Recognition of Handwritten Music Notes by CNN

Wei-Che Hsu
Department of Electrical
and Computer Engineering
Texas A&M University
weichehsu76@tamu.edu

Chu-Ching Hsu
Department of
Computer Science
Texas A&M University
jency267@tamu.edu

Abstract

We do the topic regarding "Recognition of handwritten music scores" [1] for the course project. In this topic, the input data contains hand-written sketch. The goal is to recognize types of music symbol in the input image with single symbol or in the music score with multiple symbols. The training data and testing data are from database Handwritten On-line Musical Symbols (HOMUS) [2] and CVC-MUSCIMA [3] [4]. Beyond this paper [1], we would like to apply convolutional neural network (CNN) to enhance the recognition accuracy of music notes.

1. Introduction and Related Work

1.1. Recognition of Handwritten Music Scores

This paper [1] proposes off-line software which is for recognizing handwritten music scores. It separates music notes into two group: one is isolated notes which are handled by learning method, the other is compound notes which are handled by hierarchical representation.

First, staff lines input images are removed which can be handled by [5]. Then, use the characteristic of staff lines crossing to select braces, and take some special aspect ratios of horizontal long connected components as ties. Preprocessing detects braces as well as ties, and remove them.

Second, to find each music symbol, it takes horizontal projection and vertical projection to do segmentation. By the intensity of projection signal to find the position of each symbol.

Third, detect how many notes in each symbol by mathematical morphology. If the number of notes is 0 or 1, take it as an isolated note. Otherwise, take it as a compound note.

Next, for isolated note, extract features by Zoning or Blurred Shaped Model (BSM) [6]. Both of these methods are like low-pass filter on the image to leave essential information in the format of feature vector. Training data use K-means to get K centroid of K group, and when testing this paper applies K-NN to classify testing note by K neighbors. The paper provides BSM regions = 7 and K = 9.

On the other hand, for compound note, remove bar lines before handle compound note. There are 2 characteristics for bar line: it crosses all staffs and it is without note heads. Check whether vertical lines have the similar length, and remove outliers which are high chance to be bar lines. Also, detect beams is important task to know the structure of compound note. when detecting consecutive note heads, this paper applies characteristic Loci Features [7] detection in the vertical direction to know the most possible number of beams. It also takes neighbor beam number information to enhance the correctness. Based on the position and the number of nodes, stems, and beams. Construct dendrogram by music notation rules from bottom to top to get the most possible compound note.

This paper has acceptable performance on isolated note recognition (about 60%-70%), but still not good at compound note recognition (about 40%-50%). The main problems behind this paper are the lower accuracy of head notes detection, and limited music notation rules.

1.2. Convolutional Neural Networks

Convolutional neural networks are a type of neurobiologically inspired feed-forward artificial neural network which consist of multiple layers of neurons, with neurons in each layer collected into sets [8]. CNNs are often used in recommender systems [9], natural language processing [10], image and sketch recognition. A CNN consists of a number of convolutional and subsampling layers optionally followed by fully connected layers. The input to a convolutional layer is a $m \times m \times r$ image where m is the height and width of the image and r is the number of channels. The convolutional layer will have k filters (or kernels) of size $n \times n \times q$ where n is smaller than the dimension of the image and q can either be the same as the number of channels r or smaller and may vary for each kernel. The size of the filters gives rise to the locally connected structure which are each convolved with the image to produce k fea-

ture maps of size $mn + 1$. Pooling layers down-sample the image data by partitioning the input image into a set of non-overlapping rectangles, and, for each sub-region, output the maximum. Pooling layers serve to decrease processing time and control overfitting. Fully connected (Dense) layers perform classification on the features extracted by the convolutional layers and down-sampled by the pooling layers. In a dense layer, every node in the layer is connected to every node in the preceding layer.

One of the well-known CNN for image recognition is ImageNet CNN [11]. Figure 1 shows the layer structure of ImageNet CNN. It is a 8-layer network with 5 convolutional layers and 3 fully connected layers. The output of the last layer is connected to a 1000-way softmax, thus producing a distribution of the 1000 class labels.

