

# Recognition of music symbol notation using convolutional neural network

Ciara Setyo, Gede Putra Kusuma

Computer Science Department, Binus Graduate Program–Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia

## Article Info

### Article history:

Received Aug 11, 2023

Revised Oct 1, 2023

Accepted Nov 4, 2023

### Keywords:

Convolutional neural network

Data augmentation

Deep learning

Musical notes

Optical music recognition

## ABSTRACT

Musical notation is one thing that needs to be learned to play music. This notation has an important role in music because it can help in visualizing instructions for playing musical instruments and singing. Unfortunately, musical symbols that are commonly written in musical notation are difficult for beginners who have just started learning music. This research proposed a solution to create an optical music recognition (OMR) using a deep learning model to classify musical notes more accurately with some of the latest convolutional neural network (CNN) architectures. The research was carried out by implementing vision transformer (ViT), CoAtNet-0, and ConvNeXt-Tiny architecture. The training process was also combined with data augmentation to provide more information for the model to learn. Then the accuracy results of each model were compared to find out the best model for the OMR solution in this research. This experiment uses the Andrea dataset and Attwenger dataset which both get the best result by using the augmentation method and ConvNeXt-Tiny as the model. The best accuracy for the Andrea dataset is 98.15% and for the Attwenger dataset is 98.43%.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



## Corresponding Author:

Ciara Setyo

Computer Science Department, Binus Graduate Program–Master of Computer Science, Bina Nusantara University

Jakarta-11480, Indonesia

Email: ciara.setyo@binus.ac.id

## 1. INTRODUCTION

Music has become a cultural phenomenon that exists in every country and continues to evolve. Music can be used to express many things such as local identities, traditions, moral values, beliefs, aspirations, and social interactions [1]. With a quite diverse use of music, it has become a hobby that many people have and a common thing to learn in schools [2]. Music symbol notation is a knowledge that must be learned to play music. This notation is important in music because it can help visualize instructions of pitch scale signature and duration for each music part. With this use of musical notation, any musician can plan their musical performance and do collaborative musical activities more easily [3]. Unfortunately, musical notes that are written in musical notation are difficult for most people who just started learning music. This is because in reading musical notes, people need to process two separate pieces of information. The first information comes from the music symbol which represents the duration of a certain pitch scale. The second information comes from the staff line that represents the pitch scale itself. Therefore, ordinary people who are generally only used to reading in alphabetical and numeric form will have difficulty in reading musical symbol notation [4]. To overcome these difficulties, several ways can be used. One of them is to use a solmization system. With a solmization system, musical notes will be changed into forms that are easier to understand, such as in alphabetic or numeric forms [5]. But of course, this will take quite a long time if done

manually, especially for songs that have a long duration or high complexity. Therefore, we need technology that can be used to do solmization to assist in reading musical notes.

The solution that can be offered to this problem is optical music recognition (OMR), a field of research searching for a method that helps computers read musical notation documents [6]. OMR can be implemented using deep learning algorithms that have been proven to have good effectiveness in recognizing various image data. Even though image classification like OMR can be done using only machine learning methods, in general, these methods are still not as good as deep learning methods. This is because deep learning can automatically create new features from existing features in the training dataset. With this advantage, deep learning does not need to do feature extraction manually and the training process can be faster than traditional machine learning methods [7]. In addition, even though the image of musical notation looks simple briefly, most of its musical symbols have a two-dimensional nature. This means that the classifier model must learn the image from more than one point of view. In this case, the model needs to learn the shape of each music symbol as well as the vertical position of the staff line to get complete information from the music notation [8]. With these conditions, the use of only machine learning methods might give low accuracy results and difficulties in adapting to new data that may be added in the future.

Convolutional neural network (CNN) is a deep learning method that is often used in various applications such as computer vision and object recognition. CNN is used quite often because of its ability to find important features of an object without human supervision or intervention. Because CNN is a deep neural network, CNN generally has a feature where there are several hidden layers in it, so it is very suitable for performing complex computations, especially in image pattern recognition problems [9]. Therefore, CNN is also quite widely used in OMR field classification research and most of this research gets an accuracy of around 80% to 90%. Some of the previous OMR research using CNN is shown in Table 1.

Table 1. Previous OMR research using CNN

Reference	Model	Augmentation	Dataset	Total class	Total data	Accuracy /F1-Score
[10]	GoogleNet	No	Handwritten online music symbol (HOMUS)	32	15,200	94.60%
[11]	GoogleNet	Yes	HOMUS	32	15,200	96.01%
[12]	CNN custom	Yes	HOMUS writer	32	15,200	97.20%
[12]	CNN custom	Yes	HOMUS general	32	15,200	96.22%
[12]	CNN custom	Yes	Capitan score	30	10,230	99.16%
[12]	CNN custom	Yes	Capitan stroke	30	10,230	95.63%
[13]	CNN custom	No	Collected by researcher	30	10,150	96.40%
[14]	Visual geometry group network-16 (VGGNet-16)	Yes	Made with Lilypond	60	22,200	99.00%
[15]	CNN custom	No`	International Conference on Document Analysis and Recognition (ICDAR) 2013 Competition on music scores staff removal contest	2	400,000	99.29%
[16]	CNN Custom + K-nearest neighbor (KNN)	No	HOMUS	32	15,200	94.66%
[17]	Inception residual network-V2 (Inception ResNet-V2)	No	Collected by researcher	32	15,258	98.00%
[18]	CNN custom	No	Attwenger	90	4,286	95.56%
[19]	CNN custom	No	Collected by researcher	37	14,373	97.00%
[19]	CNN custom	No	Collected by researcher	14	14,373	93.80%
[19]	CNN custom	No	Collected by researcher	157	14,373	91.80%
[20]	Residual network-101 (ResNet-101)	Yes	Attwenger	2	13,237	99.02%
[20]	ResNet-101	Yes	OMR dataset	5	11,249	81.47%
[21]	CNN custom	Yes	HOMUS	32	15,200	95.74%
[22]	CNN custom	Yes	Collected from muse score	13	1,625	80.00%
[23]	CNN custom	No	Collected by researcher	7	4,094	92.00%

Much other OMR research has similar good accuracy results using CNN. Unfortunately, all this research still has some flaws that must be fixed. The first flaw is related to the dataset used in the research. Most of the OMR research indeed gets good classification accuracy results, but the dataset for the research itself often does not have too much data variety. This lack of data variety can cause serious problems when the model must predict images that have different styles, viewpoints, backgrounds, deformation, illumination, and other things that can disturb model prediction. However, this problem can be solved by using several data augmentation methods that can give the dataset more data variety and amount [24]. There is still much

previous OMR research that does not use data augmentation for creating the classification model even though it might improve their accuracy results a lot. Because of this flaw, this research desires to try using a combination of several data augmentation methods for improving musical symbol classification model accuracy.

