Deep Learning-Based Optical Music Recognition for Semantic Representation of Non-overlap and Overlap Music Notes

Rana L. Abdulazeez[1] and Fattah Alizadeh[2]

[1]Department of Software and Informatics Engineering, College of Engineering, Salahaddin University-Erbil, Kurdistan Region – F.R. Iraq

[2]Department of Computer Engineering, School of Science and Engineering, University of Kurdistan Hewler, Kurdistan Region – F.R. Iraq

***Abstract*—In the technology era, the process of teaching a computer to interpret musical notation is termed optical music recognition (OMR). It aims to convert musical note sheets presented in an image into a computer-readable format. Recently, the sequence-to-sequence model along with the attention mechanism (which is used in text and handwritten recognition) has been used in music notes recognition. However, due to the gradual disappearance of excessively long sequences of musical sheets, the mentioned OMR models which consist of long short-term memory are facing difficulties in learning the relationships among the musical notations. Consequently, a new framework has been proposed, leveraging the image segmentation technique to break up the procedure into several steps. In addition, an overlap problem in OMR has been addressed in this study. Overlapping can result in misinterpretation of music notations, producing inaccurate findings. Thus, a novel algorithm is being suggested to detect and segment the notations that are extremely close to each other. Our experiments are based on the usage of the Convolutional Neural Network block as a feature extractor from the image of the musical sheet and the sequence-to-sequence model to retrieve the corresponding semantic representation. The proposed approach has been evaluated on The Printed Images of Music Staves dataset. The achieved results confirm that our suggested framework successfully solves the problem of long sequence music sheets, obtaining SER 0% for the non-overlap symbols in the best scenario. Furthermore, our approach has shown promising results in addressing the overlapping problem: 23.12 % SER for overlapping symbols.**

***Index Terms*—Sequence-to-sequence, Long short-term memory network, Convolutional neural network, Segmentation, Semantic representation, Overlapping.**

I. Introduction

Musical notation is a visual representation of a musical sound heard or imagined or a sequence of visual instructions for playing music or performing it later by the musician. As such, it is a necessary approach toward conserving musical compositions and enabling the preservation of the music phenomenon. Despite the fact that musicians can read and interpret fairly sophisticated musical notation, no system can do so as of yet (Calvo-Zaragoza, et al., 2020; Castellanos, et al., 2020).

The digitalization process of musical score libraries is an essential initial step in different data-driven methods of musical analysis or other uses requiring the utilization of digital forms to modernize and enhance many facets of the music industry from performance and research to education and cooperation. Indeed, optical music recognition (OMR) aims to translate musical shapes from musical notation graphics. The outcome will be a new version of the score, which can be read by the computer, such as MusicXML and Musical Instrument Digital Interface. These forms would keep and maintain musical properties and stuff such as pitches, duration, dynamics, and notes (Shatri and Fazekas, 2020). Even though there is a great deal of research conducted in the literature, still some challenges are available that should be tackled to enhance their accuracy and performance. Problems such as segmentation, recognition accuracy, overlapping symbols, low-quality paper sheets, and process speed, are among the ones that attracted the researcher's attention.

In the field of OMR, some researchers in previous works tend to adopt Machine translation techniques such as the sequence-to-sequence (seq2seq) model, which is made up of two recurrent neural networks (RNN): an encoder and a decoder. Seq2seq models have a significant role in recent achievements in natural language processing (NLP) approaches such as machine translation and speech recognition. However, seq2seq models are unable to preserve global implicit information from a long sequence of words (Jang, Seo, and Kang, 2019) causing improper music note recognition and a high error rate in long sequence musical sheets. The OMR systems for converting images of music scores into another music notation format such as semantic representation produce a long sequence of words to tell their musical meaning. Furthermore, in the case of multiple-voice polyphonic staves (overlapping), various symbols

ARO-The Scientific Journal of Koya University
Vol. XII, No. 1 (2024), Article ID: ARO.11402. 9 pages
Doi: 10.14500/aro.11402
Received: 13 September 2023; Accepted: 01 March 2024
Regular research paper: Published: 11 March 2024
Corresponding author's e-mail: rana.abdulazeez@su.edu.krd

may occur at the same time (Shatri and Fazekas, 2020). The segmentation of complex notations is still a challenging problem in OMR given that proper segmentation is necessary to obtain correct symbol recognition.

In this study, a novel segmentation algorithm has been proposed to address the long sequence problem along with an OMR system. The presented OMR architecture includes two common deep learning models, convolutional neural networks (CNNs) and RNNs. The CNN block is used to learn the feature representation of the input image. The target tokens are then produced directly by the seq2seq model from the acquired representation learned by the CNN. The major advantage of this approach is the ability of the seq2seq model to be trained by pairs of inputs: The input images and their corresponding transcription, and it can handle variable lengths and capture context information. In addition, to engage with the complex notations issue, a sophisticated algorithm to detect and solve the overlap situation has been suggested. The following list summarizes the main contributions of this study:

- This study proposes a new segmentation algorithm, which is then applied to the (PrIMuS) dataset to generate a new version of it
- The previous OMR works used the VGG pre-trained model to extract the relevant features, whereas ResNet50, Xception, and VGG16, 19 pre-trained models were used in this work as feature extractors
- A novel method is proposed to detect and segment the overlapping music notes to solve the overlap issue.