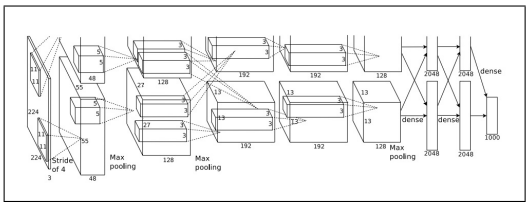

Figure 1. Layer Structure of AlexNet [11]

2. Major Body - Our methods for this project

2.1. System Design

Our goal is to predict the meaning for all symbols on music scores like Figure 2. Isolated symbols can be easily assigned label to them, but it is hard for compound symbols. To avoid infinite labels for compound symbols, we refer the representative in the paper [12], to separate compound symbol to isolated symbols. Thus, we use duration and pitch to represent the meaning of each symbol.

Our program flow is shown in Figure 3. In the training flow, input image contains hand-written sketch without staff lines. Then, in the preprocessing block, because test data may come from different datasets, we have to scale the size of image to the fixed size 64×64 as CNN input. Next, train this CNN model until it has good enough performance, and we can use this model for prediction flow.

In the prediction flow, when input is a line of music score, it contains many symbols, and it can apply projection method to get meaningful segment as the input of CNN. Also, when get the segmentation of music score, the position of staff line has to record for pitch prediction block. Next, CNN takes fixed size image without staff line as input, and set one of 32 possible labels from training set as output. In the pitch prediction block, it considers the position information of symbol ,the corresponding image with staff lines, and the note mask convolution value, so it is possible to decide the correct pitch of symbols in a music score.

For music score, like Figure 2, we will input the score line by line, and try to predict the meaning of each symbol. In the following subsections, we will describe our method for each block and propose some notable points.

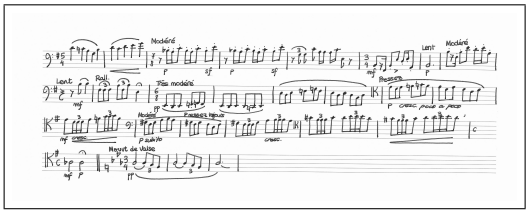

Figure 2. Example of music score in dataset [3].

2.2. Preprocessing Block

The datasets we used for model training, validation, and testing is a combination dataset derived from HOMUS [2]and MUSCIMA++[3, 4, 13]. HOMUS is a dataset collection consists of over 15200 hand written common music symbols encoded in the format of x, y, and temporal coordinates of each sketch data. In order to translate the coordinates into data we could use for training and testing, we use our own parser program written in Python language to parse the data format of HOMUS and translate it into 2D RBG image files. MUSCIMA++ is another relatively new dataset collection. The data collection of MUSCIMA++ is derived from a previous dataset CVC-MUSCIMA which collects over 1000 hand written music sheet. MUSCIMA++ derived the data from CVC-MUSCIMA and annotated each music symbol in the music sheet, and encoded the position, mask, relation between other music symbols, and category property of each music symbol into XML format. We also wrote specified XML parser to translate MUSCIMA++ XML data into 2D RGB images which we needed as test case and training data, such as images without staff, images with pure symbols, and etc.

For training CNN model, we convert image data into TFRecord file by the script on TensorFlow official github [14]. This file type is official format, and TensorFlow provides many useful functions to support TFRecord input, like shuffling and refilling ran-out batch buffer. We refer these functions and TFRecord Reader from [15] and [16]. The reader transfer data from RGB format into Gray level format, and the magnitude of every pixel is between 0 and 1. Because HOMUS provides labeled dataset, Our main dataset is from HOMUS, separating 15714 samples as 13654 samples in training set and 1520 samples in validation set. For generalize it, we generate the similar pattern but with twice thickness sketch for every sample image. Thus, there are 27308 samples in training set and 3040 samples in validation set. Also, these sets include additional samples from CVC-MUSCIMA, which will be explained in