The other flaw from the previous OMR research is related to the CNN architecture which is used for making a classification model. Most of this previous OMR research was using a custom CNN architecture that has not been proven to be good. This makes most research that uses custom CNN architecture have difficulty beating the OMR research that uses official CNN architecture. An example can be seen from the previously mentioned research where the research with the HOMUS dataset gets 96.01% accuracy by using GoogleNet [11] which is bigger than the one that uses custom architecture which gets 95.56% accuracy [21]. It is proven more with the research that uses a lesser amount of data and classes but has quite low accuracy which is 80% [22] if compared to the research that uses GoogleNet [11]. Based on this information, it is better to try the currently developed CNN architecture that has been available and has proven good in another research field. One source of research that can be used to seek the development of this CNN architecture is ImageNet dataset classification research. This research is one of CNN's studies which is quite popular and has many developments currently because it uses complex image data with a total of 1000 classes and a total of 1.2 million images. With this information, this research also desires to implement several new CNN architectures from ImageNet classification research together with the data augmentation method. The architectures that are used in this research are vision transformer (ViT), CoAtNet-0, and ConvNeXt-Tiny which have never been used in other OMR classification research. These architectures have already proven good, especially in ImageNet dataset classification where ViT gets 88.5% accuracy [25], CoAtNet-0 gets 83.9% accuracy [26] and ConvNeXt-Tiny gets 82.1% accuracy [27]. Then the classification results of these architectures are evaluated and compared to determine the best architecture for classifying musical symbol notation. Inception-V3 was also used as the main benchmark in this research because Inception architecture has been proven to have quite good results in much OMR research. For the dataset, this research used several datasets from previous research that are available online and still have room for improvement. The dataset is the Attwenger dataset from research 18 and the Andrea dataset from research [22].

## 2. THEORY AND METHOD

### 2.1. Inception-V3

Inception-V3 is the result of GoogleNet architecture improvement that aims to make Inception architecture with better computation efficiency and fewer parameters while keeping its advantage where each layer in the architecture can extract information from various spatial sizes. In Inception-V3, the architecture is still divided into several stacked blocks like GoogleNet. Each block of this architecture has several layers such as the convolutional layer, pooling layer, and others. However, some of these layers run in parallel during the learning process and are concatenated at the end of each block to combine the learning results of each layer into a vector [28].

There were several modifications made to GoogleNet to make the Inception-V3 born. The main modification is to do factorization to the several layers to reduce the number of parameters produced. This is done by converting a  $5 \times 5$  convolutional layer into a smaller convolution, which is two  $3 \times 3$  convolutional layers. Then the factorization is continued by changing all the  $3 \times 3$  convolutional layers to become asymmetric convolutions consisting of  $1 \times 3$  convolutional layers and  $3 \times 1$  convolutional layers that are stacked vertically. The result of this factorization is the first version of Inception in Inception-V3 which is executed 5 times in its block. Afterward, it is followed by a second version of Inception which is the same as the first version, but it changes the  $1 \times 3$  and  $3 \times 1$  convolutional layers to  $1 \times n$  and  $n \times 1$ . The recommended number  $n$  is 7 and this version is executed 4 times in its block. Then it is followed by the last Inception version which is the result of convolution layer factorization  $n \times n$  with  $n=3$ . However, unlike the second Inception version, the factorization results in the last Inception version are stacked horizontally. The last Inception version has a different purpose from other Inceptions, which is to support the capabilities of Inception-V3 for learning high-dimensional representation [28].

The other modification is where only one auxiliary layer is used to become the model regularizer. There is also an additional block called grid size reduction which is applied at every block change of each Inception version. This block is used to perform downsizing which is less expensive but still an efficient network for the Inception-V3 [28].

### 2.2. Vision transformer (ViT)

Vision transformer (ViT) is an architecture that uses transformers directly into image patch sequences to provide good classification accuracy results. Classification using ViT begins by dividing the image into a set of image patches that have the same size. Then that image patch sequence is used as

embedding. Position encoding is also carried out on each of these image patches. Then each image patch is converted into a vector and this vector is multiplied by an embedding matrix. After the results are obtained, these results go into the transformer encoder together with the positional embedding obtained from the previous positional encoding. After the transformer encoder is passed, the information goes to the fully connected layer that acts as the output layer [25].

Some operations must be done in the transformer encoder. This operation starts with multi-head self-attention where the model looks at each image patch to study its relationship. In its computation, self-attention has three vectors, which are "key" which represents information from the patch, "query" which determines how much attention needs to be given to other patches, and "value" which represents the features of each patch. All these vectors are obtained from the previous embedding patch. These three vectors are then used to calculate the attention score of each patch. The attention score gives information about how important the relationship between each other patches is. Then the result from that process is going through the fully connected feed-forward network. This network works separately for each patch without considering its relationship with other patches. This is because each patch is considered an individual element that is marked with its position encoding [25].

### 2.3. CoAtNet-0

CoAtNet is an architecture that was created to combine the advantage of transformers which have a larger model capacity and the advantage of convolutional networks which have better generalization. CoAtNet merges both the convolutional network and the transformer by stacking them vertically. CoAtNet-0 architecture is started with the stem section. This stem section has two  $3 \times 3$  convolutional layers. After the stem, there is a residual connection that is used to reduce the risk of degradation of the neural network. With the residual connection, the model can eliminate the vanishing gradient problem, perform better optimization, and provide alternative paths that make it easier for the model to learn when the network gets deeper. The residual connection is also used for every transition from one section to another section [26].

The next section is the mobile inverted residual bottleneck convolution (MBConv) section which uses an inverted residual connection containing the bottleneck layer and the depth-wise separable convolution. The inverted residual connection is a residual connection with an inverted structure. This structure begins to work by increasing the number of channels first using the convolution layer. Then it will perform deep separable convolutions for efficiency. Afterward, it ended with a bottleneck layer to restore the original channel size. The bottleneck layer is a layer consisting of  $1 \times 1$  pointwise convolution which aims to reduce the number of channels so that the complexity can also be reduced a lot without the need to reduce the model capacity. Depthwise separable convolution is a layer consisting of depthwise convolution which uses one filter for each input channel and pointwise convolution which uses  $1 \times 1$  convolution to combine outputs. Depthwise separable convolution aims to reduce processing costs [26].

After the MBConv section, the next part of the CoAtNet is the transformer section. For this section, the main learning process is carried out by relative attention, which is one of the attention extension mechanisms used to improve the model's ability to capture information on relations between tokens in sequences. Relative attention works in a very similar way to self-attention. However, in calculating attention scores, relative attention uses two additional matrices which are called relative positional encoding matrices. These matrices are used to modify the dot multiplication results used in calculating the attention score. With this matrix, the model can capture relative distance or movement information between tokens and provide positional information that can be used to assist the process of calculating attention scores [26].

CoAtNet has several versions that are different in terms of the layer number and channel size. CoAtNet-0 is one of the CoAtNet versions that have the smallest layer number and channel size of all the other versions [26]. The details of CoAtNet-0 are shown in Table 2 where mobile inverted residual bottleneck convolution is abbreviated as MBConv and transformer relative attention will be abbreviated as TFM Rel.

