*Journal of Computer Science and Cybernetics, V.38, N.1 (2022), 63–84*
DOI no. 10.15625/1813-9663/38/1/15961

# AUTOMATIC IDENTIFICATION OF SOME VIETNAMESE FOLK SONGS CHEO AND QUANHO USING DEEP NEURAL NETWORKS

CHU BA THANH[1,2,*], TRINH VAN LOAN[1,2], DAO THI LE THUY[3]

[1] *HungYen University of Technology and Education, Khoai Chau District, Hung Yen Province, Viet Nam*
[2] *School of Information Communication and Technology, University of Science and Technology, 01 Dai Co Viet Street, Hai Ba Trung District, Ha Noi, Viet Nam*
[3] *University of Transport and Communications, 3 Cau Giay Street, Lang Thuong Ward, Dong Da District, Ha Noi, Viet Nam*



**Abstract.** We can say that music in general is an indispensable spiritual food in human life. For Vietnamese people, folk music plays a very important role, it has entered the minds of every Vietnamese person right from the moment of birth through lullabies for children. In Vietnam, there are many different types of folk songs that everyone loves, and each has many different tunes. In order to archive and search music works with a very large quantity, including folk songs, it is necessary to automatically classify and identify those works. This paper presents the method of determining the feature parameters and then using the Convolution Neural Network (CNN), Long-Short Term Memory networks (LSTM), and Convolutional Recurrent Neural Network (CRNN) to classify and identify some Vietnamese folk tunes as *Quanho* and *Cheo*. Our experimental results show that the average highest classification and identification accuracy are 99.92% and 97.67%, respectively.

**Keywords.** Identification, classification, folk songs, Vietnamese, *Cheo*, *Quanho*, CNN, LSTM, CRNN.

## 1. INTRODUCTION

With the explosion of the Internet, music stored and transmitted digitally (hereinafter referred to as digital music) are becoming more and more popular in life of human. Today, owning thousands of music tracks is common for individuals and they can build their own music library using music management software. The manual classification and labeling method required a lot of time, patience and was very confusing, making it impossible to keep up with the development of digital music. Therefore, the requirement is that music labeling/classifications must be done automatically.

There have been many studies using traditional machine learning algorithms such as

*Corresponding author.
*E-mail addresses*: thanhcb.dce@utehy.edu.vn (C.B.Thanh), loantv@soict.hust.edu.vn (T.V.Loan), thuydtl@utc.edu.vn (D.T.L.Thuy)

KNN [1, 2], HMM [3–5], GMM [6–8], SVM [9, 10],... to classify music by genre. However, with the architectural boom of artificial neural networks over the past decade, especially CNN (Convolutional Neural Network), it has been very effective in areas such as image recognition [11–13] and natural language processing. With that, the huge data explosion in all sectors, combined with the incredible development of high-performance computing hardware systems (GPU - Graphics Processing Unit), has made neural networks more and more widely studied and applied in many fields of life.

The success of CNN in the field of image recognition has inspired researchers in the field of MIR (Music Information Retrieval), especially the music genre classification. In the music signal, the spectrum of the music signal is also considered an image, it contains full information about the time, frequency and amplitude of a track. Therefore, it is entirely possible to use neural networks to classify musical genres and this is considered a new solution compared to the traditional methods mentioned above. In this paper, we propose to use CNN, LSTM and CRNN to classify two folk music genres, namely *Cheo* and *Quanho*, and identify some tunes in each of these folk music genres.

The remainder of the paper is organized as follows. An overview of relevant studies is presented in Section 2. Section 3 presents the feature selection method and the used model. Results and discussion are presented in Section 4, the final section is our conclusion.

## 2. RELATED WORKS

Initiation of research on music classifications was carried out by Matityaho and Furst [14] in 1995. In their publication, the authors used a neural network-based model to classify two genres such as classical and pop in 8 works of music by Mozart and Tchaikovsky. In the following years, the number of publications in this field was not much, and it was until October 2000 that the International Symposium on Music Information Retrieval (ISMIR) was held for the first time. The first has attracted many researchers interested in this field. Table 1 [15] shows information about venue and years of the ISMIR conference.

Along with ISMIR, MIREX (Music Information Retrieval Evaluation exchange proposed at ISMIR 2005) is also held annually to promote advancements in the MIR field, and to evaluate systems and algorithms in this field. It is managed by the International Music Information Retrieval Systems Evaluation Laboratory (IMIRSEL) at the University of Illinois, Urbana-Champaign city, Chicago. By 2002, Tzanetakis and Cook [16] completed the first genre-based music dataset called GTZAN containing 1000 30s excerpts of 10 genres (blues, classical, country, disco, hiphop, jazz, metal. , pop, reggae, rock). This is considered to be the first public dataset used for classifying music by genre and is the most used in the articles published in this field [15].

Since the inception of Tzanetakis and Cook's research [16], genre classifications have become a hot topic in the MIR community [17–19]. According to Sturm's [20] statistics (up to 2012), the number of published articles in the field of music genre classification is about 500 articles. However, classifying music genres is still a very difficult problem, as boundaries between different genres can be very blurred. On the other hand, the concept of the genre is still unclear, ambiguous, highly dependent on human emotions and perception [21]. Figure 1 is the statistics of articles published in the field of Music Classification by Genre (MCG) of Sturm [20] as of 2012.

Table 1: Time and venue for annual ISMIR conferences

| Conferences | Years | Venue |
|---|---|---|
| ISMIR 2020 | 2020 | 11-15 October 2020, Montreal (Canada) |
| ISMIR 2019 | 2019 | 4-8 November 2019, Delft (Netherlands) |
| ISMIR 2018 | 2018 | 23-27 September 2018, Paris (France) |
| ISMIR 2017 | 2017 | 23-27 October 2017, Suzhou (China) |
| ISMIR 2016 | 2016 | 7-11 August 2016, New York City (USA) |
| ISMIR 2015 | 2015 | 26-30 October 2015, Malaga (Spain) |
| ISMIR 2014 | 2014 | 27-31 October 2014, Taipei (Taiwan) |
| ISMIR 2013 | 2013 | 4-8 November 2013, Curitiba (Brazil) |
| ISMIR 2012 | 2012 | 8-12 October 2012, Porto (Portugal) |
| ISMIR 2011 | 2011 | 24-28 October 2011, Miami (USA) |
| ISMIR 2010 | 2010 | 9-13 August 2010, Utrecht (Netherlands) |
| ISMIR 2009 | 2009 | 26-30 October 2009, Kobe (Japan) |
| ISMIR 2008 | 2008 | 14-18 September 2008, Philadelphia (USA) |
| ISMIR 2007 | 2007 | 23-30 September 2007, Vienna (Austria) |
| ISMIR 2006 | 2006 | 8-12 October 2006, Victoria, BC (Canada) |
| ISMIR 2005 | 2005 | 11-15 September 2005, London (UK) |
| ISMIR 2004 | 2004 | 10-15 October 2004, Barcelona (Spain) |
| ISMIR 2003 | 2003 | 26-30 October 2003, Baltimore, Maryland (USA) |
| ISMIR 2002 | 2002 | 13-17 October 2002, Paris (France) |
| ISMIR 2001 | 2001 | 15-17 October 2001, Bloomington, Indiana (USA) |
| ISMIR 2000 | 2000 | 23-25 October 2000, Plymouth, Massachusetts (USA) |

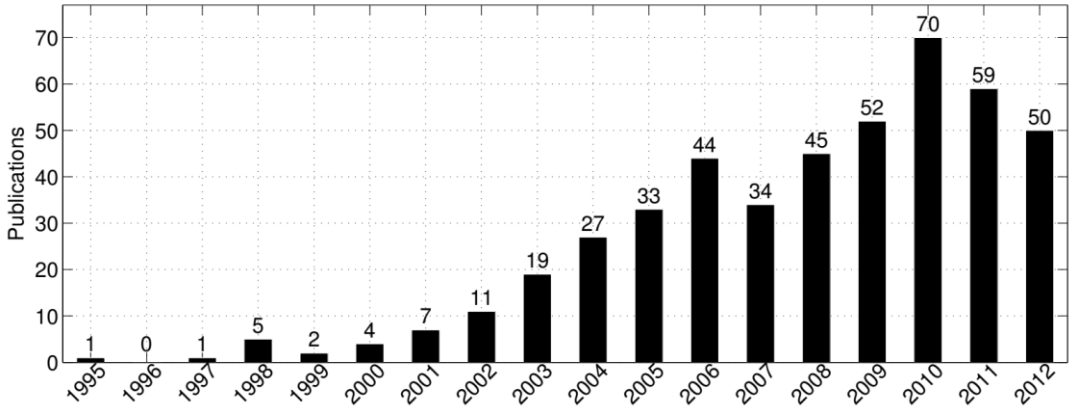


Figure 1: The number of publications on MCG from 1995 to 2012 [20]

The previously published studies on music classification were mainly based on content and using traditional tools such as KNN [1,2], HMM [3–5], GMM [6–8], SVM [9,10], Decision Tree [22]..., where the analysis of musical structures from sound is the decisive factor for a labeled genre of music [17,21,23]. In recent years, deep learning models have been widely used in the realm of musical classification by genres [24–28]. The research in [25] has shown that the neural network algorithms are more effective than the traditional algorithms previously done in the field of music classification by genre.