The structure of the paper is as follows: Sec. II reviews the state-of-the-art methods relevant to this work. Sec. III detailed the proposed approach starting from the dataset preparation phase and model architecture design to the prediction phase. Sec. IV explains the usage of the image augmentation technique. Sec. V addresses the overlapping symbols. Sec. VI provides the results and discussions. Finally, Sec. VII gives the conclusion and future work.

II. Related Work

This section describes the most common OMR methodologies and the state-of-the-art methods relevant to this work. We first provide an overview of the traditional OMR approaches and then review the most current deep-learning-based available techniques.

Over the past few decades, the OMR problem has been tackled by computer vision and pattern recognition based on traditional techniques. Since the musical score follows a sequence, the Hidden Markov Model has been employed (Pugin, 2006; Pugin, Burgoyne, and Fujinaga, 2007). Although this method has shown promising results, there is room for more experimentation in the OMR field due to traditional approaches are unable to cover all challenges.

In recent years, the performance of OMR systems has been significantly enhanced thanks to deep learning models, and promising results on OMR problems were shown by deep learning algorithms. Calvo-Zaragoza, Valero-Mas and Pertusa (2017) have set the basis for the evolution of models that directly work with a major part of the OMR framework by submitting a system based on an end-to-end approach. The presented model is based on a recurrent CNNs architecture that accepts the image of the monophonic notes as input and the output is a sequence of music descriptions. Then, Calvo-Zaragoza and Rizo (2018) presented their dataset called Printed Images of Music Staves (PrIMuS) to train the suggested OMR deep-learning model based on (RCNN). However, the usage of this architecture might not offer the accuracy at the symbol level needed for musical content transcription. In addition, the current approach could not effectively capture long-range dependencies and global context.

There are attempts to adapt some machine translation techniques in the OMR field, such as the seq-to-seq model. In 2017, a novel convolutional seq2seq architecture model was presented by Van Der Wel and Ullrich (2017) in the direction of a trainable end-to-end OMR pipeline. Two common deep learning algorithms have been employed: First, the CNN block converts the input image window to feature vectors. Second, an encoder RNN encodes the feature vectors to the context vector representation. Followed by a decoder RNN decodes the context vector to the sequence of labels. The algorithm's image input is described as a series of image patches produced by sliding a window across the original input score. Then, an OMR system based on the seq2-seq model with an attention mechanism has been presented by Baró, Badal, and Fornés (2020). They tried to adapt the seq2seq model which is used in handwritten recognition and machine translation problems to recognize old music notations, and their study was inspired by the work submitted by Michael, et al. (2019) for handwritten text recognition. To decrease statistically improbable outcomes of Handwritten Music Recognition, Torras, et al. (2022) presented a study based on making use of a language model with the seq2seq model. The pipeline of the system started with giving the input image to the VGG 19 to extract the high-level features. The encoder was a stacked layer of bi-directional gated recurrent units to create the intermedium context vector. In the decoder phase, at each time step, an attention-weighted summary of the hidden state is computed and given the output token. The current output token and the final state are used as input for the next time step. To improve the OMR system performance, Language Model Integration has been employed with the seq2seq model. Furthermore, a novel sequence (Seq2Seq) framework, based on the transformer with a masked language model (ST-MLM) was submitted by Wen and Zhu (2022). This approach consists of five modules: Pre-processing, encoder, decoder, transformer with masked language model (T-MLM), and output. The encoder layer uses a multi-scale CNN and Bi-LSTM for music symbol information, whereas the T-MLM layer uses masked self-attention. Nevertheless, employing the attention mechanism with seq2seq models has limitations: First, increasing the computational cost and level of complexity of the model through needing to add more parameters. Second, Seq2seq

models might be influenced by noise and redundancy through generating unnecessary or repetitive input or output positions or by producing too uniform or sparse attention weights. Third, different structures lengths, vocabularies, domains, or languages between input and output sequences can cause alignment problems and mismatches in the domains that attention mechanisms process.

III. Proposed Approach

The general flow of the proposed framework includes the following three phases: Dataset preparation phase, model architecture design phase, and prediction phase, as depicted in Fig. 1. In the sequel, a complete explanation of each step will be presented.

A. Dataset Preparation phase

Segmentation of the music sheets is an important step of the proposed work by which each music symbol or several symbols connected by a beam will be extracted from the input music sheet. Since the music symbols are mainly stored in a sheet containing a long sequence of individual symbols, a robust and efficient segmentation algorithm can play a significant role in the recognition output. The PrIMS dataset (Calvo-Zaragoza and Rizo, 2018) which contains images of music scores and their musical meanings (semantic representation as a text file) is utilized as a basic material to generate our dataset. The musical meanings might have a large number of sequence words which makes the recognition process quite challenging by seq2seq models. The seq2seq model is a unique type of RNN architecture that is usually used in complex language problems such as machine translation (Jang, Seo, and Kang, 2019). Consequently, the suggested resolving, splitting the process into smaller steps which means dividing the image (its associated semantic representation) into several smaller pieces of images/ semantic representation. A segmentation algorithm inspired by the vertical projection concept in image processing has been designed. The starting and ending columns of each symbol or several symbols connected by a beam are found. Next, each sub-images between starting and ending pixels are cropped and saved, separately, as shown in Fig. 2. Around 100 images are randomly selected from the (PrIMS) dataset and they have been fed to the segmentation algorithm. The complete algorithm is shown in Algorithm 1. After applying the proposed segmentation algorithm on the selected sheet, a dataset of 800 images and their description file is generated.