### 2.4. ConvNeXt-Tiny

ConvNeXt is an architecture that is constructed based on the standard ResNet architecture and made to resemble transformers. This architecture is divided into a stem and several multi-layer perceptron (MLP) sections. For the stem, this architecture uses a patchify strategy from swin transformer which has a convolution layer  $4 \times 4$  and stride 4. For each MLP section, it is divided again into several blocks. These blocks use an inverted bottleneck design like a transformer where each hidden layer has dimensions four times larger than the input and output layers. Each block of this architecture always begins with a depthwise convolution layer  $7 \times 7$ , followed by two  $1 \times 1$  convolution layers. In this architecture, there is also a residual connection that is applied in each block and a convolution layer  $2 \times 2$  that acts as a down sampling layer at each transition of the MLP section. In its original research, ConvNeXt has several versions, one of them is

ConvNeXt-Tiny which is the smallest architecture of all ConvNeXt Version [27]. The details of ConvNeXt-Tiny are shown in Table 3 where mobile inverted residual bottleneck convolution is abbreviated as MBCConv and transformer relative attention will be abbreviated as TFM Rel.

Table 2. CoAtNet-0 architecture details [26]

Section name	Total block in section	Target channel size
Conv (Stem)	2	64
MBCConv 1	2	96
MBCConv 2	3	192
TFM Rel 1	5	384
TFM Rel 2	2	768

Table 3. ConvNeXt-Tiny architecture details [27]

Section name	Total block in section	Target channel size
Convolution (Stem)	1	96
MbConv 1	3	96
MbConv 2	3	192
TFM Rel 1	9	384
TFM Rel 2	3	768

### 3. METHOD

#### 3.1. Dataset and preprocessing

There are two datasets used for the experiment. The first dataset was taken from Attwenger's research which was divided into 90 classes (a combination of 5 duration classes with 17 musical note classes and a rest note class). This dataset has a total of 3824 images with a size of 30×50 pixels [29]. However, this dataset has a few things to note. The first thing to note is that some data do not have complete labels. To handle this problem, these data are not included in the learning process. The second thing to note is the amount of data that is not balanced for each class. To solve the second problem, the average weighted F1-Score is used as one of the evaluation metrics. The second dataset is taken from research conducted by Andrea *et al.* [22] which was divided into 65 classes (a combination of 5 duration classes and 13 musical note classes). This dataset has similar data as Attwenger's dataset, although several classes are not included and have different names for the labels. The total data provided in this dataset is 1,625 images and each image has a different size.

Several preprocessing processes were also carried out. The first preprocessing is resizing each image to the same size for all datasets. Each image in the dataset is resized to 65×65 pixels for ViT, CoAtNet-0, and ConvNeXt-Tiny model input and 75×75 pixels for Inception-V3 model input. Image sizes are made different for the Inception-V3 model due to Inception-V3 limitation which only accepts input with a minimum size of 75×75. After each image has the same size, another preprocessing that is carried out is normalization. Normalization is carried out by changing the value of each pixel in the image into a value between 0–1.

#### 3.2. Data augmentation

Data augmentation was also carried out to create a more diverse dataset and create a model that can provide better performance. Data augmentation is used to give slight changes to the training data using several image augmentation methods. Some of the data augmentation methods that are used are shown in Table 4.

Table 4. Data augmentation method list

Data augmentation	Description
Translation	Carried out by shifting the image to the left, right, up, or down
Brightness change	Carried out by changing the image brightness level to be lighter or darker
Scale	Carried out by changing the image size to be bigger or smaller
Horizontal flip	Carried out by flipping the image horizontally
Rotation	Carried out by rotating the image a few degrees from the original

#### 3.3. Model development

The experiment was done by using three pre-trained models that have proven to be good at classifying ImageNet datasets. These models are ViT which uses transformer architecture to classify images, CoAtNet-0 which is a combination of CNN and transformer architectures, and ConvNeXt-Tiny which is a CNN architecture that has transformer-like architecture. There are also some adjustments given to these models, especially for the input and output layers. The input layer was adjusted to the size of the input data, which is 65×65 pixels. As for the output layer, a fully connected layer was added which has 90 units of neurons for the Attwenger dataset and 65 units of neurons for the Andrea dataset.

### 3.4. Experimental design

For each dataset, 60% of the data are used for training, 20% of the data are used for validation, and 20% of the data are used for testing. The training was conducted for 200 epochs. However, to reduce the risk of overfitting, early stopping was also used with 20 epochs of patience, and validation accuracy was used as the monitored value for the early stopping. This means that when the validation accuracy does not increase in 20 epochs of training, the training process is stopped even though it has not reached 200 epochs.

Several other hyperparameter tunings are used in this training process. For the optimizer, Stochastic gradient descent (SGD) and adaptive moment estimation (ADAM) optimizers are used to be tested on each architecture for this research. In addition, this research also used two learning rate values which are 0.01 and 0.001 to be tested on each architecture. Meanwhile, some of the fixed parameters used in this research are 0.9 as the momentum value, 0.01 as the weight decay value, categorical cross entropy as the loss function used, and 32 as the batch size value.

### 3.5. Performance metric

To evaluate each model's performance, testing was carried out using testing data that has already been divided before running the training process. This testing data's prediction result was displayed using a confusion matrix of 90×90 for the Attwenger dataset and a confusion matrix of 65×65 for the Andrea dataset. By using true positive (TP), true negative (TN), false positive (FP), and false negative (FN) information from this confusion matrix, overall accuracy can be calculated.

$$\text{Overall Accuracy} = \frac{1}{N} \sum_{i=1}^N \left( \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \right) \quad (2)$$

Because of the imbalanced dataset condition of the Attwenger dataset, another performance metric was also used which is the average of the weighted F1-Score. F1-Score is calculated by using the weighted average between precision and recall. Precision is the value of the model reliability level when it gives a positive label prediction. While recall is the value of the model's accuracy in correctly predicting positive data. Then weighted F1-Score can be calculated by using the average of the F1-Score from each class that is already multiplied by the support proportion (SP). SP is the actual proportion of the class in the dataset.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (3)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (4)$$

$$\text{Average Weighted F1 Score} = \frac{1}{N} \sum_{i=1}^N \left( 2 \times \left( \frac{\text{Precision}_i \times \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} \right) \times SP_i \right) \quad (5)$$

## 4. RESULTS AND DISCUSSION

### 4.1. Training and validation results on the Andrea dataset

Figure 1 shows the plot of training and validation accuracy from the ViT model when using the unaugmented Andrea dataset in Figure 1(a) and when using the augmented Andrea dataset in Figure 1(b). It shows that when using the unaugmented Andrea dataset, the ViT model becomes very overfitting. After using the augmented Andrea dataset, ViT model gives a lot less overfitting result.