In the Zalo AI Challenge 2018 contest [29] held by the Vietnamese AI community, Dung

Nguyen Ba's group of authors won the first prize in the field of recognizing Vietnamese songs. The dataset of 867 files of 10 Vietnamese genres was broken down by the authors group, extracted the sound spectrum, then converted into images and finally used Deep Network for analysis, gave the highest recognition results, reaching the accuracy 70%. In [30] we built a RAN (Residual Attention Network) model and tested on this dataset; the results improved the recognition accuracy rate to 71.7%.

Vietnam is a multi-ethnic, multi-lingual country with 54 ethnic groups and more than 90 different languages, so Vietnamese folk music has an extremely rich number of genres [31]. In studies [31–34], the GMM model was used to identify *Cheo* and *Quanho* melodies. The highest average accuracy rates were 85.6% for the identification and 93.8% for the classification test. In fact, the GMM model proved to be effective in many different fields such as speaker recognition [35], emotion recognition [36], dialect identification [37],..., and there have also been studies using GMM model to identify music genres [6,8]. The advantage of the GMM model is that it does not take much time to compute, and the data does not need to be very large for the algorithm to converge. Research results in [31] showed that with a short excerpt of a tune, GMM also gave convergent results. The application of the deep neural network models was previously difficult due to technological limitations. In this paper, with the desire to improve recognition accuracy in terms of available GPU, we will use ANN and deep learning to classify *Cheo* and *Quanho* genres and identify some *Cheo* and *Quanho* tunes.

## 3. METHODOLOGY

### 3.1. Data collection

The data used for our experiments includes 2 datasets of Vietnamese folk music, namely *Cheo* and *Quanho*, each dataset contains 25 *Cheo* or *Quanho* tunes. *Quanho*'s dataset is supplied by Kinh Bac Quan Ho Cultural Center - Bac Ninh province, while *Cheo*'s dataset is selected from various sources on the Internet. All data has a sampling frequency of 16 kHz and 16 bits/sample. The total capacity for the *Cheo*'s dataset is 457 MB, and the *Quanho*'s dataset is 446 MB.

This is the dataset that we have used in previous studies [31, 34]. The experiments for *Cheo* and *Quanho* classification and identification using the GMM model were published in [34]. In [31], the experiments of *Cheo* and *Quanho* identification are based on short excerpts using GMM and i-Vector.

*Quanho* and *Cheo* classification problems in our research belong to supervised learning and not to semi-supervised learning. In our problem, the data is all labeled, so we do not use unlabeled data as in the case of semi-supervised learning. On the other hand, in fact, there are New-Old combined forms (*Tan Co giao duyen*) such as the combination of *Cai luong* (*Vong Co*) and neo-music. However, in fact, to our knowledge, data combining *Cheo* and *Quanho* or combining *Cheo*, *Quanho* with neo-music exists very rarely or even does not exist. Therefore, this issue was not posed in our study.

For the experiments in this paper, data augmentation is applied by dividing each file in the dataset into two 10-second files. The division of each *Quanho* tune into two halves only means that it is still *Quanho*, but these two halves are different in terms of lyrics and the melody is not identical. Therefore, the result of this division can also be seen as we have two

different data but still belong to *Quanho* and not another folk song genre. The same goes for dividing a *Cheo* tune into two halves. Therefore, the file number of each tune will be 40 files and each dataset will contain 1000 files. The total number of files on the dataset used for our experiments is 2000 files.

### 3.2. Feature extraction

We use the LibROSA [38] API to analyze and extract features from experiement data. LibROSA is a package in Python, developed for audio and music analysis. Selected features include 157 mel spectrum coefficients and 26 other parameters (pitch (1), tonnetz (6), spectral contrast (7), and chroma (12)).

In the field of sound processing, the mel spectrum coefficients are typical characteristics of the different frequencies present in the signal. In music signals, in addition to the inherent characteristics of sound, it also has a number of other characteristic parameters such as pitch, tonnetz, spectral contrast and chroma.

Pitch reflects the bass or the treble of the perceived sound, each sound is emitted with a fixed frequency. The pitch of the sound depends on these frequencies, the higher the frequency of the sound, the higher the frequency of perceived sound will increase and vice versa.

The tonnetz concept was first described by Leonhard Euler in 1739 [39, 40]. It is a layout of notes in music (which can be called a grid or a graph of notes), which shows the relationship between notes and is often used in harmony.

Spectral contrast is the difference between the peak of the spectrum and the concave region of the spectrum in each sub-band. In music signals, the peaks of the spectrum correspond to the harmonic components, and the depression of the spectrum corresponds to the non-harmonic (or noise) components. This feature, therefore, reflects the relative distribution of harmonic and non-harmonious features in the spectrum. The MFCC (Mel Frequency Cepstral Coefficient) characteristics obtained by being based on the mean spectrum distribution are not sufficient to represent the characteristics of the musical spectrum. Research in [41] when testing five types of music (Baroque, Romantic, Pop, Jazz and Rock) using spectral contrast feature showed that, the average accuracy rate was 82.3%, compared with 74.1% when using MFCC.

Music is a continuation of sounds that are chronologically organized. Chroma-based [42] features are a powerful tool for music analysis. Every chord recognition process relies on some kind of chroma representation [43–46]. Hence, chroma-based features are a powerful tool in retrieving sounds based on content such as cover song identification [47, 48], audio matching [49–51] or audio hashing [52, 53].

For our experiments, the extracted feature parameters are divided into 2 sets of parameters. The first parameter set (S1) includes 157 mel spectrum coefficients and the second parameter set (S2) consists of the parameter set S1 and an additional 26 coefficients as shown in Table 2. The number of frames for each file (10s) is taken as the number of parameters, i.e., equal to 157 frames (S1) or 183 frames (S2). So, the corresponding frame rate is 15.7 frames/s or 18.3 frames/s. The frame width is taken as the number of samples to calculate the FFT and is equal to 2048 samples.

Table 2: Feature Dataset summary

| Features | # of Features | S1 | S2 |
|---|---|---|---|
| Mel spectrogram | # 157 | 157 | 183 |
| pitch | # 1 | | |
| tonnetz | # 6 | | |
| spectral contrast | # 7 | | |
| chroma | # 12 | | |
| Total | 183 | 157 | 183 |

### 3.3. Deep neural networks for experiments

Like human behavior, with complex algorithms and artificial neural networks, deep learning has been able to train computers to learn from experience, classify and recognize data and images. There have been many artificial neural network models used in deep learning. Among those models, Convolutional Neural Network (CNN) is widely used for image and object recognition and classification. For CNN, convolutional layers are used to capture the relationships between pixels using a set of filters or kernels that slide over the image with a certain stride and calculate the convolution operation at each step. CNN proved to be suitable for classification algorithms. Besides classification algorithms, there are also object detection algorithms. For example, detecting objects such as people, certain objects (planes, cars...) or animals (birds, dogs, and cats...) in a photo. In such cases, CNN variants are more appropriate such as R-CNN (Region-Based Convolutional Neural Networks) [54], Fast R-CNN [55], and Faster R-CNN [56] models. The general idea of CNN's variant models is that instead of having to select a very large number of regions, select only a smaller number (as equal to 2000 in R-CNN). For R-CNN, each region proposal is fed independently to the CNN for feature extraction. This makes it impossible to run R-CNN in real-time. Fast R-CNN and Faster R-CNN are improved versions of R-CNN. The Fast R-CNN uses selective search to generate its region proposals. Faster R-CNN is faster than Fast R-CNN thanks to the region proposal network.

If CNNs are feed-forward neural networks that use filters and pooling layers, then RNNs (Recurrent Neural Networks) [57, 58] feed results back into the network. Recurrent neural networks are helpful in modeling sequence data. A variation of RNN is LSTM (Long Short-Term Memory) [59]. LSTM has been designed to address the vanishing and exploding gradient problems of conventional RNN.

Within the limited scope of this paper to perform classification and identification of *Cheo* and *Quanho*, which will use feature parameters like image form but not belonging to object detection in the image, we will use CNN, LSTM, and CRNN (Convolutional Recurrent Neural Networks - a combination of CNN and LSTM) [60, 61] models to conduct our research. The following is a detailed description of these models.

#### 3.3.1. Convolutional neural network (CNN)

The CNN structure was first proposed by Fukushimain in 1980 [62]. However, it was not widely used because the algorithm used for training was difficult to use at the time. It was until the 1990s that LeCun et al. [63] applied a gradient descent learning algorithm based on CNN and obtained successful results. Subsequently, the researchers continued to prove

that CNN gave good results in pattern recognition [64,65] (digit recognition, alpha numeric, traffic signs and other classes of objects). Recently, CNN has been applied a lot in the field of classifying music by genre [28,66–69].

The CNN network consists of a set of basic layers, including [70]: Convolutional layer, Non-linear layer, Pooling layer, and Fully connected layers). These layers are linked together in a certain order. When an image is passed through the convolutional layer and the first nonlinear layer, the calculated values will propagate through the Pooling layer. Set of 3 layers Convolutional layer + Nonlinear layer + Pooling layer can be repeated many times in the network. Then, it is propagated through the Fully connected layer and Softmax to calculate the probability and decide which object it is.