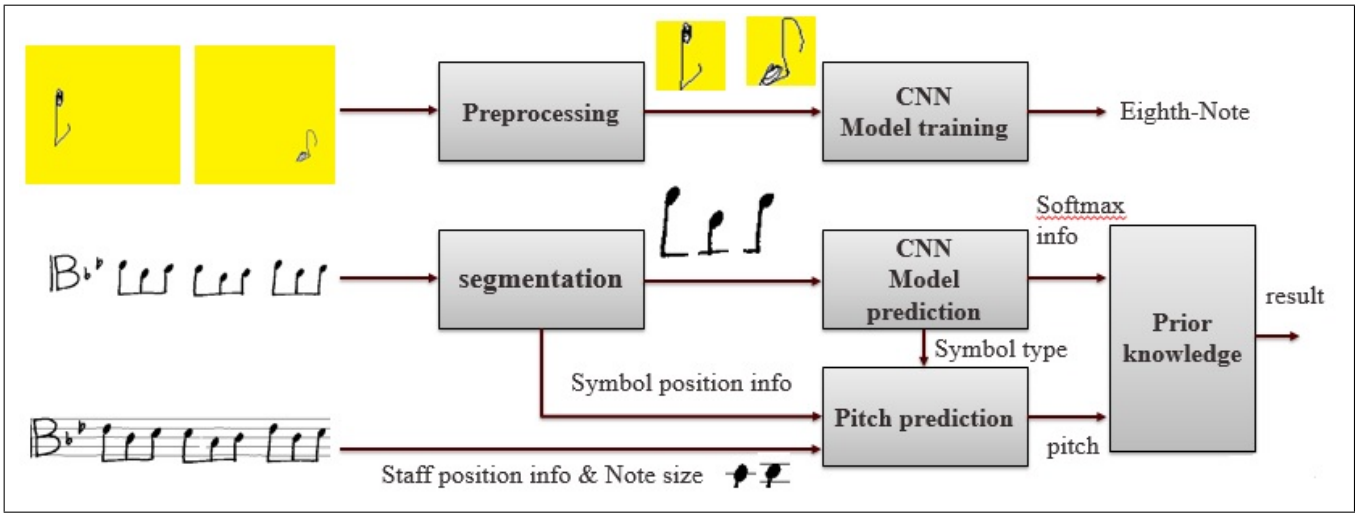

Figure 3. Flow diagram

the later section. Finally, the training dataset includes 27730 samples and the validation dataset is 3090 samples with 32 labels.To test its generalization, we collect additional 1198 testing samples from CVC-MUSCIMA which only includes eighth note, sixteenth note, and thirty two note. We would like to test our model against the frequent segmentation in compound symbol cases. In addition, we add 1520 samples original HOMUS validation set. Thus, this mixture validation set includes 2718 testing samples. Now, we have a training set and two validation sets for CNN model, and the training result will show in Experimental Result section.

2.3. Convolutional Neural Network Block

Initially, we follow the setting as Table 1 to build CNN. The developing language is Pyhton with Tensorflow package. Although its validation accuracy is about 87%-90% for HOMUS dataset, this model performs not well for unseen test data in CVC-MUSCIMA, only 20%-30%. This means the training model is over-fitting to its training dataset and the likely validation set though the validation rate still doesn't drop dramatically. Because CVC-MUSCIMA dataset is without label, we generate the CVC-MUSCIMA test data by linking the line and note which has relationship in MUSCIMA++. However, these symbols usually bring longer tail or more thick sketch. The appearance comparison between HOMUS and CVC-MUSCIMA is shown in Figure 4. We try to find methods to improve this problem. There are 4 methods we adopt for this problem.

First, we take data augmentation to avoid over-fitting easily, when batch images are taken, before input to CNN model, they are applied random data augmentation: ratio scale, offset, rotate, and do one of below changes: blurring or adding Gaussian noise on image, or doing nothing. The example for sixty-four note is shown in Figure 5: A is original 64×64 input. B is from randomly scaling A between 64×64 and 32×32 but retain height-width ratio, and do random offset but stay all symbol in the range of 64×64, padding zero to its background. C is from rotating B between 15 to -15 degrees. After that, some C images will do blur to become D with random variance. Some of C images

setting	value
conv layer 1	16 filters (5×5)
maxpool	half$\times$ half
conv layer 2	36 filters (5×5)
maxpool	half$\times$ half
conv layer 3	64 filters (3×3)
maxpool	half$\times$ half
conv layer 4	128 filters (3×3)
maxpool	half$\times$ half
fc layer 1	1024 neurons
output layer	32 labels
softmax	
drop-off	0.5
learning rate	1E-4
activator	ReLU
Batch size	128

Table 1. Previous CNN setting

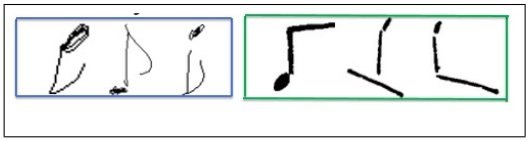

Figure 4. Eighty notes comparison: Left group is from HOMUS. Right group is from CVC-MUSCIMA