Algorithm 1: Segmentation Algorithm.

Input: The music sheet contains several musical notes.
Output: Set of images of individual music symbols or a group of beaming music notes.

1- Read the input image, and convert it to the binary form
2- Apply staff lines removable.

Object detection part

1- Set the following variables with values:
start=[]
end=[]
s_flag=False
e_flag=True
t_flag=False
sum=0

2- Check every pixel, column-wise
For i in range(width)
Begin
For j in range(height)
Begin
If in_image[j][i]== 0 and s_flag==False and e_flag==True
(means: detecting the start/first column of the symbol)

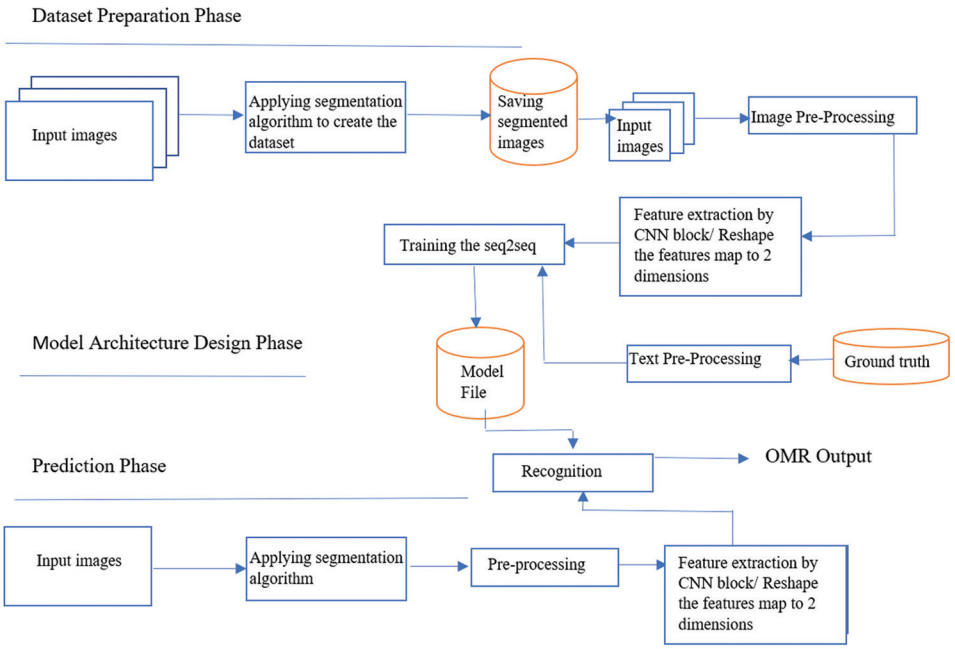

Fig. 1. Proposed approach.

http://dx.doi.org/10.14500/aro.11402

Then
s_flag=True
t-flag=True
e_flag=Fals
if in_image[j][i]==255 and s_flag==True and e_flag==True (seeking for the first white column after the symbol detection to get the end column)
then
sum+=1
end
if temp_f== True then
start.append(column)
if sum== height (means: getting the first white column after symbol detection to obtain the end column)
then
end.append(end)
end

Segmentation part

1. or_img=raw image (with five lines)
2. for I in range (length(start))
   crop the raw image with a fixed height and variable width which is taken from start and end lists.
   Cropped=or_img[0:height,start[i]: end[i]
3. save the cropped image

End of Algorithm

The suggested deep learning model requires two types of input data: Image (musical notes) and text (musical meaning). Thus, the image with its description should be encoded before being fed to the model. From the created corpus of music notes, each image is read from the defined path. The dimensions of every image are altered according to the architecture of the feature extractor that is used (block of CNN). It is extremely common to normalize the input images in computer vision tasks. As mentioned above, the text data are one of the inputs to the OMR system (semantic representation). Hence, a part of this work is related to NLP. The methods that are used in text pre-processing for NLP problems are followed. Because (PrIMuS) dataset is a clean one, the only step is needed to convert the words to numbers in an efficient manner to enable the machine to deal with them.

Every token or string of the semantic representation is read from the ground truth, and then, <start> and <end> tokens are added to the beginning and end of the string. Finally, each string or word is converted to an integer representation to be suitable for the embedding layer.

B. Model Architecture Design

The proposed OMR system pipeline consists of two main parts: A block of a CNN and a seq2seq model (an encoder and a decoder).

The variety of CNN block structures as a feature extractor has proven to be an effective method (Brownlee, 2019) among which two approaches are used in this study: CNN with transfer learning and CNN without transfer learning. Transfer learning is a machine-learning technique in which a pre-trained model is reused as a fundamental point in another task. High-performance models are trained on large datasets, such as ImageNet datasets, to detect common features and extract them. The output will be from the layer before the output layer of the model (Brownlee, 2019). In this study, the feature extractor has been implemented with some pre-trained models such as VGG19, ResNet50, and Xception. Then, the features are reshaped into a two-dimensional features map to be later input into the seq2seq model. Adding to the previous models, another deep learning model is suggested consisting of a block of CNN and the seq2seq model without employing the transfer learning technique to examine the role of transfer learning on the results. The CNN used in this experiment includes four convolutional layers with an increased number of filters starting with 32, 64, 128, and 256 to enhance its ability to learn complex and diverse features from input data, and kernel size (3 × 3), four batchnormalization layers; each batchnormalization layer transforms the inputs to maintain the mean output close to zero and the output standard deviation close to one, (2 × 2) first two max pooling layers and second two max-pooling layers with (2 × 1). The rectified linear unit activations have been used. All images are resized into (224,224,3). The configurations of the proposed CNN are depicted in Table I.