**The basic layers in CNN include:**

- **Convolutional layer:** This is the first and most important layer in CNN. Any computation (convolution) will be performed in this layer. In image classification, this layer is used to detect specific features of an image such as angle, edge, color or even texture of the image. The objects in this layer are the input matrices, filters, receptive fields, and feature maps. The filters will slide through each position on the image from left to right and top to bottom to calculate the convolution between the image at this potion and the receptive field (the area where neurons can recognize and make decisions) and generate a new matrix ( feature map). Convolution is performed at the positions passed by the filter and is determined by the formula (convolution symbol: $\otimes$) [71]

$$\boldsymbol{Y} = \boldsymbol{X} \otimes \boldsymbol{W}, \tag{1}$$

  where $\boldsymbol{X}$ is the input matrix, $\boldsymbol{W}$ is the kernel. For each element $x_{ij}$ in matrix $\boldsymbol{X}$, a matrix (calculated from the upper left corner) is obtained with the size equal to the size of the kernel with the element $x_{ij}$ centered, called matrix $\boldsymbol{A}$. Then, the sum of the elements of the element-wise calculation of matrix $\boldsymbol{A}$ with matrix $\boldsymbol{W}$ is computed, and stores the result in matrix $\boldsymbol{Y}$.

- **Nonlinear layer:** Some commonly used nonlinear activation functions are ELU (Exponential Linear Unit) [72] and ReLU (Rectified Linear Units) [73]. However, ELU is the activation function commonly used in CNN today. Before 2010, neural networks mainly used sigmoid and tanh functions. The ReLU activation function has the form $f(x) = max(0, x)$ for input values, placed right after the convolutional layer. Basically, this function will assign negative activation values equal to zero, thereby increasing the nonlinearity of the model and the entire network without affecting the convolutional layer. The ELU [72] function has the form as (3.2) with $\alpha > 0$

$$f(z) = \begin{cases} x & \text{if} \quad x \geqslant 0, \\ \alpha(\exp(x) - 1) & \text{if} \quad x < 0. \end{cases} \tag{2}$$

  There are many different activation functions. However, within the limited scope of this paper, we used the ELU activation function as recommended by some other authors [72,73]. The same goes for the Batch Normalization we used as recommended in [74–76].

- **Pooling layer:** After the activation function it will usually be the polling layer, also called subsampling or downsampling layer. The purpose of this layer is to reduce the size of the input matrix but still retain the outstanding inherent feature in the input matrix. There are two popular types of pooling layers: Max pooling and Average pooling.

- **Fully connected layer:** It is the last layer in the CNN model, used to connect neurons in two layers together (the neurons in the back layer are fully connected with the neurons in the front it). This layer has the function of converting the feature matrix in the previous layer into an N-dimensional vector (N - the number of classes to classified) containing the probabilities of the objects that need to be predicted. Finally, the CNN model training process is to have a loss function to calculate errors and accuracy for validation, as well as to use a backpropagation algorithm for the weight update process.

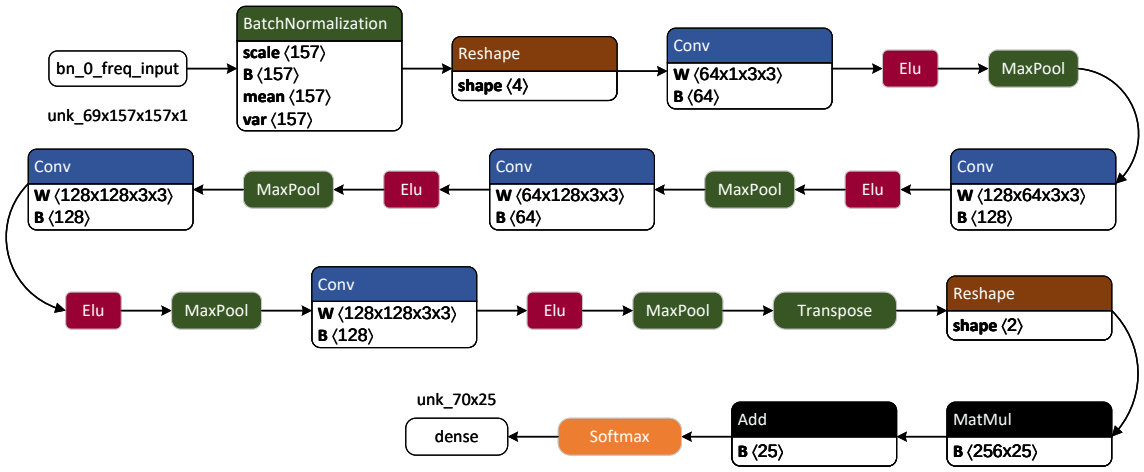


Figure 2: Configuration of CNN model with 157 parameters for identification

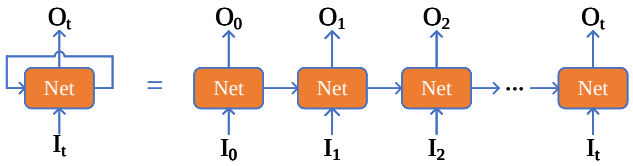


Figure 3: Recurrent neural networks

Table 3 is the CNN parameters used for training and test in the classification. In the case of classification of *Cheo* and *Quanho*, the output number of the model is equal to 2. If it is to identify *Cheo* or *Quanho* tunes, the output number of the model is 25 (25 is the number of tunes that need to be identified as which tune of *Cheo* or *Quanho*). The detailed information about the configuration of the CNN is shown in Figure 2. Netron [77], which is a viewer for neural network, deep learning, and machine learning models, was used to draw the configurations of CNN, LTSM and CRNN in Figures 2, 4 and 5.

Table 3: Network parameters of CNN for Training and Test in the classification

**Model: "sequential"**

| Layer (type) | Sets of parameter S1 (157 parameters) | | Sets of parameter S2 (183 parameters) | |
|---|---|---|---|---|
| | Output Shape | Param # | Output Shape | Param # |
| bn_0_freq (BatchNormalization) | (None, 157, 157, 1) | 628 | (None, 183, 183, 1) | 732 |
| conv1 (Conv2D) | (None, 157, 157, 64) | 640 | (None, 183, 183, 64) | 640 |
| bn1 (BatchNormalization) | (None, 157, 157, 64) | 256 | (None, 183, 183, 64) | 256 |
| elu (ELU) | (None, 157, 157, 64) | 0 | (None, 183, 183, 64) | 0 |
| pool1 (MaxPooling2D) | (None, 78, 78, 64) | 0 | (None, 91, 91, 64) | 0 |
| dropout (Dropout) | (None, 78, 78, 64) | 0 | (None, 91, 91, 64) | 0 |
| conv2 (Conv2D) | (None, 78, 78, 128) | 73856 | (None, 91, 91, 128) | 73856 |
| bn2 (BatchNormalization) | (None, 78, 78, 128) | 512 | (None, 91, 91, 128) | 512 |
| elu_1 (ELU) | (None, 78, 78, 128) | 0 | (None, 91, 91, 128) | 0 |
| pool2 (MaxPooling2D) | (None, 39, 39, 128) | 0 | (None, 45, 45, 128) | 0 |
| dropout_1 (Dropout) | (None, 39, 39, 128) | 0 | (None, 45, 45, 128) | 0 |
| conv3 (Conv2D) | (None, 39, 39, 128) | 147584 | (None, 45, 45, 128) | 147584 |
| bn3 (BatchNormalization) | (None, 39, 39, 128) | 512 | (None, 45, 45, 128) | 512 |
| elu_2 (ELU) | (None, 39, 39, 128) | 0 | (None, 45, 45, 128) | 0 |
| pool3 (MaxPooling2D) | (None, 19, 19, 128) | 0 | (None, 22, 22, 128) | 0 |
| dropout_2 (Dropout) | (None, 19, 19, 128) | 0 | (None, 22, 22, 128) | 0 |
| conv4 (Conv2D) | (None, 19, 19, 128) | 147584 | (None, 22, 22, 128) | 147584 |
| bn4 (BatchNormalization) | (None, 19, 19, 128) | 512 | (None, 22, 22, 128) | 512 |
| elu_3 (ELU) | (None, 19, 19, 128) | 0 | (None, 22, 22, 128) | 0 |
| pool4 (MaxPooling2D) | (None, 6, 6, 128) | 0 | (None, 11, 11, 128) | 0 |
| dropout_3 (Dropout) | (None, 6, 6, 128) | 0 | (None, 11, 11, 128) | 0 |
| Conv5 (Conv2D) | (None, 6, 6, 64) | 73792 | (None, 11, 11, 64) | 73792 |
| Bn5 (BatchNormalization) | (None, 6, 6, 64) | 256 | (None, 11, 11, 64) | 256 |
| elu_4 (ELU) | (None, 6, 6, 64) | 0 | (None, 11, 11, 64) | 0 |
| pool5 (MaxPooling2D) | (None, 1, 1, 64) | 0 | (None, 5, 5, 64) | 0 |
| dropout_4 (Dropout) | (None, 1, 1, 64) | 0 | (None, 5, 5, 64) | 0 |
| flatten (Flatten) | (None, 64) | 0 | (None, 1600) | 0 |
| dense (Dense) | (None, 2) | 130 | (None, 2) | 3202 |
| | Total params: 446,262<br>Trainable params: 444,924<br>Non-trainable params: 1,338 | | Total params: 449,438<br>Trainable params: 448,048<br>Non-trainable params: 1,390 | |