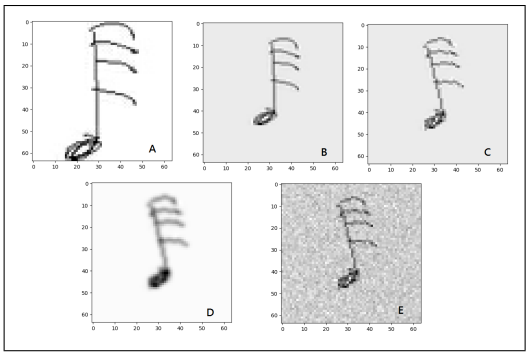

Figure 5. Example of data augmentation: A,B,C,D,E are described in the third method in subsection 2.3.

are added additional Gaussian noise with random variance as well as 0 mean to become E. There are some images stay the same as C. The sigma value for blur is between 0.2 and 1 as well as the variance for Gaussian noise is between 0.001 and 0.005. I control the random range which is not confusing human vision recognition. After doing so, the model can train for longer time without over-fitting and improve 1%-5% for CVC-MUSCIMA dataset.

Second, we add CVC-MUSCIMA type data into my training data, but not use them to do any test or validation. The previous training dataset includes 27308 training data from HOMUS, which we use two different sketch width when producing these images. Then, I only add 422 images including 270 symbols at the types of eighth note, sixteenth note, and thirty two note as well as 152 whole notes on the new training dataset. Although 422 new images take little ratio of overall training set, it remarkably increase the prediction accuracy for CVC-MUSCIMA dataset to 60%. This means that additional samples bring important features of CVC-MUSCIMA for CNN model to learn.

Third, we try to reduce the level of CNN because we don't have large number of training samples and only 32 output labels. More levels bring high level feature extraction, but we need more basic level features to fit CVC-MUSCIMA dataset rather than high level features from HOMUS. After tuning, we decide to use 2 level CNN structure, and the parameters will be provided in Table. 2. However, with less maxpool layers, CNN model has more parameters to adjust because there are more channels before fully connected layers. Combined this method with the first method, we can get a more general prediction model.

Last, when I measure the loss value, I found that accuracy become very low when the value of loss function is nan. I see the discussion on the Internet, and know the reason of nan results from overfitting, because all output labels are close to 0. If the loss function is cross entropy, logarithm of output vector with nearly zero value will cause very large negative value, which causes nan through back-propagation to the trained model everywhere. I see some solutions like reduce learning rate or add very small offset on softmax output. The second solution may sacrifice little accuracy from the best result but ensure more training steps. In my problem, I don't have high enough accuracy for CVC-MUSCIMA when nan happens. Thus, choose the second solution to run more training steps to gain the accuracy. This method provides us 10%-15% accuracy improvement on HOMUS and CVC-MUSCIMA dataset. To sum up all changes from these 4 methods, the setting of our final CNN model is in Table 2.

setting	value
conv layer 1	32 filters (3×3)
maxpool	half$\times$ half
conv layer 2	140 filters (3×3)
maxpool	half$\times$ half
fc layer 1	1024 neurons
output layer	32 labels
softmax	1E-8 offset
drop-off	0.5
learning rate	1E-4
activator	ReLU
Batch size	128
data augmentation	A,B,C,D,E in Figure 5

Table 2. Final CNN setting

2.4. Segmentation Block

As the first step of the overall system flow, the basic segmentation of music symbols serves as an important role in the whole system. Since we put our main focus on the CNN model training and recognition, the segmentation we implemented will be based on the projection techniques [17]. At the first step of segmentation, in order to segment basic unit of music symbol without errors caused such as trivial discontinuous sketches, we applied Gaussian filter to the x axis projection as shown in Figure 6. The Gaussian filtered projection will then be segmented into sections with continuous non zero value. As the first stage of segmentation is done , the second segmentation aka compound music symbol segmentation begins.

The challenges in segmentation of compound music symbols comes as follows: 1. Projection of compound music symbols are all continuous non zero values, therefore cant apply non zero filter. 2. The depth and height in the music symbols are different. 3. How to tell if one is a compound music symbols?

In regards of challenge 3, we use a simple detection function to tell if a segmented music symbol might be a compound music note or not. The detection is based on 2 fact: 1. A compound music note must have two or more stem.

2. Distance between the left-most note head and the right-most note head takes big proportion of the overall width of the note. Using these 2 ground truth, we detect if the Gaussian filtered projection has more than two peaks which has height value greater than threshold, and then check if the distance between these two peaks is over the threshold value or not. To solve challenge 1 and 2, like solving challenging 3, the compound music note is segmented based on the valley in the projection. The valley of the projection represents the stem in between each note head, and stems will be always in between two note heads. The over segmentation flow will be : 1. Basic segmentation of unit music symbol. 2. Check if the unit symbol is a compound music note. 3. If it is, do second segmentation based on the valley of the projection.