The second part of the proposed OMR system is a seq2seq model (encoder–decoder) which is used in the text recognition method and has been adapted to music scores. The aim rationale behind seq2seq which follows the encoder–decoder structure is to separate the decoding from the feature extraction. Two main components are available in the model: First, the encoder, which converts the entire input sequence to a fixed-size representation (context vector), then the decoder that produces the output sequence one token at each time step (Van Der Wel and Ullrich, 2017; Mondal, et al., 2022; Neubig, 2017; Baró, Badal, and Fornés, 2020). LSTM is simplified and represented in equation (1).

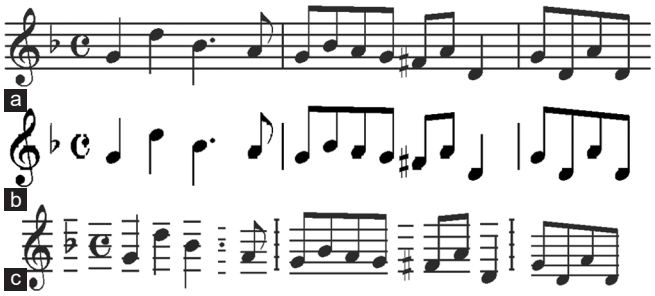

Fig. 2. Segmentation steps results, (a) the input image from (PrIMuS), (b) the image after applying the staff line removable function, and (c) the final segmented sub-images.

TABLE I
CNN Customized Model Structure

Input (224,224,3)
Convolutional block
Conv (32, 3×3, relu), Batch Normalization(), (Max Pooling (2×2))
Conv (64, 3×3, relu), Batch Normalization(), (Max Pooling (2×2))
Conv (128, 3×3, relu), Batch Normalization(), (Max Pooling (2×1))
Conv (256, 3×3, relu), Batch Normalization(), (Max Pooling (2×1))

$$ht,ct = LSTM\ (xt,ht-1,ct-1) \quad (1)$$

At each time step t, the encoder iteratively inputs the sequence $x_1,\ldots,x_T$ of length T and updates the hidden state vector h_t and cell memory state vector c_t.

In the following, the layers of the proposed network along with their functionality will be detailed.

1. Two input layers are available in seq2seq architecture: The input layer is used to feed the encoder with the features map of the input image takes the shape of two dimensions NumPy array and the input layer that used to supply the decoder with the target tokens to fulfill the teacher forcing training concept and takes the shape of the length of the input sequence
2. The encoder is a single LSTM layer, with 256 hidden units and 50% dropout, to avoid the vanishing gradients problem. In the encoder, the final hidden state and cell state are kept for further use as an initial state of the decoder, whereas the outputs of the encoder at each time step are discarded. The encoder produces a final encoded feature map (context vector). Formally, the feature map X of the input image I is generated by the CNN block, this is equivalent to a series of column vectors $X = (X1,\ldots, XM)$, where M is the length of the input sequence. Followed by this sequence's processing by the LSTM layer (encoder) to produce the final feature map $H=(h1,\ldots,hM)$
3. The decoder module is formed by a single LSTM layer with 256 hidden units and 50% dropout to generate the target token of the note that is present in the input image. As mentioned, the initial state of the decoder is set with the final cell and hidden states of the encoder, this helps the decoder to generate the target token. In the decoder, the output from each time step serves as the input for the following time step. The decoder determines the most likely token yt, at each time step t by computing a probability distribution over the vocabularies of potential tokens. dependent on its prior predictions (y1,…,yt-1) along with some context vector ct which involves details from the encoded features H. It determines the probability throughout the series of outputs Y=(y1,…,yT), dividing the combined probability into the ordered conditionals $p(Y)=\Pi^{T}_{t=1}p(yt \mid y1,\ldots,y\text{-}1,ct)$, where the output sequence length is represented by T
4. An embedding layer is a word embedding layer or word representation layer employed in the seq2seq model to represent the target tokens that feed into the decoder (Van Der Wel and Ullrich, 2017; Mondal, et al., 2022; Baró, Badal, and Fornés, 2020; Neubig, 2017). It is a class of approaches used for word representation, it represents each word in a fixed-size vector corresponding to particular words in the vocabulary (Sutskever, Vinyals, and Le, 2014; Brownlee, 2019)
5. Two dense layers exist. First, a dense layer with 256 neurons and a ReLU activation function to improve the performance of RNN (LSTM). ReLU activation function in the Dense layer efficaciously tackles the vanishing gradient problem (Matrenin, et al., 2020). Second, the dense layer of (vocab_size) and Softmax activation function. A Softmax to find a probability distribution to determine the final output (token) (Van Der Wel and Ullrich, 2017; Mondal, et al., 2022; Baró, Badal, and Fornés, 2020; Neubig, 2017). Fig. 3. shows the whole OMR system.