Table 4: Network parameters of LSTM for training and test in classification

**Model: "LSTM"**

| Layer (type) | LSTM-1 | | | | LSTM-2 | | | |
|---|---|---|---|---|---|---|---|---|
| | 157 parameters | | 183 parameters | | 157 parameters | | 183 parameters | |
| | Output Shape | Param # | Output Shape | Param # | Output Shape | Param # | Output Shape | Param # |
| bn_0_freq (BatchNormalization) | (None, 157, 157) | 628 | (None, 183, 183) | 732 | (None, 157, 157) | 628 | (None, 183, 183) | 732 |
| lstm (LSTM) | (None, 157, 256) | 423936 | (None, 183, 256) | 450560 | (None, 157, 256) | 423936 | (None, 183, 256) | 450560 |
| dropout (Dropout) | (None, 157, 256) | 0 | (None, 183, 256) | 0 | (None, 157, 256) | 0 | (None, 183, 256) | 0 |
| lstm_1 (LSTM) | (None, 256) | 525312 | (None, 256) | 525312 | (None, 512) | 1574912 | (None, 512) | 1574912 |
| dropout_1 (Dropout) | (None, 256) | 0 | (None, 256) | 0 | (None, 512) | 0 | (None, 512) | 0 |
| dense (Dense) | (None, 128) | 32896 | (None, 128) | 32896 | (None, 128) | 65664 | (None, 128) | 65664 |
| activation (Activation) | (None, 128) | 0 | (None, 128) | 0 | (None, 128) | 0 | (None, 128) | 0 |
| dropout_2 (Dropout) | (None, 128) | 0 | (None, 128) | 0 | (None, 128) | 0 | (None, 128) | 0 |
| dense_1 (Dense) | (None, 25) | 3225 | (None, 25) | 3225 | (None, 25) | 3225 | (None, 25) | 3225 |
| | Total params: 985,997<br>Trainable params: 985,683<br>Non-trainable params: 314 | | Total params: 1,012,725<br>Trainable params: 1,012,359<br>Non-trainable params: 366 | | Total params: 2,068,365<br>Trainable params: 2,068,051<br>Non-trainable params: 314 | | Total params: 2,095,093<br>Trainable params: 2,094,727<br>Non-trainable params: 366 | |

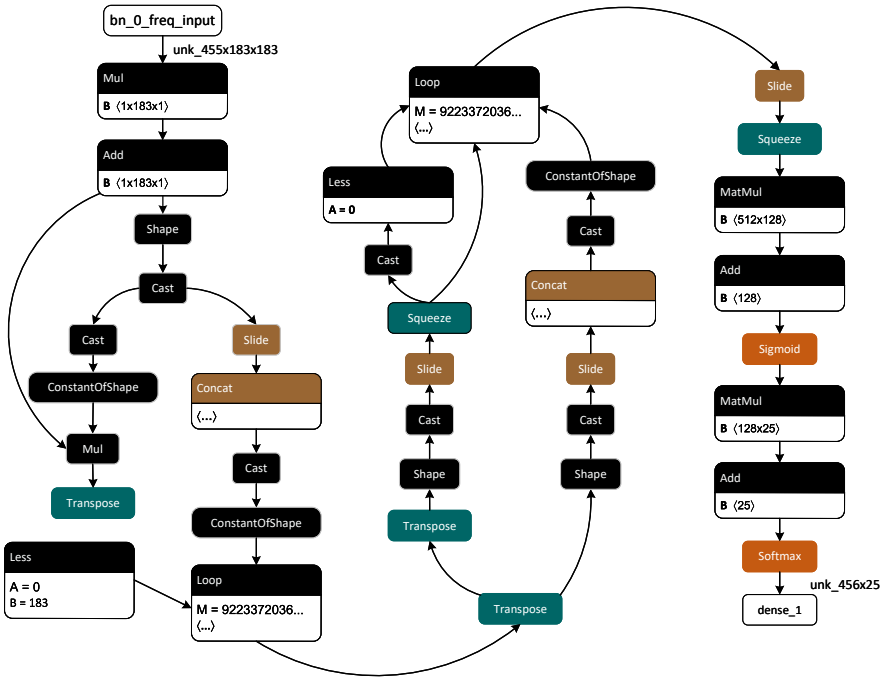


Figure 4: Configuration of LSTM model with 183 parameters for identification

Table 5: Network parameters of CRNN for training and test in the classification

**Model: "CRNN"**

| Layer (type) | CRNN-1 | | | | CRNN-2 | | | | Connected to |
|---|---|---|---|---|---|---|---|---|---|
| | 157 parameters | | 183 parameters | | 157 parameters | | 183 parameters | | |
| | Output Shape | Param # | Output Shape | Param # | Output Shape | Param # | Output Shape | Param # | |
| input_1 (InputLayer) | (None, 157, 157, 1) | | (None, 183, 183, 1) | | (None, 157, 157, 1) | 0 | (None, 183, 183, 1) | 0 | |
| conv_1 (Conv2D) | (None, 157, 157, 64) | 640 | (None, 183, 183, 64) | 640 | (None, 157, 157, 64) | 640 | (None, 183, 183, 64) | 640 | input_1[0][0] |
| conv_2 (Conv2D) | (None, 157, 157, 128) | 73856 | (None, 183, 183, 128) | 73856 | (None, 157, 157, 128) | 73856 | (None, 183, 183, 128) | 73856 | conv_1[0][0] |
| conv_3 (Conv2D) | (None, 157, 157, 256) | 295168 | (None, 183, 183, 256) | 295168 | (None, 157, 157, 256) | 295168 | (None, 183, 183, 256) | 295168 | conv_2[0][0] |
| bn_3 (BatchNormalization) | (None, 157, 157, 256) | 1024 | (None, 183, 183, 256) | 1024 | (None, 157, 157, 256) | 1024 | (None, 183, 183, 256) | 1024 | conv_3[0][0] |
| maxpool_3 (MaxPooling2D) | (None, 78, 78, 256) | 0 | (None, 91, 91, 256) | 0 | (None, 78, 78, 256) | 0 | (None, 91, 91, 256) | 0 | bn_3[0][0] |
| conv_4 (Conv2D) | (None, 78, 78, 256) | 590080 | (None, 91, 91, 256) | 590080 | (None, 78, 78, 256) | 590080 | (None, 91, 91, 256) | 590080 | maxpool_3[0][0] |
| conv_5 (Conv2D) | (None, 78, 78, 512) | 1180160 | (None, 91, 91, 512) | 1180160 | (None, 78, 78, 256) | 590080 | (None, 91, 91, 256) | 590080 | conv_4[0][0] |
| bn_5 (BatchNormalization) | (None, 78, 78, 512) | 2048 | (None, 91, 91, 512) | 2048 | (None, 78, 78, 256) | 1024 | (None, 91, 91, 256) | 1024 | conv_5[0][0] |
| maxpool_5 (MaxPooling2D) | (None, 39, 39, 512) | 0 | (None, 45, 45, 512) | 0 | (None, 39, 39, 256) | 0 | (None, 45, 45, 256) | 0 | bn_5[0][0] |
| conv_6 (Conv2D) | (None, 39, 39, 512) | 2359808 | (None, 45, 45, 512) | 2359808 | (None, 39, 39, 128) | 295040 | (None, 45, 45, 128) | 295040 | maxpool_5[0][0] |
| conv_7 (Conv2D) | (None, 39, 39, 512) | 2359808 | (None, 45, 45, 512) | 2359808 | (None, 39, 39, 64) | 73792 | (None, 45, 45, 64) | 73792 | conv_6[0][0] |
| bn_7 (BatchNormalization) | (None, 39, 39, 512) | 2048 | (None, 45, 45, 512) | 2048 | (None, 39, 39, 64) | 256 | (None, 45, 45, 64) | 256 | conv_7[0][0] |
| reshape (Reshape) | (None, 39, 19968) | 0 | (None, 45, 23040) | 0 | (None, 39, 2496) | 0 | (None, 45, 2880) | 0 | bn_7[0][0] |
| fc_9 (Dense) | (None, 39, 128) | 2556032 | (None, 45, 128) | 2949248 | (None, 39, 64) | 159808 | (None, 45, 64) | 184384 | reshape[0][0] |
| lstm_10 (LSTM) | (None, 39, 128) | 131584 | (None, 45, 128) | 131584 | (None, 39, 64) | 33024 | (None, 45, 64) | 33024 | fc_9[0][0] |
| lstm_10_back (LSTM) | (None, 39, 128) | 131584 | (None, 45, 128) | 131584 | (None, 39, 64) | 33024 | (None, 45, 64) | 33024 | fc_9[0][0] |
| add (Add) | (None, 39, 128) | 0 | (None, 45, 128) | 0 | (None, 39, 64) | 0 | (None, 45, 64) | 0 | lstm_10[0][0]<br>lstm_10_back[0][0] |
| lstm_11 (LSTM) | (None, 39, 128) | 131584 | (None, 45, 128) | 131584 | (None, 39, 64) | 33024 | (None, 45, 64) | 33024 | add[0][0] |
| lstm_11_back (LSTM) | (None, 39, 128) | 131584 | (None, 45, 128) | 131584 | (None, 39, 64) | 33024 | (None, 45, 64) | 33024 | add[0][0] |
| concatenate (Concatenate) | (None, 39, 256) | 0 | (None, 45, 256) | 0 | (None, 39, 128) | 0 | (None, 45, 128) | 0 | lstm_11[0][0]<br>lstm_11_back[0][0] |
| dropout2 (Dropout) | (None, 39, 256) | 0 | (None, 45, 256) | 0 | (None, 39, 128) | 0 | (None, 45, 128) | 0 | concatenate[0][0] |
| flatten (Flatten) | (None, 9984) | 0 | (None, 11520) | 0 | (None, 4992) | 0 | (None, 5760) | 0 | dropout2[0][0] |
| dense (Dense) | (None, 512) | 5112320 | (None, 512) | 5898752 | (None, 512) | 2556416 | (None, 512) | 2949632 | flatten[0][0] |
| dropout3 (Dropout) | (None, 512) | 0 | (None, 512) | 0 | (None, 512) | 0 | (None, 512) | 0 | dense[0][0] |
| dense_1 (Dense) | (None, 2) | 1026 | (None, 2) | 1026 | (None, 2) | 1026 | (None, 2) | 1026 | dropout3[0][0] |
| | Total params: 15,060,354<br>Trainable params: 15,057,794<br>Non-trainable params: 2,560 | | Total params: 16,240,002<br>Trainable params: 16,237,442<br>Non-trainable params: 2,560 | | Total params: 4,770,306<br>Trainable params: 4,769,154<br>Non-trainable params: 1,152 | | Total params: 5,188,098<br>Trainable params: 5,186,946<br>Non-trainable params: 1,152 | | |