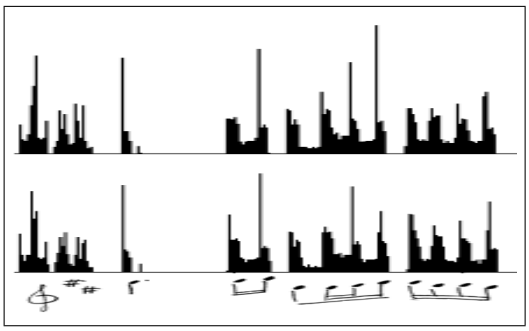

Figure 6. The above image is the original music sheet image. The top image is the original x axis projection of the bottom image. The middle image is the Gaussian filtered image of the top image

2.5. Pitch Prediction Block

Besides pure music symbol recognition, such as full-note, G-clef, dot, etc., we also add basic pitch detection function into our system. Pitch detection function is called after the CNN pure music symbol recognition is done in the flow. Pitch detection will only be called when the sketch is recognized as a Note by the CNN model classifier. The whole detection of pitch requires four inputs: two original images, one with staff lines and one without staff lines, as shown in Figure 7, the recognition result from CNN model, and the note head convolution result. The note head convolution result is the return value from function note-head convolution which is described in its own section. Function note-head-convolution returns the height with the highest value of the location of the note head. The original images with and without staff lines serve as providing staff lines positions. We filter the music symbols from staff line images and get the highest and lowest position of y axis projections to get the relative position of 5 staff lines. Once we get the position of top staff line and bottom staff line we get height position of each staff line by evenly divide the height from the top staff line to the bottom staff line. We divide such value into 9 section, based on the relative order of 5 staff lines and the blank between them, named as first staff line and first blank relatively. Since the absolute pitch, such as C, D, E not only based on the position of the note head and staff line, but also depends on the clef. For the pitch detection function we focus on finding the relative position between staff lines and note heads.

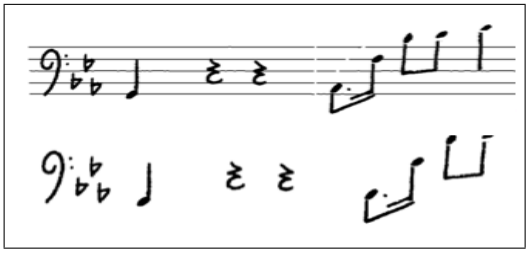

Figure 7. Example of original images with and without staff lines

Next, we describe how convolution method finds the note position in each segment to decide the pitch for this segment. We cut some notes from CVC-MUSCIMA score, some is black note, and some is white note Depends on CNN prediction result to use which type node. This notes are called as note masks. Then, the testing segment does convolution with the corresponding masks, and the maximum value point in the convolution result is the estimation of the note center. In each 2D convolution, the mask doesn't have to reverse because we only care the shape correlation. If there are multiple positions with maximum value, we just select the first position as the estimation result. In addition, it is an ensemble scheme because we use multiple masks to take the average value form its result.

3. Experimental Results

We separate the experimental results as two parts to discuss our system, one is training flow, and the other is prediction flow.

3.1. Training Flow: CNN Model Performance

To analyze CNN performance, we record the loss and the accuracy from CNN model every 100 iterations, each iteration includes 128 training samples. The curve of "HOMUS Train" only means the measurement of 128 training samples in the record iteration. The curve of "HOMUS Valid" means the measurement for overall 3090 validation samples. The curve of "Mix Valid" means the measurement for 2718 mixture validation samples. We choose the model with the highest accuracy for the mixture validation set to obtain the general model for prediction flow.

There are three interesting point in Figures 8 and 9, I would like to describe: first, before I apply softmax offset method, our model training usually early stop at near the

10000th iteration where is still on the slope to the better area. Through adding softmax offset, the model can gain performance because the average accuracy of each curves in Figure 8 at 60000th iteration is higher than that in each curves at 10000th iteration. Second, we use a little fierce modification in data augmentation, so the curve "HOMUS Train" sway fiercely than other 2 curves. To improve convergence of 3 curves, we can loose the magnitude of data augmentation or increase the training batch size. Due to argumentation, we can observe that the curve of two validation sets improve faster than the curve of training set. Last, after training 60000 iterations, the constant difference between the curve of "HOMUS Valid" and the curve of "Mix Valid" shows that our method of using additional samples form CVC-MUSCIMA still can't represent all important features of CVC-MUSCIMA. I believe the difference can be smaller with more additional samples from CVC-MUSCIMA.