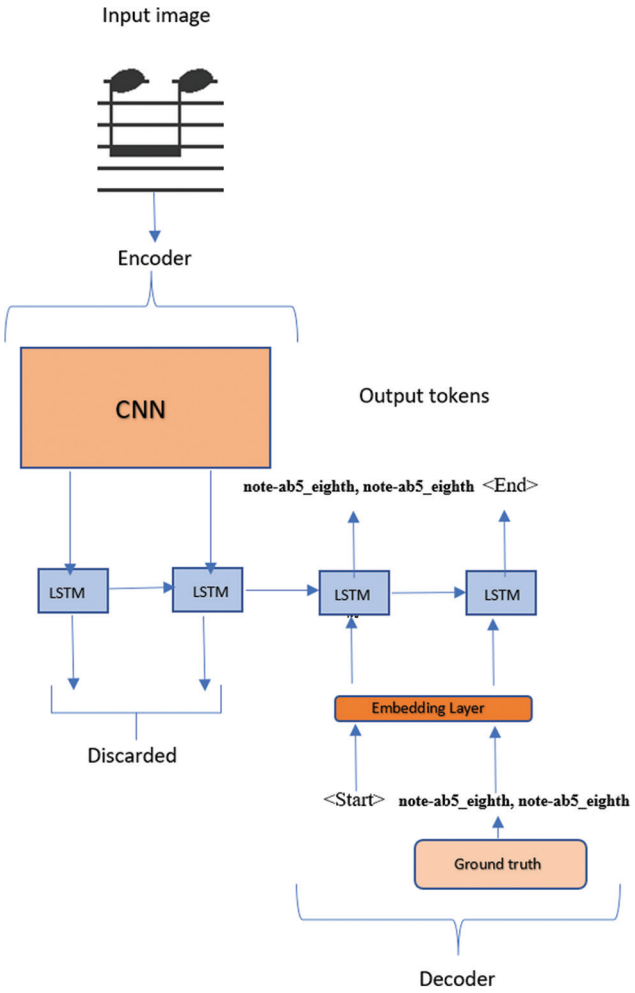

Fig. 3. Optical music recognition system.

The neural seq2seq model is trained to make a prediction of the probability distribution for the next token given the previous context. At each time step, the probability is maximized and assigned to the correct token by making use of the categorical_crossentropy loss function. For a given random variable or series of events, categorical_crossentropy is used for a multi-class classification model and the output will be assigned to an integer value (Brownlee, 2019). Training is done using the Adam optimizer to iteratively adjust network weights using training data. As mentioned before, the target tokens are expanded in pre-processing step adding two artificial tokens <start> and <end>. At training time, making use of the teacher force ensures that the decoder can see the correct tokens of the previous time step for rapidly and efficiently training the RNN model (Sutskever, Vinyals, and Le, 2014; Brownlee, 2019).

C. Prediction Phase

Once the defined seq2seq model is trained and saved, it can be used for the prediction. Nevertheless, the structure

 http://dx.doi.org/10.14500/aro.11402

of the model is not engineered to be used recursively to generate one token at a time because the model is defined for the training phase to learn weights. Therefore, two models are required for the prediction phase. A model for encoding the features map, and a model gets the token generated so far and its encoding as inputs and predicts the next token in the sequence (Brownlee, 2019).

The pipeline of the prediction process in our experiment:

Input: The image of the music sheet (a long sequence of notes from [PrIMS] dataset).

Output: The musical meaning of all notes presented on the input image.

1. Applying a segmentation algorithm to split the input image into individual notes or grouped notes connected by beamline
2. Depending on the number of segmented images, the prediction operation is iterated

IV. Image Augmentation

Computer vision models frequently utilize image augmentation techniques to improve the accuracy of the model by increasing its ability to recognize new variants of the training data. In this work, the impacts of three types of image augmentation techniques have been explored in the results. The blurring technique is implemented with (5,5) kernel size, in the rotation technique the images are rotated at (−15,15) degrees, and the images are resized between 50% and 150% in the scale technique (Fig. 4).

V. Dealing with Overlapping Symbols

Shatri and Fazekas (2020) have addressed some of the major challenges such as overlapping and complexity of common Western music notation. The overlapping situation happens when various notes might be played at the same time. The OMR system should be able to recognize each symbol individually. As a consequence, a novel algorithm is designed to detect the overlap problem and segment the existing symbols in the input image. The presented algorithm was inspired by the set intersection concept in mathematics. The intersection is represented by $A \cap B$, more formally, $x \in A \cap B$ if $x \in A$ and $x \in B$. The intersection of two sets that include the same elements in both sets (Rosen, 2018). In the case of overlapping, after removing the staff lines, each image has two symbols that are overlapping vertically. All columns that the symbol passes through are considered a set. Thus, each of the two symbols is associated with a set of columns. Once overlapping occurs, the input image has two symbols: One above and one below. The two symbols can have some common columns which results in a segmentation problem. Algorithm 2 shows the steps to solve the overlapping case:

Algorithm 2: Overlap Detection Algorithm

1. Check the input image if contains one symbol or more by testing each pixel row by row
2. Find the start column and the end column of the upper symbol, then save them in variables
3. Find the start column and the end column of the lower symbol, then save them in variables
4. Create set 1 which contains numbers between the start and the end columns of the upper symbol:
   For example:
   If the first (upper) symbol starts in column 4 and ends in column 10 the set1 will be:
   Set1= {4,5,6,7,8,9,10}.
5. Create set 2 which contains numbers between the start and the end columns of the lower symbol:
   For example:
   If the second (lower) symbol starts in column 3 and ends in column 6 the set 2 will be:
   Set2= {3,4,5,6}.
   From set 1 and set 2, we can see that both symbols overlapped in columns 4, 5, and 6.