Figure 2 shows the plot of training and validation accuracy from the CoAtNet-0 model when using the unaugmented Andrea dataset in Figure 2(a) and when using the augmented Andrea dataset in Figure 2(b). It shows that when using the unaugmented Andrea dataset, the model already gives a less overfitting result which is better than ViT. Then it becomes not overfitting when using the augmented Andrea dataset.

Figure 3 shows the plot of training and validation accuracy from the ConvNeXt-Tiny model when using the unaugmented Andrea dataset in Figure 3(a) and when using the augmented Andrea dataset in Figure 3(b). It shows that when using the unaugmented Andrea dataset, the ConvNeXt-tiny model almost does not give an overfitting result. The ConvNeXt-Tiny model is not overfitting after using the augmented Andrea dataset.

Figure 4 shows the plot of training and validation accuracy from the Inception-V3 model when using the unaugmented Andrea dataset in Figure 4(a) and when using the augmented Andrea dataset in Figure 4(b). It shows that when using the unaugmented Andrea dataset, the Inception-V3 model gives a quite overfitting result but is still better than the ViT model. It also becomes better after using the augmented Andrea dataset even though still gives a little overfitting result and is not stable enough.

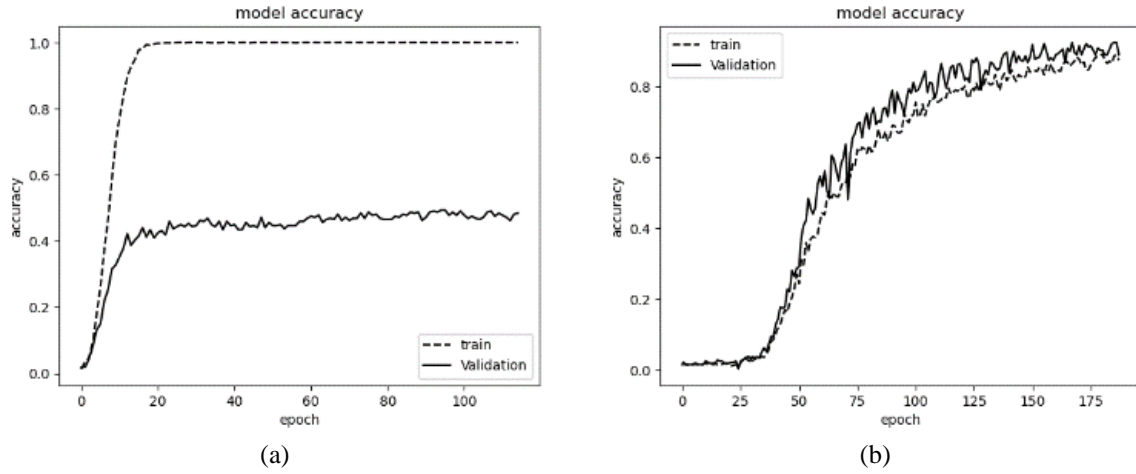


Figure 1. Comparing the ViT model accuracy plot for the Andrea dataset (a) without augmentation and (b) with augmentation

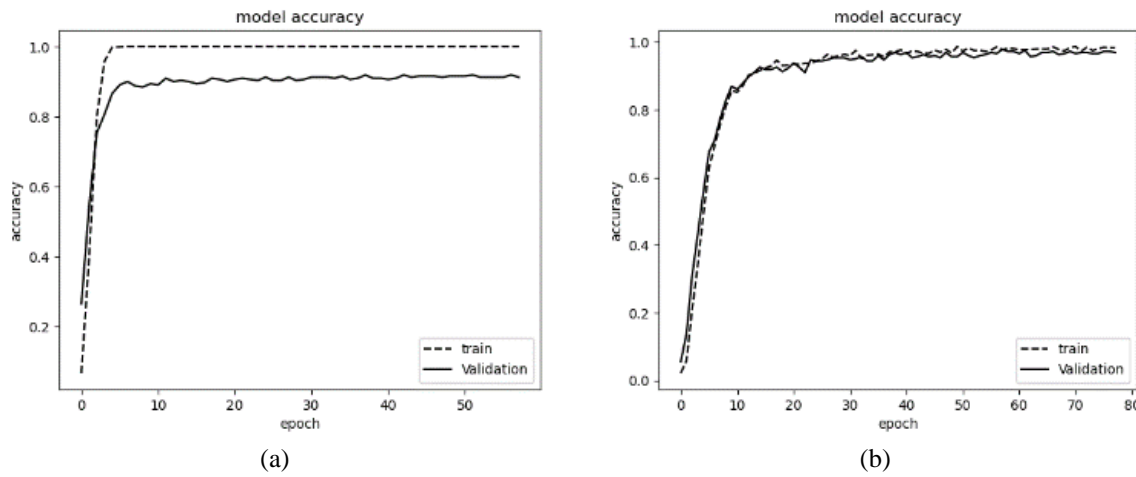


Figure 2. Comparing the CoAtNet-0 model accuracy plot for the Andrea dataset (a) without augmentation and (b) with augmentation

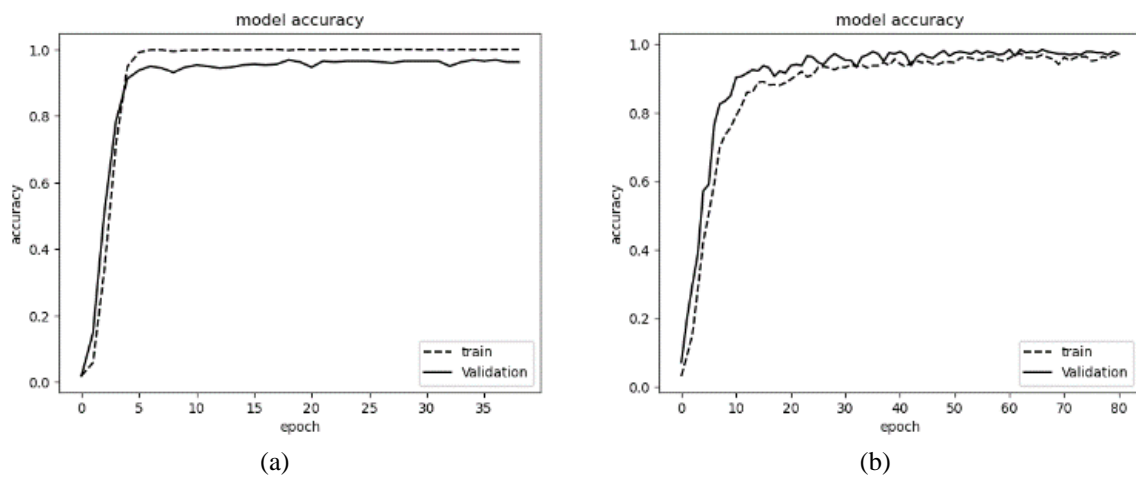


Figure 3. Comparing the ConvNeXt-Tiny model accuracy plot for the Andrea dataset (a) without augmentation and (b) with augmentation