To estimate accuracy for only CVC-MUSCIMA set, there are 1520 HOMUS samples in mixture validation set. The number of remaining samples from CVC-MUSCIMA is 1198. It is about 93.7% accuracy for this 1520 samples because they are also shown in HOMUS validation set. The model with 86.2% accuracy for "Mix Valid" set should predict 2343 samples correctly. we subtract $1520 \times 93.7\%$ from it. The model should predict 919 samples correctly within 1198 remaining samples. Thus, accuracy for CVC-MUSCIMA dataset is about 76.7%, which is much better than the performance of our previous model in Table 1.

According to my record for Figure 8, our trained model, under the setting as Table 2, achieves 86.2% accuracy for"Mix Valid" set, and 93.7% accuracy for HOMUS validation set at the same iteration. Although the validation set is different with the paper [18] because we can't get exactly 10-fold validation sets they used, we still show the result in Table 3 compare to the performance of [18]. Our model can achieve 95% accuracy for our HOMUS validation set, but we choose the model with the best performance for "Mix Valid" set when the model is training. For HOMUS dataset, the performance of our selection model is close to the performance of AlexNet structure as shown in Table 3. For HOMUS set with 32 labels, We only use 2 convolution layers but apply many methods from AlexNet to improve our model to get the similar result of AlexNet. For this dataset, maybe we don't need very deep CNN structure.

3.2. Prediction Flow Test

For the prediction test, we used 653 one line music sheet as our test data. To evaluate our overall system performance, we discuss the prediction result and accuracy over two step of the system: the segmentation part and pitch prediction part. The overall accuracy of the segmentation part is shown in Table 4. The result is divided into 3 categories: right segmentation, false segmentation that can be further divided into 2 sub categories: wrong segmentation and missing segmentation. Wrong segmentation represents segmentation of symbols that should not be segmented, and missing segmentation represents music symbols that should be segmented but did not get segmented. Through the repeated test flow we discovered situation that could lead to false segmentation 1 and 2. False segmentation 1 usually occurs when a music symbol is written too sparsely. Figure 10 shows possible music symbols that usually falls into this category. Music symbols that do not found in Figure 10 but occur in false case number 1 could also be the

Model	Accuracy
DTW	77%-85%
MLP	76%-80%
CifarNet	84%-90%
AlexNet	94%
GoogleNet	95%
Our CNN model	93.7%

Table 3. HOMUS dataset: Accuracy Comparison with [18]

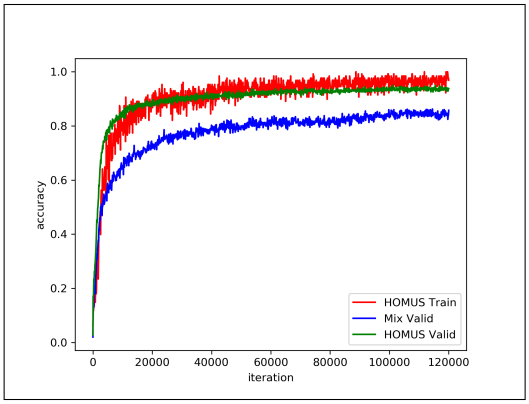

Figure 8. accuracy_iteration

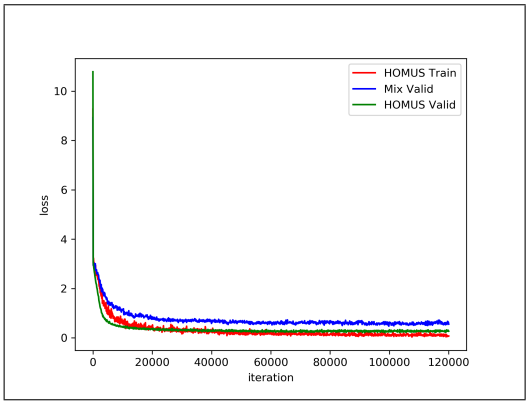

Figure 9. loss_iteration

sheet writers personal style problem. False segmentation 2 mostly happens to compound segmentation. According our detection and segmentation algorithm of compound music symbol, it could happen if the detection fails to recognized a compound music note if : 1. The length of the stems of the compound music note is relatively too short. 2. If the distance/width of the compound music segmentation is too short. Another situation that falls into such situation is because of too little distance between two music symbols, or to be more specifically, music symbols that falls on positions that their x axis projection overlapped. Such case are shown in Figure 11.