The overlapping can be determined when we apply the intersection concept.

After determining the overlapping input images, the available symbols are segmented into two images, as shown in Fig. 5.

However, after applying the above algorithm, each segmented image loses some staff lines. On the other hand, the deep learning model is trained using samples of notes with the staff lines. As a result, adding the missing lines of every segmented image is a crucial step to enable the model to recognize the music notes properly. To add the missing lines, the thickness of the staff line and the white space between every two lines are found. Then, the number of missing staff lines is calculated for each segmented image. Then, two plain white images are created, and then, a specified number of lines equal to the number of missing

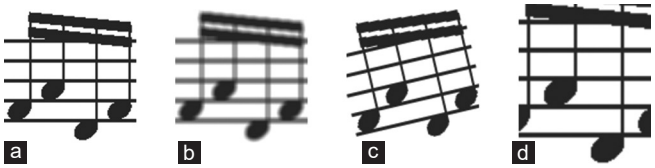

Fig. 4. Original image with its augmented images; (a) original image, (b) blurring image, (c) rotation image, and (d) scaling image.

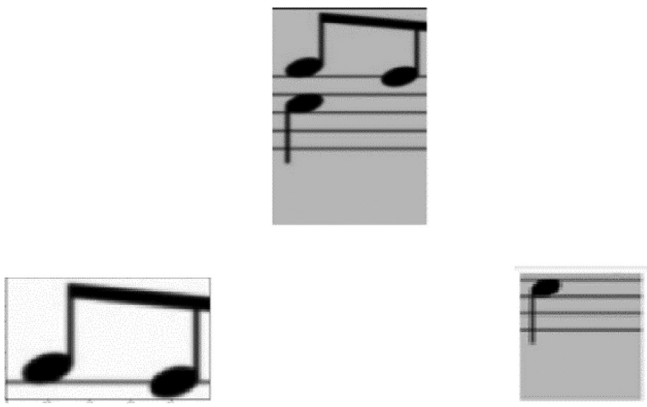

Fig. 5. The input image with its two segmented overlapped notes.

lines will be drawn in each generated image. Finally, each segmented image is concatenated with its corresponding created white image (the white image contains the staff lines) to form an output image containing a music note with five staff lines.

VI. Results and Discussion

A. Evaluation Metric

At present, the commonly used evaluation metric to assess OMR models is Symbol Error Rate (SER) as a substitute for the well-known character/word error rate which is utilized in text recognition tasks, due to the concern in the computational aspect (Baró, Badal and Fornés, 2020). The error rate is used to determine the extent of the difference between the OMR transcribed text and the ground truth text. The SER can be found by (2) equation.

$$SER = (S+D+1)/N \quad (2)$$

where S refers to the substitutions, D the deletions, I the insertions, and N the number of characters in the ground truth.

B. Dataset

To evaluate the proposed methodology, our generated dataset is used which consists of 800 images with its ground_ truth text file. The dataset is randomly divided into two sets: The training set and the unseen set of samples. Three strategies for splitting data are used in this study. Multiple scenarios are trained using data augmentation techniques including blurring, rotation, and scaling.

C. Training seq2srq Models

Diverse models are trained with the original dataset and using data augmentation depending on the learning method used in the feature extractor models: With the transfer learning concept, including VGG pre-trained model, ResNet50 pre-trained model, and Xception pre-trained model, and without the transfer learning concept, using the proposed customized model. The epochs number used in this study is up to 30, and batch sizes are 8,16, and 32.

Table II shows the SER % of a test set using VGG19 as a feature extractor and the best results values are in bold formatting to indicate the minimum SER %. In general, the ability of the model to learn was improved by increasing the batch size whereas applying data augmentation techniques badly affected SER. Likewise, as illustrated in Table III, the achieved SER from training the seq2seq model with Resnet50 has decreased through increasing the batch size, particularly in batch size 32, the best results have been bolded to draw attention to the minimum SER % in this experiment. However, the SER has been increased once data augmentation techniques are applied. Furthermore, the results of using Xception as a feature extractor as depicted in Table IV are sanguinely improved compared with VGG19 and Resnet50 in both batch sizes 16 and 32. Bold formatting has been used to emphasize the experiment's best results values which represent minimum SER %. To test, compare, and evaluate other available pre-trained models with CNN block without transfer learning, we trained the seq2seq model with the proposed CNN model as a feature extractor. The practical results are presented in Table V.

From the presented results, there is a considerable difference between the performance of the feature extractor pre-trained models and the proposed CNN block for feature extraction. The architecture of the feature extractor model plays a vital role in enhancing the performance of the seq2seq model. Based on the achieved SER, the usage of CNN block with very deep layers including VGG19, Resnet50, and Xception has decreased SER. It is also worth mentioning that the utilization of the Xception model to extract the relevant features reduced SER compared with VGG19 and ResNet50. The Xception model relies on both depthwise and pointwise convolution concepts, with its effective architecture, it can accurately and efficiently capture small details in the images.