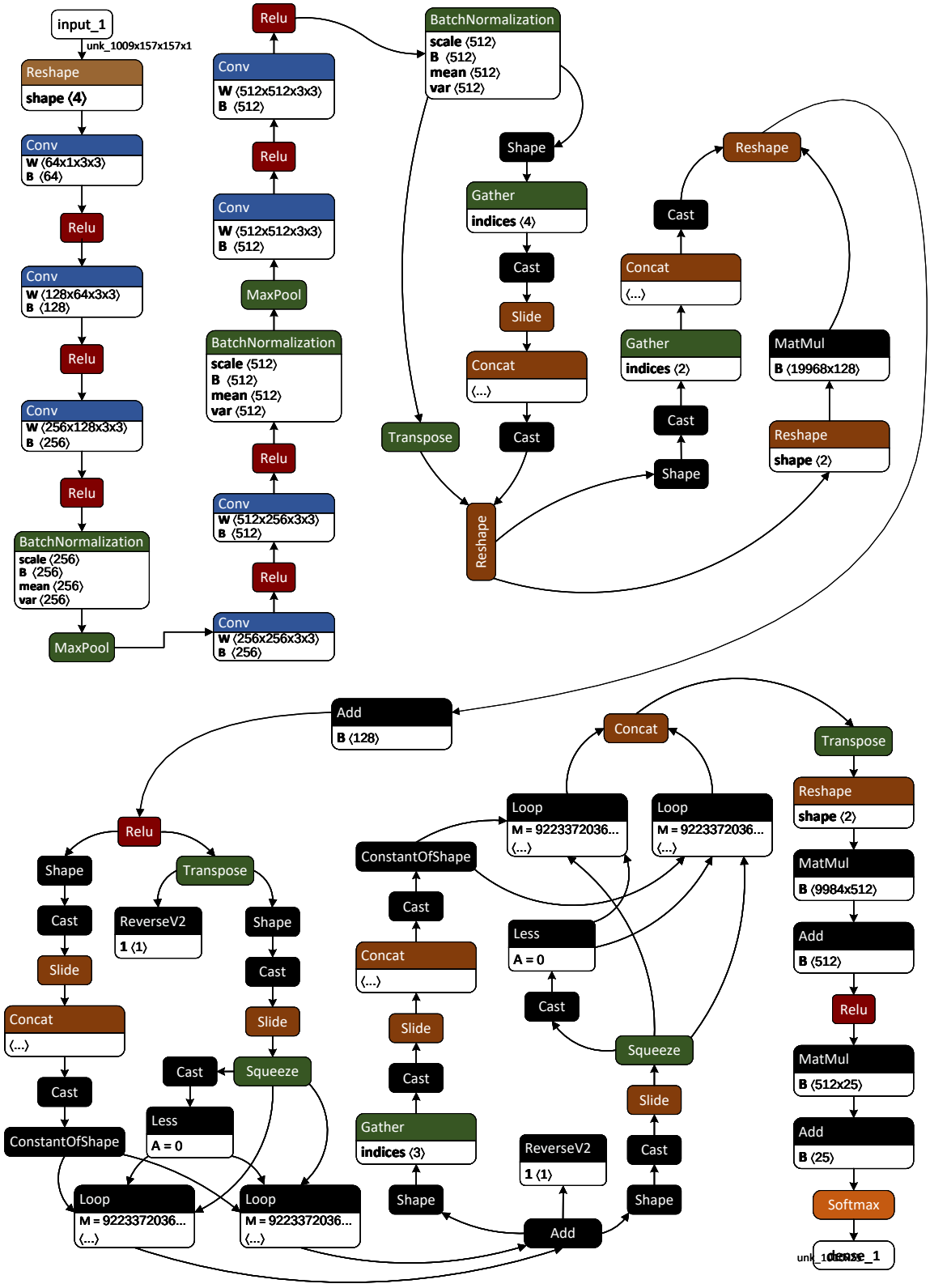


Figure 5: Configuration of CRNN model with 157 parameters for identification

The architecture of the CNN used in this paper belongs to the deep CNN, which has had good results in our other study on emotion recognition for Vietnamese spoken [78]. The common point of the CNN model for emotion recognition and the CNN model for music classification and identification in our case is that the characteristic parameters for voice and music are converted into images. In other words, both emotion recognition and music classification and identification can be considered to be the image recognition problem. Therefore, in the framework of this paper, we would like to conduct our research on deep CNN for classification, identification of *Cheo* and *Quanho* data.

### 3.3.2. Long-Short term memory networks (LSTM)

Recurrent neural network (RNN) [57, 58] have a feedback loop from output to input by which the network has memory properties, inferring the next which is partly based on the previous one. However, the memory capacity of the RNN is inversely proportional to the distance [79, 80]. Figure 3 depicts an RNN with an input of $I_t$ and an output of $O_t$.

LSTM is an improvement of RNN, proposed in 1997 by Hochreiter & Schmidhuber [59] to overcome the limitation of the short-term memory capacity of RNN. Nowadays, LSTM has been effectively applied in many different fields, especially in Natural Language Processing (NLP) [81–85]. In our case, we tested 2 different LSTM models LSTM1 and LSTM2. These two models differ in the number of nodes of the LSTM and Dense layers as shown in Table 4 with the parameters and configurations of LSTM1 and LSTM2 networks used for training and test. Figure 4 is the configuration of the LSTM network for 183 parameters.

### 3.3.3. Convolutional recurrent neural network (CRNN)

CRNN [60,61] is a combination of CNN and RNN (LSTM), often used to build sequential data processing models such as audio signals, text strings, etc. The structure of CRNN used in document classification was first proposed by Duyu Tang et al. [86], then Zhen Zuo et al. applied it in image classification [87] and music transcription by Sigtia et al. in 2016 [88]. In [61], Choi et al. showed the effectiveness of CRNN in classifying music by genre. In our case, we tested 2 different CRNN models CRNN1 and CRNN2. As for the two models LSTM1 and LSTM2, the CRNN1 and CRNN2 models were selected among the CRNN models that we tested with different configurations and they have the better results. These two models differ in the number of nodes of the Convolution, LSTM and Dense layers as shown in Table 5 with the parameters and configurations of CRNN1 and CRNN2 networks used for training and test. Figure 5 is the configuration of the CRNN model for 157 parameters.

It can be seen in Figure 3, Figure 4, and Figure 5, the activation function used for the dense layer in all models is Softmax. In our models, the layers of all models use the Keras library, where the loss function is "categorical crossentropy" and the accuracy metric is "accuracy" defined in Keras.