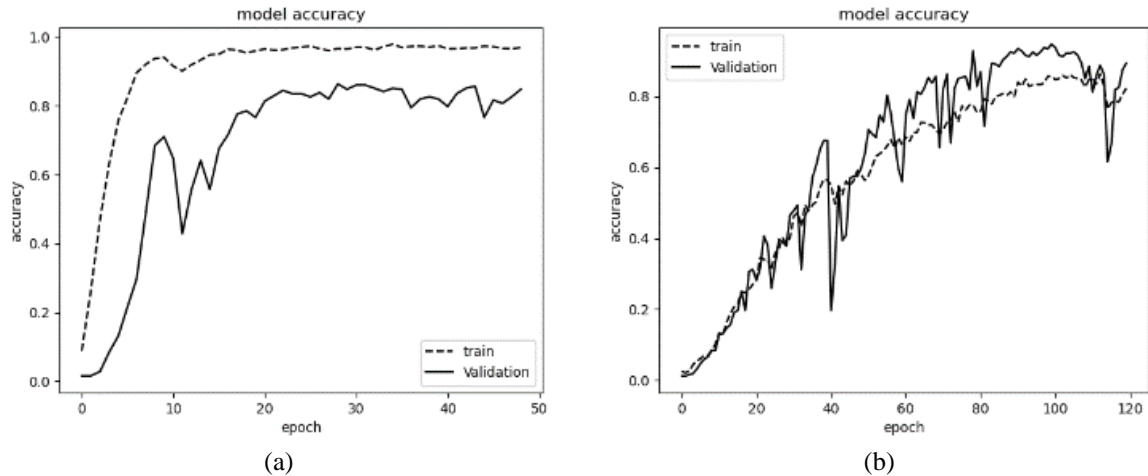


Figure 4. Comparing the Inception-V3 model accuracy plot for the Andrea dataset (a) without augmentation and (b) with augmentation

Based on the training and validation accuracy plots of each model for the Andrea dataset, the use of unaugmented datasets always gives overfitting results. For this reason, it can be concluded that data augmentation is necessary to prevent overfitting. When the models are compared, ConvNeXt-Tiny is the model that gives the least overfitting results, followed by the model from CoAtNet-0. The model from ViT can also provide results that are less overfitting when the dataset has been augmented. However, when the unaugmented dataset is used as the training input, the ViT model can become very overfitting. Meanwhile, the Inception-V3 model, which is the benchmark of this research, still loses in terms of overfitting. This is because the Inception-V3 model still gives quite overfitting results for both augmented and unaugmented datasets. However, Inception-V3 is still better to use than the ViT model when the given dataset is not augmented. From the epoch, it also can be concluded that ConvNeXt-Tiny is the best model for Andrea dataset. It is because ConvNeXt-Tiny needs the least epoch to get its best accuracy for both when using augmentation and not using augmentation. Then it is followed by CoAtNet-0 in second place, Inception-V3 in third place, and ViT in last place for models that need the least epoch to get their best accuracy.

Overall, most of the best models for the Andrea dataset were obtained from hyperparameter tuning using the ADAM optimizer with a learning rate of 0.001. Then it is followed by SGD with a learning rate of 0.01 and 0.001. There is no model that gets its best result by using the ADAM optimizer with a learning rate of 0.01. The best training results for each model with its hyperparameter are shown in Table 5.

Table 5. Best Andrea training dataset result for each model

Model	Augmentation	Training accuracy	Validation accuracy	Optimizer	Learning rate
ViT	Yes	86.53%	92.50%	SGD	0.01
ViT	No	100%	49.38%	SGD	0.01
CoAtNet-0	Yes	97.67%	97.81%	ADAM	0.001
CoAtNet-0	No	100%	91.87%	ADAM	0.001
ConvNeXt-Tiny	Yes	95.97%	98.44%	SGD	0.01
ConvNeXt-Tiny	No	99.79%	96.88%	ADAM	0.001
Inception-V3	Yes	85.68%	94.69%	SGD	0.001
Inception-V3	No	96.50%	86.25%	ADAM	0.001

#### 4.2. Training and validation results on the Attwenger dataset

Figure 5 shows the plot of training and validation accuracy from the ViT model when using the unaugmented Attwenger dataset in Figure 5(a) and when using the augmented Attwenger dataset in Figure 5(b). It shows that when using the unaugmented Attwenger dataset, the ViT model gives quite an overfitting result. On the contrary, when using the augmented Andrea dataset, it gives a lot less overfitting result even to the point it is not overfitting anymore in the end.

Figure 6 shows the plot of training and validation accuracy from the CoAtNet-0 model when using the unaugmented Attwenger dataset in Figure 6(a) and when using the augmented Attwenger dataset in Figure 6(b). It shows that when using the unaugmented Attwenger dataset, the CoAtNet-0 model gives a



slight overfitting result. The CoAtNet-0 model does not overfit anymore when it uses the augmented Attwenger dataset.

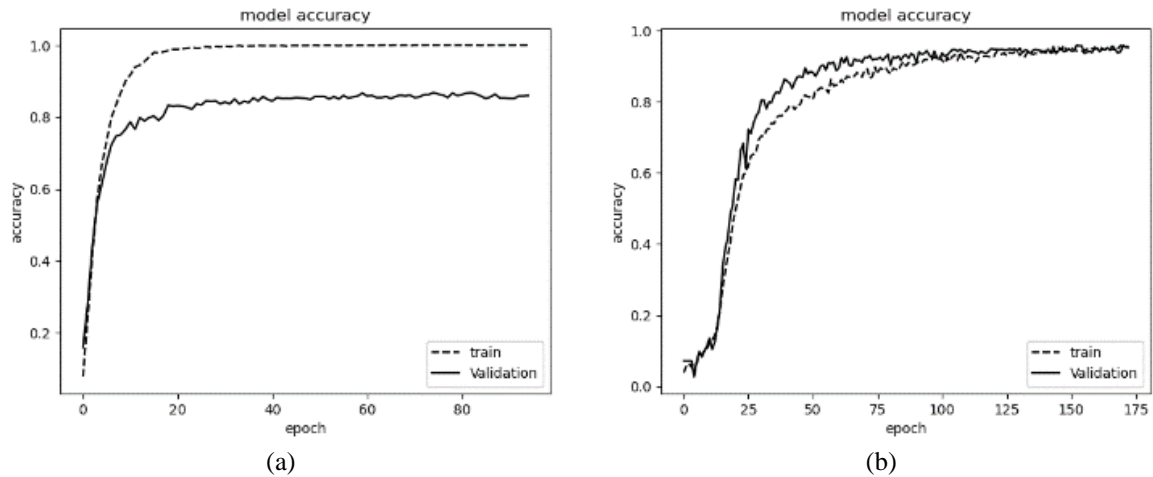


Figure 5. Comparing the ViT model accuracy plot for the Attwenger dataset (a) without augmentation and (b) with augmentation