Moreover, employing a pre-trained learning model is useful to speed up the training process (Fig. 6). Noticeably, Xception requires less training time compared with other pre-trained CNN models (VGG16,19 and ResNet50), that is because of the depthwise convolution and the pointwise convolutions concepts used in Xception. Depthwise convolution helps reduce the computational complexity of the convolutional layer by reducing the number of parameters and computations compared to traditional convolution. Pointwise convolution helps adjust the number of channels and perform feature fusion or dimensionality reduction. Due to the usage of the trained parameters to

TABLE II
Symbol Error Rate (%) in Different Scenarios using the Original dataset and Data Augmentation Employing the VGG19 Model as a Feature Extractor

Batch size	Splitting strategy (%)	Original dataset	Original dataset+blurring	Original dataset+rotation	Original dataset+scaling
8	(90/10)	1.30	24.43	9.04	11.24
	(80/20)	1.1	12.95	13.37	12.98
	(70/30)	0.98	80.32	18.42	56.74
16	(90/10)	**0**	5.25	6.76	1.6
	(80/20)	0.06	1.45	1.19	3.85
	(70/30)	**0.04**	1.28	2.42	1.92
32	(90/10)	**0.03**	1.03	0.81	0.79
	(80/20)	**0.04**	0.35	0.57	0.15
	(70/30)	0.05	0.32	0.51	0.29

 http://dx.doi.org/10.14500/aro.11402

TABLE III
SYMBOL ERROR RATE (%) IN DIFFERENT SCENARIOS USING THE ORIGINAL DATASET AND DATA AUGMENTATION EMPLOYING THE RESNET50 MODEL AS A FEATURE EXTRACTOR

Batch size	Splitting strategy (%)	Original dataset	Original dataset+blurring	Original dataset+rotation	Original dataset+scaling
8	(90/10)	6.04	20.65	16.81	48.67
	(80/20)	6.56	19.49	26.19	16.56
	(70/30)	1.44	24.64	19.69	12.42
16	(90/10)	0.18	6.93	7.23	23.18
	(80/20)	0.17	0.14	0.04	2.24
	(70/30)	0.21	2.99	2.04	3.2
32	(90/10)	**0.03**	2.47	1.76	1.97
	(80/20)	**0**	0.87	0.33	**0.02**
	(70/30)	**0**	0.11	0.11	0.16

TABLE IV
SYMBOL ERROR RATE (%) IN DIFFERENT SCENARIOS USING THE ORIGINAL DATASET AND DATA AUGMENTATION EMPLOYING THE EXCEPTION MODEL AS A FEATURE EXTRACTOR

Batch size	Splitting strategy (%)	Original dataset	Original dataset+blurring	Original dataset+rotation	Original dataset+scaling
8	(90/10)	**0**	1.03	0.81	0.79
	(80/20)	0.71	10.57	0.39	0.14
	(70/30)	**0**	1.21	32.88	1.44
16	(90/10)	**0**	0.35	0.81	0.34
	(80/20)	**0**	0.14	**0.04**	2.24
	(70/30)	**0**	0.05	**0.1**	**0**
32	(90/10)	**0**	**0**	**0.02**	**0**
	(80/20)	**0.04**	**0**	**0.04**	**0**
	(70/30)	**0**	**0.02**	**0**	**0**

TABLE V
SYMBOL ERROR RATE (%) OF THE CUSTOMIZED CNN MODEL

Splitting strategy (%)	Batch size	Original dataset	Original dataset+blurring
(80/20)	16	30.46	55.2
	32	5.22	26.63

TABLE VI
SYMBOL ERROR RATE (%) OF THE OVERLAPPING SET USING THE XCEPTION MODEL AS A FEATURE EXTRACTOR

Splitting strategy (%)	Batch size	Original dataset
(80/20)	32	23.12

TABLE VII
SYMBOL ERROR RATE (%) OF THE OVERLAPPING SET USING THE CUSTOMIZED CNN AS A FEATURE EXTRACTOR

Splitting strategy (%)	Batch size	Original dataset
(80/20)	32	28.34

extract features instead of starting from scratch, training the model with our customized CNN needs more time compared with the usage of pre-trained CNN models to get feature maps.

There might be reasons why using data augmentation for OMR activities leads to producing poor results in comparison to other computer vision tasks. Here are some potential justifications:

1. OMR requires accurate identification of musical symbols and notations; augmentation procedures can alter structural integrity, reducing the model's ability to detect and comprehend notation effectively
2. Musical symbols require precise alignment and regular spatial connections, but random transformations during augmentation can reduce system performance
3. Various musical symbols require different augmentation techniques to preserve distinctive characteristics and requirements effectively.

D. Overlapping Symbols Situation

This study focuses on addressing overlapping musical notes. Therefore, a set of widely available images that contain two overlapping symbols is used. This set consists of 20 images collected from the Internet in different conditions. Then, every image was segmented into two images of individual symbols and performed all necessary steps before being fed to the model to predict its musical meaning. By making a trade-off between all

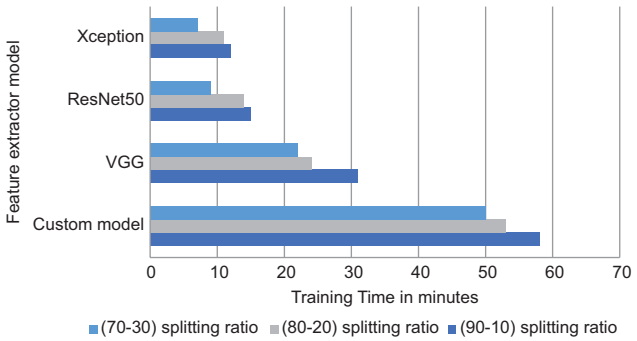

Fig. 6. The training time required for different feature extractor models and three various splitting ratios.

the previous models' architectures, the Xception feature extractor model is used in this experiment. Table VI shows the SER of testing the set of these segmented images. Then, our customized model has been involved in this experiment to earn more results for comparison and evaluation (Table VII). The original dataset was used to train the model.