## 3.4. Implementation

The experiment is conducted in two cases: *Cheo* and *Quanho* classification and identification of *Cheo* and *Quanho* tunes. In the classification, the data need to be classified as *Cheo* or *Quanho*. For identification, the data will be identified as which tune in *Cheo* or

Table 6: Summary of classification results for *Cheo* and *Quanho*

| Parameters | S1 (157 parameters) | | | | | S2 (183 parameters) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Models / Folds | CNN | CRNN1 | CRNN2 | LSTM1 | LSTM2 | CNN | CRNN1 | CRNN2 | LSTM1 | LSTM2 |
| CRV0 | 99.50 | 98.75 | 99.5 | 99.25 | 98.8 | 100 | 99.25 | 99.5 | 100 | 100.0 |
| CRV1 | 99.75 | 98.50 | 100.0 | 99.50 | 99 | 99.75 | 100 | 99.8 | 100 | 99.8 |
| CRV2 | 99.50 | 98.75 | 100.0 | 99.50 | 99.3 | 100 | 99.75 | 100.0 | 99.25 | 99.3 |
| CRV3 | 99.25 | 98.00 | 99.8 | 99.75 | 99.3 | 100 | 99.50 | 100 | 99.50 | 99.5 |
| CRV4 | 100 | 100 | 100.0 | 99.25 | 99.8 | 100 | 99.25 | 99.5 | 99.50 | 99.5 |
| CRV5 | 99.50 | 99.25 | 100.0 | 98.50 | 99.5 | 99.75 | 99.75 | 99.0 | 100 | 99.8 |
| CRV6 | 99.50 | 99.75 | 100.0 | 99.25 | 99.3 | 100 | 99.75 | 99.5 | 99.75 | 99.8 |
| CRV7 | 99.50 | 99.50 | 98.8 | 99.25 | 99.5 | 100 | 99.25 | 100 | 100 | 99.8 |
| CRV8 | 99.50 | 99.50 | 98.8 | 99.75 | 100 | 99.75 | 99.25 | 99.8 | 100 | 99.8 |
| Aver. Acc. (%) | **99.55** | 99.16 | 99.66 | 99.34 | 99.44 | **99.92** | 99.56 | 99.69 | 99.75 | 99.63 |
| Aver. Acc. Difference between CRNN2&CRNN1, LSTM2&LSTM1 (%) | | 0.5 | | 0.1 | | ~ | 0.13 | | **-0.12** | |
| Aver. Acc. Difference between 183 and 157 parameters (%) | | | | | | 0.37 | 0.41 | 0.03 | 0.41 | 0.19 |

Table 7: Summary of results for identifying 25 *Quanho* tunes

| Parameters | S1 (157 parameters) | | | | | S2 (183 parameters) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Models / Folds | CNN | CRNN1 | CRNN2 | LSTM1 | LSTM2 | CNN | CRNN1 | CRNN2 | LSTM1 | LSTM2 |
| CRV0 | 100 | 96.0 | 98.0 | 98.0 | 97.5 | 99.5 | 99.0 | 97.5 | 95.0 | 96.0 |
| CRV1 | 99.5 | 99.5 | 99.0 | 99.0 | 100 | 100 | 99.0 | 99.5 | 96.5 | 98.0 |
| CRV2 | 98.0 | 96.5 | 97.5 | 99.0 | 99.0 | 100 | 98.5 | 97.5 | 97.5 | 98.0 |
| CRV3 | 99.5 | 97.5 | 97.5 | 98.5 | 99.5 | 100 | 99.5 | 100 | 98.5 | 98.0 |
| CRV4 | 99.5 | 96.5 | 96.0 | 100 | 99.5 | 99.5 | 95.0 | 98.5 | 96.0 | 96.0 |
| CRV5 | 100 | 97.5 | 98.0 | 99.5 | 99.0 | 98.5 | 97.0 | 97.5 | 96.5 | 96.5 |
| CRV6 | 99.5 | 99.5 | 97.5 | 100 | 98.5 | 99.5 | 96.0 | 98.5 | 98.5 | 98.0 |
| CRV7 | 100 | 96.5 | 98.0 | 99.0 | 99.5 | 99.0 | 99.5 | 100 | 98.0 | 98.0 |
| CRV8 | 99.5 | 98.0 | 98.5 | 99.5 | 100 | 99.5 | 99.5 | 99.0 | 97.0 | 97.5 |
| Aver. Acc. (%) | **99.50** | 97.69 | 97.75 | 99.30 | 99.38 | **99.50** | 98.00 | 98.81 | 97.31 | 97.50 |
| Aver. Acc. Difference between CRNN2&CRNN1, LSTM2&LSTM1 (%) | | 0.06 | | 0.08 | | ~ | 0.81 | | 0.19 | |
| Aver. Acc. Difference between 183 and 157 parameters (%) | | | | | | 0 | 0.31 | 1.06 | **-1.99** | **-1.88** |

*Quanho*. The experiment data is divided into 10 parts, of which 1 is for testing, the other 9 are for training and validation using the cross-validation method.

The experiments were performed on the computer system with the following hardware configuration: Intel Core i7-8700@3.2GHz × 12 threads processor, 32GB RAM, 2 TB storage, NVIDIA GeForce RTX 2080Ti 11GB RAM, Ubuntu 19.10 operating system.

## 4. RESULTS AND DISCUSSION

The classification experiment was carried out on all datasets for 2 parameter sets. The purpose of this experiment is to classify two types of *Cheo* and *Quanho* folk songs. Table 6 shows the average classification accuracy with the parameter sets S1 and S2 on three models CNN, LSTM, and CRNN. In the identification test, the data input is the *Cheo*'s dataset or *Quanho*'s dataset to identify which tune of *Cheo* or *Quanho*. Table 7 is the average accuracy for identification on *Quanho*'s dataset, and the average accuracy for identification on 25 *Cheo* tunes is given in Table 8.

Table 6 is the average accuracy of the classification for *Cheo* and *Quanho*, Table 7 and Table 8 are the average accuracy of the identification for *Quanho* and *Cheo* tunes with three models CNN, LSTM, and CNN. With the same data set, the accuracy of classification and identification of 3 models CNN, LSTM and CRNN are higher than that of GMM model [31].

Table 8: Summary of results for identifying 25 *Cheo* tunes

| Parameters | S1 (157 parameters) | | | | | S2 (183 parameters) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Models / Folds | CNN | CRNN1 | CRNN2 | LSTM1 | LSTM2 | CNN | CRNN1 | CRNN2 | LSTM1 | LSTM2 |
| CRV0 | 96.5 | 93.0 | 94.0 | 97.0 | 95.0 | 98.5 | 94.5 | 95.5 | 95.0 | 95.5 |
| CRV1 | 97.5 | 97.0 | 94.5 | 95.5 | 95.5 | 97.0 | 95.0 | 95.0 | 94.0 | 93.5 |
| CRV2 | 97.5 | 96.5 | 94.0 | 97.5 | 96.0 | 97.5 | 95.5 | 95.5 | 98.5 | 95.0 |
| CRV3 | 94.5 | 93.5 | 96.5 | 96.0 | 95.5 | 97.5 | 95.5 | 95.5 | 96.0 | 97.0 |
| CRV4 | 97.5 | 91.5 | 97.0 | 95.5 | 98.0 | 98.0 | 94.0 | 97.5 | 96.5 | 95.0 |
| CRV5 | 97.0 | 94.5 | 96.0 | 96.0 | 97.0 | 98.0 | 95.5 | 96.0 | 98.5 | 97.5 |
| CRV6 | 94.0 | 93.5 | 96.5 | 96.5 | 97.0 | 97.0 | 95.0 | 97.0 | 95.5 | 97.0 |
| CRV7 | 99.0 | 91.5 | 92.5 | 96.0 | 93.5 | 98.0 | 97.0 | 97.0 | 97.5 | 97.0 |
| CRV8 | 95.5 | 97.5 | 97.5 | 97.0 | 98.0 | 97.5 | 97.5 | 95.0 | 97.0 | 97.0 |
| Aver. Acc. (%) | **96.56** | 94.44 | 95.56 | 96.25 | 96.31 | **97.67** | 95.63 | 96.06 | 96.68 | 96.13 |
| Aver. Acc. Difference between CRNN2&CRNN1, LSTM2&LSTM1 (%) | | 1.12 | | 0.06 | | ~ | 0.43 | | **-0.55** | |
| Aver. Acc. Difference between 183 and 157 parameters (%) | | | | | | 1.11 | 1.19 | 0.50 | 0.43 | **-0.18** |

Table 9: Difference in accuracy of the CNN model compared to the other two models

| Set of parameters | Classification (%) | | | Identification (%) | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | *Quanho & Cheo* | | | *Quanho tunes* | | | *Cheo tunes* | | |
| | CNN | MIN | Difference | CNN | MIN | Difference | CNN | MIN | Difference |
| S1 (157) | 99.55 | 99.16 | **0.39** | 99.50 | 97.69 | 1.81 | 96.56 | 94.44 | 2.12 |
| S2 (183) | 99.92 | 99.56 | 0.36 | 99.50 | 97.31 | **2.19** | 97.67 | 95.63 | 2.04 |