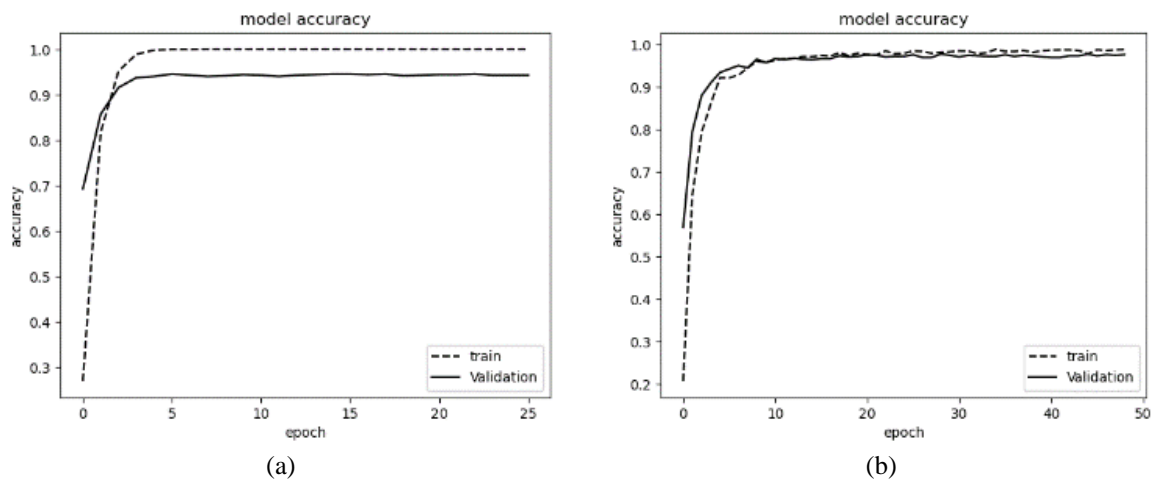


Figure 6. Comparing the CoAtNet-0 model accuracy plot for the Attwenger dataset (a) without augmentation and (b) with augmentation

Figure 7 shows the plot of training and validation accuracy from the ConvNeXt-Tiny model when using the unaugmented Attwenger dataset in Figure 7(a) and when using the augmented Attwenger dataset in Figure 7(b). It shows that when using the unaugmented Attwenger dataset, the ConvNeXt-Tiny model almost does not give an overfitting result. Then it becomes not overfitting after using the augmented Attwenger dataset like the case in the Andrea dataset.

Figure 8 shows the plot of training and validation accuracy from the Inception-V3 model when using the unaugmented Attwenger dataset in Figure 8(a) and when using the augmented Attwenger dataset in Figure 8(b). It shows that when using the unaugmented Attwenger dataset, the Inception-V3 model gives a slight overfitting result. It becomes better after using the augmented Attwenger dataset where it gives less overfitting result and not overfitting in the end.

Based on the training and validation accuracy plot of each model for the Attwenger dataset, the unaugmented Attwenger dataset also always gives overfitting results like the Andrea dataset. This gives more proof that augmentation is very important for preventing overfitting. For the model comparison, the Attwenger dataset also gives very similar results to the Andrea dataset. ConvNeXt-Tiny is the model that gives the least overfitting results for the Attwenger dataset. Then CoAtNet-0 comes second in terms of least

overfitting. The ViT model also has the same conditions which are good enough to be used when the dataset is augmented but becomes quite overfitting when used on unaugmented datasets. Therefore, the Inception-V3 model which has slight overfitting, is still better for the unaugmented datasets than the ViT model. Meanwhile, for the epoch, the order for the model that needs the least epoch is changing. CoAtNet-0 takes the first place for being the model that needs the least epoch then it is followed by ConvNeXt-Tiny. ViT model needs fewer epochs than Inception-V3 when training the unaugmented data and need more epoch than Inception-V3 when training the augmented data. If the number of epochs needed is included in the discussion, ConvNeXt-Tiny and CoAtNet-0 might still need to be debated to choose which is the best model for classifying the Attwenger dataset. But Inception-V3 still can be considered better than ViT when using unaugmented data since ViT did not even achieve validation accuracy above 90% for its best model weight.

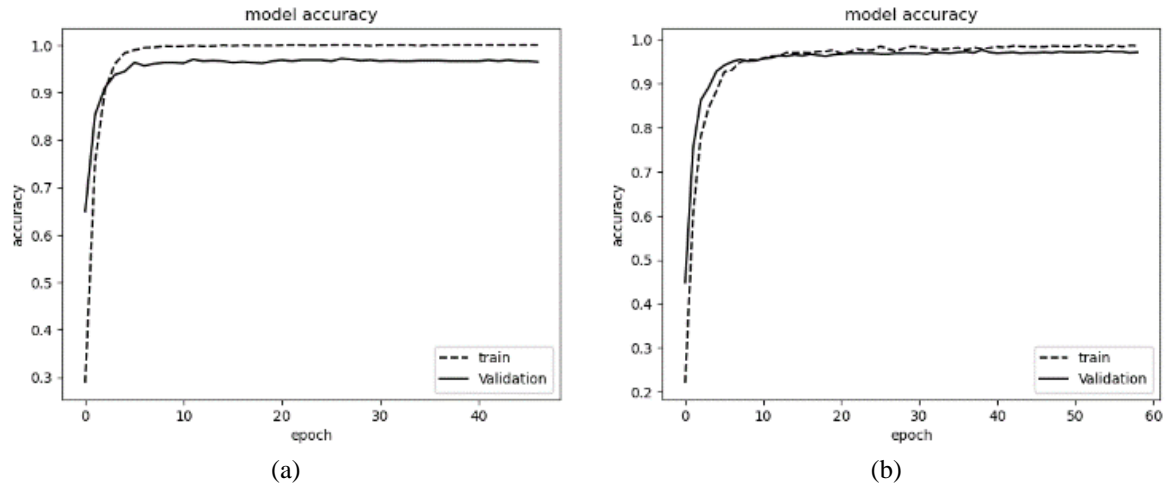


Figure 7. Comparing the ConvNeXt-Tiny model accuracy plot for the Attwenger dataset (a) without augmentation and (b) with augmentation