The pitch prediction is the combination result derived from CNN model and note head convolution. The output of the system consists of the order number of the corresponding music symbol, the category of the music symbol, the relative position between staff line and the music symbol. Overall the prediction system have high accuracy on flat, single quarter note, single eighth note, single rest symbols, G and F clef, and dot. Symbols that have higher possibility of detection failure are compound music notes, C clef. The reason behind these could be that compound music notes varies much more than the training sets we used in training our CNN model. and so is C clef, since C clef has various way in hand written, and due to its characteristic, which could be seen in Figure 10, it has relatively high possibility to be identified as compound music note and therefore being segmented and cause higher level of difficulty in recognition for CNN model.

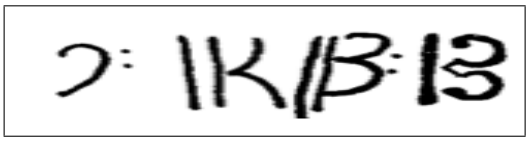

Figure 10. common music symbols that often seen as separate music symbols

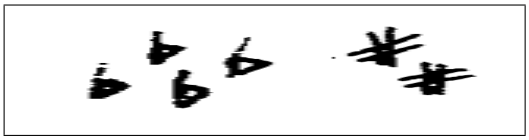

Figure 11. common music symbols that have overlapped x axis projection

Right Segmentation	91.91%
False Segmentation 1	5.26%
False Segmentation 2	2.81%

Table 4. Accuracy of segmentation

Index	Prediction	Type	Pitch
0	Eighth-Note third blank	F	F
1	Sharp	T	T
2	6-8-Time	T	T
3	Eighth-Note first blank	T	T
4	Sixteenth-Note second blank	T	T
5	Sixteenth-Note second blank	T	F
6	Eighth-Note first blank	T	T
8	Eighth-Note forth line	T	T
9	Eighth-Note third blank	T	T
10	Eighth-Note second blank	T	T
11	Eighth-Note 1st line	T	T
12	Sixteenth-Note second line	T	T
13	Sixteenth-Note first blank	T	T
14	Eighth-Note 1st line	T	T
15	Eighth-Note third blank	T	T
16	Eighth-Note third line	T	T
17	Quarter-Note second line	F	T
18	Eighth-Note beyond top staff	T	T
19	Sixteenth-Note first blank	T	T
20	Eighth-Note 1st line	F	T
21	Eighth-Note beyond top staff	T	T
22	Eighth-Note third line	T	T
23	Eighth-Note second blank	T	T
24	Quarter-Note beyond top staff	F	T
26	Eighth-Note second blank	T	T
27	Eighth-Note second line	T	T
28	Eighth-Note beyond top staff	T	T
29	Eighth-Note second line	T	T
30	Eighth-Note first blank	T	T
31	Eighth-Note beyond top staff	T	T
32	Eighth-Note third line	T	T
33	Sixteenth-Note second blank	T	T
34	Sixteenth-Note second line	T	T
35	Sixteenth-Note first blank	T	T
36	Sixteenth-Note 1st line	T	T
37	Eighth-Note beyond top staff	T	T
38	Eighth-Note first blank	T	T
39	Quarter-Note second line	F	T
40	Quarter-Note first blank	F	T
41	Eighth-Note third line	T	T
42	Eighth-Note forth line	T	T
43	Quarter-Note below bottom staff	T	T
Total	Accuracy for 42 predictions	85.7%	95.2%

Table 5. Test result of prediction flow in Figure 12

4. Conclusion

In this project, we produce dataset, train CNN model, design the system flow, and implement each block. Learn how to use many useful Python packages: TensorFlow, Pillow, NumPy, SciPy, and detect_peaks to develop our pro-