MIN = Minimum accuracy among CRNN1, CRNN2, LSTM1 and LSTM2

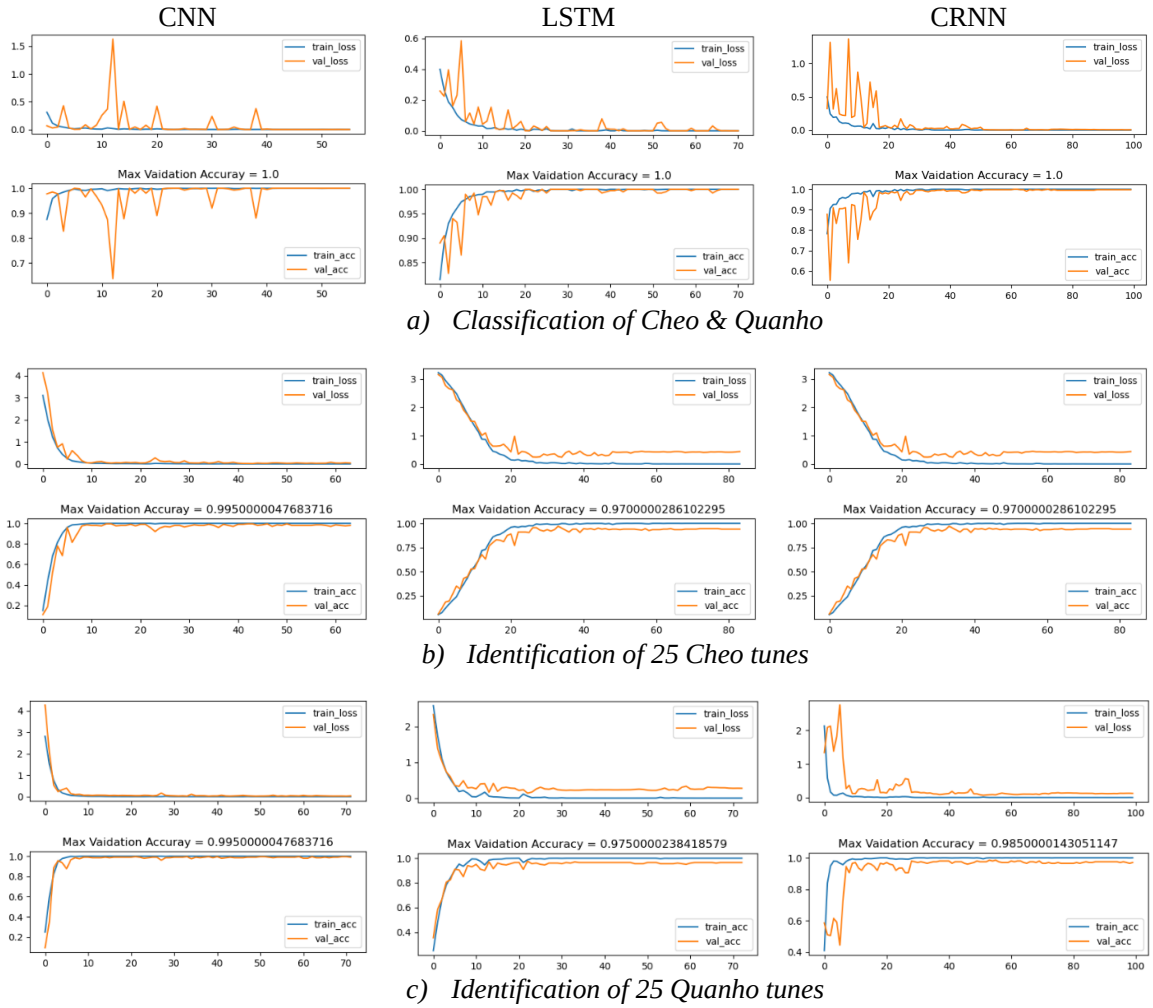


Figure 6: Examples of loss function and accuracy variation with respect to epoch for training and validation in the classification and identification cases.

In general, when increasing the number of parameters from 157 to 183, average accuracy increases for all models, except the two cases: LSTM1 and LSTM2 for the identification of 25 Quanho tunes, LSTM2 for the identification of of 25 Cheo tunes.

If we make comparisons in the same CRNN, CRNN2 always has higher accuracy than CRNN1. The LSTM2 network also always achieves higher accuracy than LSTM1 except in the two cases: identification of Cheo tunes with 183 parameters and classification for Cheo and Quanho with 183 parameters.

For the comparison between CRNN and LSTM models with the same set of parameters, we see the following: for classification of *Cheo* and *Quanho* with 157 parameters, the accuracy of CRNN is higher than that of LSTM, but with 183 parameters the accuracy of CRNN is lower than that of LSTM. For the identification of 25 *Quanho* tunes, with 157 parameters the accuracy of LSTM is higher than that of CRNN, but is lower than that of CRNN with 183 parameters. In the case of identifying the 25 *Cheo* tunes, the accuracy of LSTM is higher than that of CRNN for both sets of 157 and 183 parameters.

In our experiments, the classification and identification accuracy of the CNN model is higher than that of the LSTM and CRNN models. Table 9 is the difference in the accuracy of the CNN model compared with the other two models. It is the difference between the accuracy of the CNN model compared to the lowest accuracy of one of the other two models. If we only take out the most different case, the biggest difference in accuracy of the CNN model compared to the other two models is 0.39% for classification and 2.19% for identification (as shown in Table 9).

In our case, the inputs to the models are images where each image contains the feature corresponding to each tune. Of course, in the same image, there is a certain correlation between adjacent image elements, but this is not quite the same as the sequence of words in a meaningful sentence or the sequence of images that make up a video. If the nature of the LSTM (or RNN) network structure is to remember the previous one to infer the next one, this is not very efficient in this case. Maybe that's why the identification accuracy of LSTM and CRNN are slightly different from that of CNN.

The examples of loss function and accuracy variation with respect to epoch for training and validation in the classification case and identification case are shown in Figure 6. The training and validation results as shown in this Figure show that there is no overfitting phenomenon. If there is overfitting, there will be no relationship between loss, accuracy, and epoch as the results we had [89].

## 5. CONCLUSIONS

The paper presents our research results on the classification of *Cheo* and *Quanho* and the identification of *Cheo* and *Quanho* tunes. With deep neural network models such as CNN, LSTM and CRNN, the classification and identification accuracy are both higher than that of GMM models on the same data set. It is well known that for the classification problem or in machine learning in general, the more available data, the better the classification performance. In this case, the data augmentation performed by halving each tune also has a certain meaning to improve recognition performance.

## ACKNOWLEDGMENT

We would like to thank Faculty of Information Technology - Hung Yen University of Technology and Education, School of Information and Communication Technology - Hanoi University of Science and Technology, and Faculty of Information Technology - University of Transport and Communications for helping us complete this paper.

## REFERENCES

[1] P. Cunningham and S. J. Delany, "k-nearest neighbor classifiers," in *Multiple Classifier Systems*, 2007, vol. 34, no. 8, pp. 1–17.

[2] Y. Sazaki and A. Aramadhan, "Rock genre classification using k-nearest neighbor," in *Proceedings of The 1st International Conference on Computer Science and Engineering*, 2014, pp. 81–84.

[3] Z. Ghahramani, "An introduction to hidden markov models and bayesian networks," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 1, pp. 9–42, 2001.

[4] X. Shao, C. Xu, and M. S. Kankanhalli, "Unsupervised classification of music genre using hidden markov model," in *2004 IEEE International Conference on Multimedia and Expo (ICME)*, vol. 3, 2004, pp. 2023–2026.

[5] J. Reed and C. H. Lee, "A study on music genre classification based on universal acoustic models," in *Proceedings of the International Conference on Music Information Retrieval*, 2006, pp. 89–94.

[6] U. Bağcı and E. Erzin, "Boosting classifiers for music genre classification," in *International Symposium on Computer and Information Sciences*. Springer Berlin Heidelberg, 2005.

[7] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2013.

[8] K. Markov and T. Matsui, "Music genre and emotion recognition using gaussian processes," *IEEE Access*, vol. 2, pp. 688–697, 2014.

[9] A. Meng and J. Shawe-Taylor, "An investigation of feature models for music genre classification using the support vector classifier," in *Proceedings of the International Conference on Music Information Retrieval*, 2005, pp. 604–609.

[10] M. Li and R. Sleep, "Genre classification via an lz78-based string kernel," in *Proceedings of the International Conference on Music Information Retrieval*, 2005, pp. 252–259.

[11] Y. Wang, X. Lin, L. Wu, and W. Zhang, "Effective multi-query expansions: Collaborative deep networks for robust landmark retrieval," *IEEE Transactions on Image Processing*, vol. 26, pp. 1393–1404, 2017.

[12] L. Wu, Y. Wang, and L. Shao, "Cycle-consistent deep generative hashing for cross-modal retrieval," *IEEE Transactions on Image Processing*, vol. 28, pp. 1602–1612, 2019.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, vol. 25, no. 2, 2012, pp. 1097–110.

[14] B. Matityaho and M. Furst, "Neural network based model for classification of music type," in *Proceedings of the Convention of Electrical and Electronics Engineers in Israel*, March 1995, pp. 1–5.

[15] "Ismir conferences," [Online]. Available: http://ismir.net/conferences/.

[16] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.

[17] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "A survey of audio-based music classification and annotation," *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 303–319, April 2011.

[18] P. Knees and M. Schedl, "A survey of music similarity and recommendation from music context data," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 10, no. 1, pp. 1–21, 2013.

[19] C. D. Corrêa and F. A. Rodrigues, "A survey on symbolic data-based music genre classification," *Expert Systems with Applications*, vol. 60, pp. 190–210, 2016.

[20] B. L. Sturm, "A survey of evaluation in music genre recognition," in *International Workshop on Adaptive Multimedia Retrieval*. Springer, 2012, pp. 29–66.

[21] J.-J. Aucouturier and F. Pachet, "Representing musical genre: A state of the art," *Journal of New Music Research*, vol. 32, no. 1, pp. 83–93, 2003.

[22] A. Anglade, Q. Mary, R. Ramirez, and S. Dixon, "Genre classification using harmony rules induced from automatic chord transcriptions," in *Proceedings of the International Conference on Music Information Retrieval*, 2009, pp. 669–674.

[23] Y. Panagakis, C. Kotropoulos, and G. R. Arce, "Music genre classification using locality preserving non-negative tensor factorization and sparse representations," in *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, 2009, pp. 249–254.