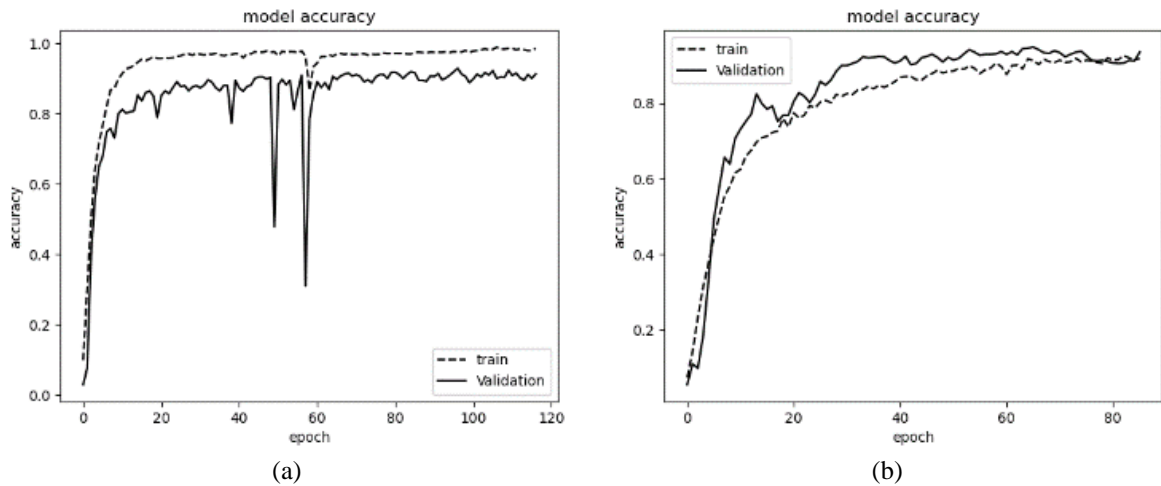


Figure 8. Comparing the Inception-V3 model accuracy plot for the Attwenger dataset (a) without augmentation and (b) with augmentation

For hyperparameter tuning, the Attwenger dataset is also very similar to the Andrea dataset where most of the best models are obtained by hyperparameter tuning using the ADAM optimizer with a learning rate of 0.001. Then it is followed by hyperparameter tuning using the SGD optimizer with a learning rate of 0.01 and 0.001. The hyperparameter tunings that give the best results for each of these models are also shown in Table 6.

Table 6. Best Attwenger dataset training result for each model

Model	Augmentation	Training accuracy	Validation accuracy	Optimizer	Learning rate
ViT	Yes	94.27%	95.79%	SGD	0.01
ViT	No	100%	86.82%	SGD	0.01
CoAtNet-0	Yes	98.10%	97.69%	ADAM	0.001
CoAtNet-0	No	99.96%	94.57%	ADAM	0.001
ConvNeXt-Tiny	Yes	97.93%	97.69%	ADAM	0.001
ConvNeXt-Tiny	No	100%	97.15%	ADAM	0.001
Inception-V3	Yes	91.22%	94.84%	ADAM	0.001
Inception-V3	No	97.40%	92.80%	SGD	0.001

#### 4.3. Testing results on the Andrea dataset

For the Andrea dataset, the augmented Andrea dataset always has better performance than the unaugmented Andrea dataset. Meanwhile, for the model itself, ConvNeXt-Tiny is the model that has the best performance for both augmented and unaugmented data. Although not as good as ConvNeXt-Tiny, CoAtNet-0 also has good performance where CoAtNet-0's performance in classifying is only slightly lower than ConvNeXt-Tiny. Unfortunately, the performance of the ViT model is still lower than Inception-V3, especially for the unaugmented Andrea dataset. The results of each model testing for the Andrea dataset are shown in Table 7.

Table 7. Andrea dataset testing result for each model

Model	Augmentation	Average weighted F1-Score	Overall accuracy
ViT	Yes	90%	90.46%
ViT	No	50%	51.69%
CoAtNet-0	Yes	97%	96.92%
CoAtNet-0	No	91%	91.69%
ConvNeXt-Tiny	Yes	98%	98.15%
ConvNeXt-Tiny	No	95%	95.07%
Inception-V3	Yes	92%	92.61%
Inception-V3	No	85%	85.23%

#### 4.4. Testing results on the Attwenger dataset

For the Attwenger dataset, most of the test results on the augmented Attwenger dataset are also better than the unaugmented Attwenger dataset. But there is one exception where the Inception-V3 model has better performance for the unaugmented Attwenger dataset instead of for the augmented Attwenger dataset. The model comparison results are also almost the same as the Andrea dataset. The best model for the Attwenger dataset is ConvNeXt-Tiny followed by CoAtNet-0. But there is also a slight difference for the Inception-V3 model where this model is better than ViT when the Attwenger dataset is not augmented but worse when the Attwenger dataset is augmented. The results of each model testing for the Attwenger dataset are shown in Table 8.

Table 8. Best Attwenger dataset testing result for each model

Model	Augmentation	Average weighted F1-score	Overall accuracy
ViT	Yes	96%	95.95%
ViT	No	83%	83.70%
CoAtNet-0	Yes	97%	97.26%
CoAtNet-0	No	94%	94.52%
ConvNeXt-Tiny	Yes	98%	98.43%
ConvNeXt-Tiny	No	97%	97.52%
Inception-V3	Yes	93%	93.22%
Inception-V3	No	93%	93.61%

## 5. CONCLUSION AND FUTURE WORK

The use of augmentation has proven its excellent quality for classifying musical notes. This experiment shows that augmentation can help all models reduce the risk of overfitting and improve their performance in classifying. This experiment also shows that ConvNeXt-Tiny is the best model for classifying musical notes. CoAtNet-0 is the second-best model for classifying musical notes with only a slight difference from ConvNeXt-Tiny. Both models have managed to beat the performance of the Inception-V3 which became the benchmark in this research. Both models also successfully surpass the accuracy of several previous research that uses the same dataset which are the Andrea dataset and the Attwenger dataset.

Unfortunately, the ViT model is still unable to beat the Inception-V3 model, especially in cases where the dataset is not augmented. However, with some modification, the ViT model still can beat Inception-V3 when the dataset is augmented because they have only a slight difference in their performance. This modification can be done from two sides. The first side is from hyperparameter tuning that can be experimented with, aside from optimizer and learning rate. The other side is from ViT architecture itself where the future work can add some layers such as an extra fully connected layer, convolution layer, dropout layer, or other potential layer. There are also already many modification results of ViT architecture that can be used and proven to be good for classifying image datasets. Some of those ViT architecture are dual attention vision transformer (DaViT), multi-axis vision transformer (MaxViT), and multiscale vision transformer (MViT).

## ACKNOWLEDGEMENTS

We would like to express our sincere appreciation to Attwenger and Andrea *et al.* for generously providing the dataset used in this research. The availability of their dataset has been instrumental in the successful completion of this research.




