Matthias Mauch and Sebastian Ewert. The audio degradation toolbox and its application to robustness evaluation. In *ISMIR*, 2013.

[24] Simon JD Prince and James H Elder. Probabilistic linear discriminant analysis for inferences about identity. In *Computer Vision, ICCV*. IEEE, 2007.

[25] Seyed Omid Sadjadi and John HL Hansen. Assessment of single-channel speech enhancement techniques for speaker identification under mismatched conditions. In *INTERSPEECH*, 2010.

[26] Seyed Omid Sadjadi, Malcolm Slaney, and Larry Heck. Msr identity toolbox-a matlab toolbox for speaker recognition research. *Microsoft CSRC*, 2013.

[27] Klaus Seyerlehner, Markus Schedl, Tim Pohle, and Peter Knees. Using block-level features for genre classification, tag classification and music similarity estimation. *MIREX*, 2010.

[28] Klaus Seyerlehner, Gerhard Widmer, and Tim Pohle. Fusing block-level features for music similarity estimation. In *DAFx*, 2010.

[29] Rui Xia and Yang Liu. Using i-vector space model for emotion recognition. In *INTERSPEECH*, 2012.