[24] D. Ghosal and M. H. Kolekar, "Music genre recognition using deep neural networks and transfer learning," in *Interspeech*, 2018.

[25] H. Bahuleyan, "Music genre classification using machine learning techniques," *arXiv preprint*, 2018.

[26] S. Oramas *et al.*, "Multimodal deep learning for music genre classification," *Transactions of the International Society for Music Information Retrieval*, vol. 1, no. 1, pp. 4–21, 2018.

[27] A. Elbir and N. Aydin, "Music genre classification and music recommendation by using deep learning," *Electronics Letters*, vol. 56, no. 12, pp. 627–629, 2020.

[28] S. Vishnupriya and K. Meenakshi, "Automatic music genre classification using convolution neural network," in *International Conference on Computer Communication and Informatics (ICCCI)*. IEEE, 2018.

[29] "Zalo ai challenge," [Online]. Available: https://challenge.zalo.ai/.

[30] Q. H. Nguyen, T. T. T. Do, T. B. Chu, L. V. Trinh, D. H. Nguyen, C. V. Phan, T. A. Phan, D. V. Doan, H. N. Pham, B. P. Nguyen, and M. C. H. Chua, "Music genre classification using residual attention network," in *2019 International Conference on System Science and Engineering (ICSSE 2019)*, Dong Hoi, Vietnam, July 20-21 2019, pp. 115–119.

[31] T. C. Ba, T. V. Loan, and N. H. Quang, "Some new results on automatic identification of vietnamese folk songs cheo and quanho," *Journal of Computer Science and Cybernetics*, vol. 36, no. 4, pp. 325–345, 2020.

[32] C. B. Thanh, T. V. Loan, and N. H. Quang, "Automatic identification of some vietnamese folk songs," in *Proceedings of the 19th National Symposium of Selected ICT Problems*, Hanoi, Vietnam, 2016, pp. 92–97.

[33] ——, "Gmm for automatic identification of some quanho bac ninh folk songs," in *Proceedings of Fundamental and Applied IT Research (FAIR)*, Da Nang, Vietnam, 2017, pp. 416–421.

[34] ——, "Classification and identification of cheo and quanho bac ninh folk songs," in *Proceedings of Fundamental and Applied IT Research (FAIR)*, Hanoi, Vietnam, 2018, pp. 395–403.

[35] S. Nainan and V. Kulkarni, "Enhancement in speaker recognition for optimized speech features using gmm, svm and 1-d cnn," *International Journal of Speech Technology*, vol. 24, p. 809–822, 2021.

[36] D. T. L. Thuy, T. V. Loan, and N. H. Quang, "Gmm for emotion recognition of vietnamese," *Journal of Computer Science and Cybernetics*, vol. 33, no. 3, pp. 229–246, 2017.

[37] H. P. Ngoc, T. V. Loan, and N. H. Quang, "Automatic identification of vietnamese dialects," *Journal of Computer Science and Cybernetics*, vol. 32, no. 1, pp. 19–30, 2016.

[38] Librosa, "Python package for music and audio analysis," [Online]. Available: https://librosa.github.io/librosa.

[39] "Tonnetz," [Online]. Available: https://en.wikipedia.org/wiki/Tonnetz.

[40] F. D. Jevtić and R. T. Živaljević, "Generalized tonnetz and discrete abel-jacobi map," *arXiv preprint*, 2020.

[41] D.-N. Jiang *et al.*, "Music type classification by spectral contrast feature," in *Proceedings IEEE International Conference on Multimedia and Expo*, vol. 1, 2002.

[42] "Chroma feature," [Online]. Available: https://handwiki.org/wiki/Chroma_feature.

[43] T. Cho and J. P. Bello, "On the relative importance of individual components of chord recognition systems," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 2, pp. 477–492, 2013.

[44] M. Mauch and S. Dixon, "Simultaneous estimation of chords and musical context from audio," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1280–1289, 2009.

[45] T. Fujishima, "Real-time chord recognition of musical sound: A system using common lisp music," in *Proceedings of the International Computer Music Conference (ICMC)*, 1999, pp. 464–467.

[46] N. Jiang *et al.*, "Analyzing chroma feature types for automated chord recognition," in *Audio Engineering Society Conference: 42nd International Conference: Semantic Audio*. Audio Engineering Society, 2011.

[47] D. P. W. Ellis and G. E. Poliner, "Identifying cover songs with chroma features and dynamic programming beat tracking," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'07)*, vol. 4. IEEE, 2007.

[48] J. Serra *et al.*, "Chroma binary similarity and local alignment applied to cover song identification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 6, pp. 1138–1151, 2008.

[49] M. Müller, F. Kurth, and M. Clausen, "Audio matching via chroma-based statistical features," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, 2005.

[50] F. Kurth and M. Müller, "Efficient index-based audio matching," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 2, pp. 382–395, 2008.

[51] M. Müller, *Music Synchronization.* Springer, 2015.

[52] Y. Yu *et al.*, "Combining multi-probe histogram and order-statistics based lsh for scalable audio content retrieval," in *Proceedings of the 18th ACM International Conference on Multimedia*, 2010.

[53] Y. Yu, M. Crucianu, V. Oria, and L. Chen, "Local summarization and multi-level lsh for retrieving multi-variant audio tracks," in *Proceedings of the 17th ACM International Conference on Multimedia*, 2009.

[54] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference On Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[55] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

[56] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[57] D. Rumelhart, G. Hinton, and R. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, p. 533–536, 1986.

[58] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, 2020.

[59] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[60] G. Keren and B. Schuller, "Convolutional rnn: An enhanced model for extracting features from sequential data," in *2016 International Joint Conference on Neural Networks (IJCNN).* IEEE, 2016, pp. 3412–3419.

[61] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "Convolutional recurrent neural networks for music classification," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2017, pp. 2392–2396.

[62] K. Fukushima, "A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, pp. 193–202, 1980.

[63] Y. LeCun *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[64] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012.

[65] D. Cireşan and U. Meier, "Multi-column deep neural networks for offline handwritten chinese character classification," in *2015 International Joint Conference On Neural Networks (IJCNN)*. IEEE, 2015.

[66] N. Pelchat and C. M. Gelowitz, "Neural network music genre classification," *Canadian Journal of Electrical and Computer Engineering*, vol. 43, no. 3, pp. 170–173, 2020.

[67] K. Choi, G. Fazekas, M. Sandler, and J. Kim, "Auralisation of deep convolutional neural networks: Listening to learned features," in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015, p. 26–30.

[68] P. Chiliguano and G. Fazekas, "Hybrid music recommender using content-based and social information," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, p. 2618–2622.

[69] M. Matocha and S. K. Zieliński, "Music genre recognition using convolutional neural networks," *Advances in Computer Science Research*, vol. 14, pp. 125–142, 2018.

[70] D. T. L. Thuy, "Emotional recognition for vietnamese spoken," Ph.D. dissertation, Hanoi University of Science and Technology, 2019.

[71] "Bài 5 - gii thiu v x lý nh," [Online]. Available: https://nttuan8.com/bai-5-gioi-thieu-ve-xu-ly-anh/.

[72] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint*, 2015.

[73] E. G. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.

[74] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML'15 Proceedings of the 32nd International Conference on Machine Learning*, France, 2015, pp. 448–456.

[75] "Normalization techniques in deep neural networks," [Online]. Available: https://www.programmersought.com/article/41731094913/.

[76] "Normalization techniques in deep neural networks," [Online]. Available: https://medium.com/techspace-usict/normalization-techniques-in-deep-neural-networks-9121bf100d8.

[77] "Netron - model visualization tool," [Online]. Available: https://github.com/lutzroeder/netron.

[78] D. T. L. Thuy, L. T. Van, and Q. N. Hong, "Deep convolutional neural networks for emotion recognition of vietnamese," *International Journal of Machine Learning and Computing*, vol. 10, no. 5, 2020.

[79] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen netzen," Ph.D. dissertation, Technische Universität München, 1991.

[80] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[81] T. Mikolov *et al.*, "Recurrent neural network based language model," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[82] ——, "Extensions of recurrent neural network language model," in *2011 IEEE International Conference On Acoustics, Speech And Signal Processing (ICASSP)*. IEEE, 2011.

[83] I. Sutskever, J. Martens, and G. E. Hinton, "Generating text with recurrent neural networks," in *ICML*, 2011.

[84] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*, 2014.

[85] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of the 31st International Conference on Machine Learning, PMLR*, vol. 32, no. 2, 2014, pp. 1764–1772.

[86] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, p. 1422–1432.

[87] Z. Zuo, B. Shuai, G. Wang *et al.*, "Convolutional recurrent neural networks: Learning spatial dependencies for image representation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, p. 18–26.

[88] S. Sigtia, E. Benetos, and S. Dixon, "An end-to-end neural network for polyphonic piano music transcription," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 5, pp. 927–939, 2016.

[89] "Overfitting in cnn - how to treat overfitting in convolutional neural networks," [Online]. Available: https://www.analyticsvidhya.com/blog/2020/09/overfitting-in-cnn-show-to-treat-overfitting-in-convolutional-neural-networks.

*Received on March 24, 2021*
*Accepted on February 02, 2022*