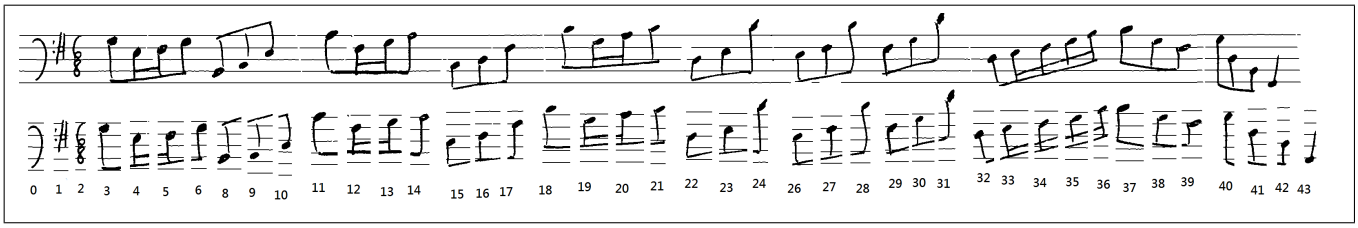

Figure 12. Music sheet and its segmentations for 45.png

gram. Our code is on [19]. Finally, for CVC-MUSCIMA music sheets, we have the prediction system which can recognize most symbols on some music sheets like the case of Table 5 although there still are complex music sheets which our model can't predict very well. For HOMUS dataset, our CNN model which only has 2 convolutional layers can achieve 93.7% accuracy nearly to AlexNet.

Idea	Both
Report and Presentation	Both
Preprocessing Block	Both
CNN model Block	Wei-Che
Segmentation Block	Chu-Ching
Pitch Block	Both

Table 6. Task Division

References

[1] A. Baro and et al, "Towards the recognition of compound music notes in handwritten music scores," *15th International Conference on Frontiers in Handwriting Recognition*, 2016.

[2] J. Calvo-Zaragoza and J. Oncina, "Recognition of pen-based music notation: The HOMUS dataset," *22nd International Conference on Pattern Recognition*, pp. 3038–3043, 2014.

[3] A. Forns, A. Dutta, A. Gordo, and J. Llads, "CVC-MUSCIMA: A ground-truth of handwritten music score images for writer identification and staff removal," *International Journal on Document Analysis and Recognition*, vol. 15, pp. 243–251, 2012.

[4] J. H. jr. and P. Pecina., "The MUSCIMA++ dataset for handwritten optical music recognition," *14th International Conference on Document Analysis and Recognition, ICDAR 2017*, pp. 39–46, 2017.

[5] M. Visani, V. C. Kieu, A. Fornes, and N. Journet, "ICDAR 2013 music scores competition: Staff removal," *12th International Conference on Document Analysis and Recognition, ICDAR 2013*, pp. 1407–1411, 2013.

[6] A. Fornes, S. Escalera, J. LLados, P. R. G. Sanchez, and O. Pujol, "Handwritten symbol recognition by a boosted blurred shape model with error correction," *Iberian Conference on Pattern Recognition and Image Analysis*, pp. 13–21, 2007.

[7] J. Llados, M. Rusinol, A. Fornes, D. Fernandez, and A. Dutta, "On the influence of word representations for handwritten word spotting in historical documents," *International journal of pattern recognition and artificial intelligence*, vol. 26, no. 5, p. 1263002, 2012.

[8] R. K. Sarvadevabhatla and R. V. Babu., "Freehand sketch recognition using deep features," *arXiv preprint arXiv:1502.00254*, 2015.

[9] A. van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," *Advances in Neural Information Processing Systems (NIPS)*, pp. 2643–2651, 2013.

[10] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," *Proceedings of the 25th International Conference on Machine Learning. ICML '08*, pp. 160–167, 2008.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, 2012.

[12] E. van der Wel and K. Ullrich, "Optical music recognition with convolutional sequence-to-sequence models," *arXiv:1707.04877*, 2017.

[13] P. P. Jan Haji jr., "In search of a dataset for handwritten optical music recognition: Introducing MUSCIMA++," *CoRR, arXiv:1703.04824 [online]*`https://arxiv.org/abs/1703.04824`, 2017.

[14] niyazpk and nealwu, "Build image data `https://github.com/tensorflow/models/blob/master/research/inception/inception/data/build_image_data.py`," 2017.

[15] A. Gray, "Demystifying data input to tensorflow for deep learning `https://agray3.github.io/`," 2016.

[16] Daniil, "Tfrecords guide `http://warmspringwinds.github.io/tensorflow/tf-slim/2016/12/21/tfrecords-guide/`," 2016.

[17] S. Marinai and P. Nesi, "Projection based segmentation of musical sheets," *Document Analysis and Recognition, ICDAR 1999*, 1999.

[18] J. O. S. Lee, S. J. Son and N. Kwak, "Handwritten music symbol classification using deep convolutional neural networks," *2016 International Conference on Information Science and Security (ICISS)*, pp. 1–5, 2016.

[19] W.Hsu and C.Hsu, "Code and dataset for csce633 project," `https://goo.gl/XPGxvp`, 2018.