## REFERENCES

- [1] B.-L. Bartleet and L. Higgins, *Introduction: An overview of community music in the twenty-first century*, vol. 1. Oxford University Press, 2018.
- [2] N. K. Hasanova, "Possibilities of music education and upbringing in the formation of personal maturity," *Theoretical & Applied Science*, vol. 100, no. 08, pp. 420–422, Aug. 2021, doi: 10.15863/TAS.2021.08.100.79.
- [3] Y. K. Wong, K. F. H. Lui, and A. C.-N. Wong, "A reliable and valid tool for measuring visual recognition ability with musical notation," *Behavior Research Methods*, vol. 53, no. 2, pp. 836–845, Apr. 2021, doi: 10.3758/s13428-020-01461-w.
- [4] J. Julia, I. Isrokatun, H. Pramajati, N. S. Sukaesih, and I. Aisyah, "Improving the song notation reading comprehension and skill of prospective elementary school teachers: An action research study in Indonesia," *Universal Journal of Educational Research*, vol. 7, no. 10, pp. 2057–2067, Oct. 2019, doi: 10.13189/ujer.2019.071003.
- [5] I. Velescu, "The musical notation-a path of interdisciplinary knowledge and development," *Învățământ, Cercetare, Creație*, vol. 6, no. 1, pp. 397–402, 2020.
- [6] A. Pacha, J. Calvo-Zaragoza, and J. Hajič, "Learning notation graph construction for full-pipeline optical music recognition," *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR 2019)*, 2019.
- [7] P. Patel and A. Thakkar, "The upsurge of deep learning for computer vision applications," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 1, pp. 538–548, Feb. 2020, doi: 10.11591/ijece.v10i1.pp538-548.
- [8] A. Rios-Vila, J. Calvo-Zaragoza, and J. M. Inesta, "Exploring the two-dimensional nature of music notation for score recognition with end-to-end approaches," in *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Sep. 2020, pp. 193–198, doi: 10.1109/ICFHR2020.2020.00044.
- [9] S. Ahlawat, A. Choudhary, A. Nayyar, S. Singh, and B. Yoon, "Improved handwritten digit recognition using convolutional neural networks (CNN)," *Sensors*, vol. 20, no. 12, Jun. 2020, doi: 10.3390/s20123344.
- [10] S. Lee, S. J. Son, J. Oh, and N. Kwak, "Handwritten music symbol classification using deep convolutional neural networks," in *2016 International Conference on Information Science and Security (ICISS)*, Dec. 2016, pp. 1–5, doi: 10.1109/ICISSEC.2016.7885856.
- [11] R. M. Pinheiro Pereira, C. E. F. Matos, G. Braz Junior, J. D. S. de Almeida, and A. C. de Paiva, "A deep approach for handwritten musical symbols recognition," in *Proceedings of the 22nd Brazilian Symposium on Multimedia and the Web*, Nov. 2016, pp. 191–194, doi: 10.1145/2976796.2988171.
- [12] J. Calvo-Zaragoza, A.-J. Gallego, and A. Pertusa, "Recognition of handwritten music symbols with convolutional neural codes," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Nov. 2017, pp. 691–696, doi: 10.1109/ICDAR.2017.118.
- [13] J. Sober-Mira, J. Calvo-Zaragoza, D. Rizo, and J. M. Inesta, "Pen-based music document transcription," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Nov. 2017, pp. 21–22, doi: 10.1109/ICDAR.2017.258.
- [14] J. Greet, *Improved optical music recognition (OMR)*. Stanford University, 2017.
- [15] J. Calvo-Zaragoza, A. Pertusa, and J. Oncina, "Staff-line detection and removal using a convolutional neural network," *Machine Vision and Applications*, vol. 28, no. 5–6, pp. 665–674, Aug. 2017, doi: 10.1007/s00138-017-0844-4.
- [16] A.-J. Gallego, A. Pertusa, and J. Calvo-Zaragoza, "Improving convolutional neural networks' accuracy in noisy environments using k-nearest neighbors," *Applied Sciences*, vol. 8, no. 11, Oct. 2018, doi: 10.3390/app8112086.
- [17] A. Pacha and J. Calvo-Zaragoza, "Optical music recognition in mensural notation with region-based convolutional neural networks," *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR 2018)*, 2018.
- [18] F. C. A. Ferano, A. Zahra, and G. P. Kusuma, "Stacking ensemble learning for optical music recognition," *Bulletin of Electrical Engineering and Informatics (BEEI)*, vol. 12, no. 5, pp. 3095–3104, Oct. 2023, doi: 10.11591/eei.v12i5.5129.
- [19] A. Nuñez-Alcover, P. J. P. de León, and J. Calvo-Zaragoza, "Glyph and position classification of music symbols in early music manuscripts," 2019, pp. 159–168.
- [20] A. Othman and C. Direkçöglü, "Recognizing musical notation using convolutional neural networks," (In Turkey), *European Journal of Science and Technology*, Nov. 2020, doi: 10.31590/ejosat.823266.
- [21] J. Calvo-Zaragoza, J. R. Rico-Juan, and A.-J. Gallego, "Ensemble classification from deep predictions with test data augmentation," *Soft Computing*, vol. 24, no. 2, pp. 1423–1433, Jan. 2020, doi: 10.1007/s00500-019-03976-7.
- [22] N. A. Andrea, N. A. Paoline, and A. Zahra, "Music note position recognition in optical music recognition using convolutional neural network," *International Journal of Arts and Technology*, vol. 13, no. 1, 2021, doi: 10.1504/IJART.2021.115764.




- [23] J. A. Barbosa and E. B. dos Santos, "CREATES - convolutional neural network applied to identification of musical elements in OMR," in *Anais do XVIII Simpósio Brasileiro de Computação Musical (SBCM 2021)*, Oct. 2021, pp. 221–224, doi: 10.5753/sbcm.2021.19452.
- [24] M. Xu, S. Yoon, A. Fuentes, and D. S. Park, "A comprehensive survey of image augmentation techniques for deep learning," *Pattern Recognition*, vol. 137, May 2023, doi: 10.1016/j.patcog.2023.109347.
- [25] A. Dosovitskiy *et al.*, "An image is worth 16x16 Words: transformers for image recognition at scale," *arXiv:2010.11929*, Oct. 2020.
- [26] Z. Dai, H. Liu, Q. V Le, and M. Tan, "CoAtNet: Marrying convolution and attention for all data sizes," *Advances in neural information processing systems*, pp. 3965–3977, 2021.
- [27] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A ConvNet for the 2020s," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 11966–11976, doi: 10.1109/CVPR52688.2022.01167.
- [28] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 2818–2826, doi: 10.1109/CVPR.2016.308.
- [29] P. Attwenger, *Recognizing musical notation using artificial neural networks*. University of Vienna, 2015.

## BIOGRAPHIES OF AUTHORS



**Ciara Setyo**    is a student currently pursuing a master's degree in computer science at Bina Nusantara University. She is currently also working as a front-end developer specializing in React.js and Next.js for web development. Her research interests include deep learning and computer vision. She actively seeks opportunities to collaborate on challenging projects and envisions a future career that harmoniously blends her front-end development expertise with cutting-edge research in deep learning. She can be contacted at email: ciara.setyo@binus.ac.id.



**Gede Putra Kusuma**    received PhD degree in electronic engineering from Nanyang Technological University (NTU), Singapore, in 2013. He is currently working as a lecturer and head of Department of Master of Computer Science, BINUS Graduate Program, Bina Nusantara University, Indonesia. Before joining Bina Nusantara University, he was working as a Research Scientist in I2R-A\*STAR, Singapore. His research interests include computer vision, pattern recognition, deep learning, face recognition, appearance-based object recognition, and indoor positioning system. He can be contacted at email: inegara@binus.edu.