As expected, the outcome of this experiment is below the desired result compared with testing non-overlapped symbols, because of the possible following reasons:

1. The images were collected from the Internet in different situations, conditions, and various numeric representations such as the thickness of the staff lines or the white space between them
2. Splitting the image which contains two overlapping symbols into two individual images and adding the missing lines might violate the image quality.

VII. Conclusion

In this study, an OMR system has been presented based on a deep learning approach including a CNN block to extract relevant features followed by seq2seq (encoder–decoder) to retrieve the semantic representation of a monophonic music sheet that is rendered on an image. It has been experimentally demonstrated that the suggested model can deal with long-sequence music notes by applying the proposed segmentation algorithm to the (PrIMS) dataset to create a new version of the dataset before being fed to the model. The achieved results indicate, that the usage of the Xception CNN block along with the seq2seq model outperformance other models with different CNN architectures (VGG, ResNet50, the customized CNN). Furthermore, the complexity of music notation has been addressed in this study, and a novel algorithm is proposed to detect and tackle the overlap problem on a set of images collected under different or non-identical conditions. It is worth mentioning that this work has some limitations, it is unable to handle polyphonic music sheets, and the overlapped symbols should be disjoint components and not have any touching pixels.

It has been proposed that the suggested framework might be used to challenges outside OMR that translates a spatial sequential representation to a set of labels, such as converting an auditory signal into a token sequence (an example of such a job would be optical character recognition).

Future work is suggested to create an OMR framework for converting polyphonic notes to musical meaning, addressing overlapping symbols including touching cases between the musical notes, and generating a training dataset for hand-written music symbols.

References

Baró, A., Badal, C., and Fornés, A., 2020. Handwritten Historical Music Recognition by Sequence-to-Sequence with Attention Mechanism. In: *2020 17th International Conference on Frontiers in Handwriting Recognition* (*ICFHR*). IEEE, United States, pp.205-210.

Brownlee, J., 2019. *Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python*. Machine Learning Mastery, Vermont.

Calvo-Zaragoza, J., and Rizo, D., 2018. End-to-end neural optical music recognition of monophonic scores. *Applied Sciences*, 8(4), p.606.

Calvo-Zaragoza, J., Valero-Mas, J.J., and Pertusa, A., 2017. End-to-End Optical Music Recognition using Neural Networks. In: *Proceedings of the 18th International Society for Music Information Retrieval Conference*. ISMIR, Canada, pp.23-27.

Castellanos, F.J., Calvo-Zaragoza, J., and Inesta, J.M., 2020. A *Neural Approach for Full-Page Optical Music Recognition of Mensural Documents*. ISMIR, Canada, pp.558-565.

Jang, M., Seo, S., and Kang, P., 2019. Recurrent neural network-based semantic variational autoencoder for sequence-to-sequence learning. *Information Sciences*, 490, pp.59-73.

Matrenin, P.V., Manusov, V.Z., Khalyasmaa, A.I., Antonenkov, D.V., Eroshenko, S.A., and Butusov, D.N., 2020. Improving accuracy and generalization performance of small-size recurrent neural networks applied to short-term load forecasting. *Mathematics*, 8(12), p.2169.

Michael, J., Labahn, R., Grüning, T., and Zöllner, J., 2019. Evaluating Sequence-to-Sequence Models for Handwritten Text Recognition. In: *2019 International Conference on Document Analysis and Recognition. ICDAR*. IEEE, United States, pp.1286-1293.

Mondal, R., Malakar, S., Barney Smith, E.H., and Sarkar, R., 2022. Handwritten English word recognition using a deep learning based object detection architecture. *Multimedia Tools and Applications*, 81, pp.1-26.

Neubig, G., 2017. Neural Machine Translation and Sequence-to-Sequence Models: A Tutorial. [arXiv Preprint] arXiv:1703.01619.

Pugin, L., 2006. *Optical Music Recognitoin of Early Typographic Prints using Hidden Markov Models*. ISMIR, Canada, pp.53-56.

Pugin, L., Burgoyne, J.A., and Fujinaga, I., 2007. *MAP Adaptation to Improve Optical Music Recognition of Early Music Documents Using Hidden Markov Models*. ISMIR, Canada, pp.513-516.

Rosen, K.H., 2007. *Discrete Mathematics and Its Applications*. The McGraw Hill Companies, United States.

Shatri, E., and Fazekas, G., 2020. Optical Music Recognition: State of the Art and Major Challenges. Computer Science, Engineering. [arXiv preprint] arXiv:2006.07885.

Sutskever, I., Vinyals, O., and Le, Q.V., 2014. Sequence to Sequence Learning with Neural Networks. In: *Advances in Neural Information Processing Systems*. Vol. 27. The MIT Press, United States.

Torras, P., Baró, A., Fornés, A., and Kang, L., 2022. Improving Handwritten Music Recognition through Language Model Integration. In: *4th International Workshop on Reading Music Systems*, p.42.

Van Der Wel, E., and Ullrich, K., 2017. Optical Music Recognition with Convolutional Sequence-to-Sequence Models. [arXiv preprint] arXiv:1707.04877.

Wen, C., and Zhu, L., 2022. A sequence-to-sequence framework based on transformer with masked language model for optical music recognition. *IEEE Access*, 10, pp.118243-118252.