SCORE DIFFICULTY ANALYSIS FOR PIANO PERFORMANCE EDUCATION BASED ON FINGERING

Pedro Ramoneda, Nazif Can Tamer, Vsevolod Eremenko, Xavier Serra, Marius Miron*

Music Technology Group, Universitat Pompeu Fabra, Barcelona

ABSTRACT

In this paper, we introduce score difficulty classification as a subtask of music information retrieval (MIR), which may be used in music education technologies, for personalised curriculum generation, and score retrieval. We introduce a novel dataset for our task, Mikrokosmos-difficulty, containing 147 piano pieces in symbolic representation and the corresponding difficulty labels derived by its composer Béla Bartók and the publishers. As part of our methodology, we propose piano technique feature representations based on different piano fingering algorithms. We use these features as input for two classifiers: a Gated Recurrent Unit neural network (GRU) with attention mechanism and gradient-boosted trees trained on score segments. We show that for our dataset fingering based features perform better than a simple baseline considering solely the notes in the score. Furthermore, the GRU with attention mechanism classifier surpasses the gradient-boosted trees. Our proposed models are interpretable and are capable of generating difficulty feedback both locally, on short term segments, and globally, for whole pieces. Code, datasets, models, and an online demo are made available for reproducibility.

Index Terms— Difficulty Analysis, Piano Technique, Music Classification, Piano Fingering, Symbolic Music Processing & Corpora

1. INTRODUCTION

Classification of music corpora is a problem well-studied under Music Information Retrieval (MIR), which is often targeted from the listeners' perspective as exemplified in genre/style [1, 2] and emotion [3, 4] classification. In a paradigm shift, music may be classified from the point of view of the performer by focusing on the required performance skills [5, 6], which is a newly emerging field of study. This paper focuses on music classification of performance difficulty, with applications in the formation of large pedagogical score databases, personalised score recommendation systems, and as an aid to both individual instrument learners and music teachers. Towards helping the students in determining where to focus their effort, and thus, increasing the efficacy in self-induced music studies, we aim at giving feedback on relative local difficulty over multiple segments of a piece.

The difficulty, by definition, is a subjective measure with multiple dimensions (i.e. expressivity, rhythm, sound, and technique [7]). Publishers, examination boards, and online repositories classify scores based on performance difficulty. Many exam boards relate these rankings with the school grades. A widely known publisher, Henle, releases piano difficulty rankings for their scores in the range 1-9, but the range is more tailored towards professionals. Certain instrument forums and websites such as 8note also release difficulty grades in different ranges, generally three or four grades from beginner to advanced. When it comes to the difficulty datasets in MIR, both previous datasets [8, 5] are not publicly available.

As a first contribution, we release Mikrokosmos-difficulty [1] – a benchmark dataset for piano score difficulty analysis derived from a corpus of 147 educational pieces authored by Béla Bartók for use in piano education. To our best knowledge, this is the first open dataset containing piano scores ranked in terms of difficulty and matching different technical levels where all the scores are composed by a single composer. Alongside the data, we also provide three different difficulty labels to study the subjectivity of performance difficulty: the first labels are the order of pieces generated by the composer himself, the second is the book divisions by the original publisher, and the third is difficulty labels in the range 1-9 by the publisher Henle, respectively. Since all the scores are composed by the same composer, the difficulty rankings are less prone to style biases, and more focused on technique difficulty.

As a second contribution, we introduce several piano technique features and two classification algorithms capable of giving both score-level and segment-level difficulty feedback [2], whilst being trained solely using score-level labels. To that end, we model the score with a novel feature representation based on piano fingering, and taking that as the input, we propose two difficulty classification methods: (i) gradient-boosted trees [9] applied to short-term segments and (ii) GRU neural network [10] with an attention mechanism. We want the selected classifiers to give feedback related to the local difficulty of a piece allowing students and teachers to focus and improve on the most difficult passages. Further, we provide a corpus visualisation tool for exploring local difficulty representations derived from both attention weights and the segment-level classification models. The remainder of this paper is organized as follows: in Section 2, we give the previous related work, in Section 3 we introduce the Mikrokosmos-difficulty dataset, in Section 4 we propose our difficulty analysis methods and give the results of our experiments in Section 5.

2. RELATION WITH PREVIOUS WORK

In this section we refer to the main methods to model difficulty in piano repertoire [8, 5, 11]. Sebastien et al. [8] propose a list of different instrument-agnostic descriptors for difficulty classification. The list of descriptors was further extended by Chen et al. [5] proposing different feature spaces for measuring difficulty in the piano repertoire. Although a pitfall is that neither approach is reproducible without

*Corresponding author: pedro.ramoneda@upf.edu

[1] Dataset available at:
https://github.com/PRamoneda/Mikrokosmos-difficulty
DOI: 10.5281/zenodo.6092709

[2] Code and models available at:
https://github.com/PRamoneda/ICASSP22

the code or datasets available, the main drawback is that they use instrument-agnostic attributes. In contrast, in the present research, we explore features related to the piano playing technique. Another work targeting piano difficulty is proposed by Nakamura et al. [11, 12] and deals with fingering frequencies and playing rate. The rationale for this proposal is that piano fingerings which occur less often, lead to increased difficulty. This method is extended to other tasks such as polyphonic transcription [13], rhythm transcription [14] or score reductions [15, 16], making clear the importance of piano technique in the creation of music technology systems.

In this research, we aim at exploring several drawbacks of this approach [11], while looking at piano fingering as a proxy for difficulty. In contrast to their work which uses a concert-oriented dataset, we use a pedagogically motivated dataset. In addition, we classify overall difficulty and not solely instantaneous difficulty. Furthermore, we propose a thorough evaluation which is lacking in the existing literature [11]. Nevertheless, we also use the Nakamura et al. approach [11] to derive piano technique related features used as input for the machine learning models.

3. MIKROKOSMOS-DIFFICULTY DATASET

Béla Bartók's *Mikrokosmos* Sz. 107 is a collection of 153 piano pieces published in six volumes between 1926 and 1939 and progressively ranked in terms of difficulty. Individual compositions go from extremely simple and easy beginner exercises to challenging advanced technical musical works.

	Beginner	Moderate	Professional
n scores	62	54	31
n notes	$\mu = 108.58$ $\sigma = 57.16$	$\mu = 260.40$ $\sigma = 111.00$	$\mu = 650.06$ $\sigma = 322.15$
n bars	$\mu = 20.37$ $\sigma = 9.64$	$\mu = 33.35$ $\sigma = 13.27$	$\mu = 63.12$ $\sigma = 29.30$
tempo	$\mu = 107.25$ $\sigma = 24.48$	$\mu = 112.62$ $\sigma = 59.75$	$\mu = 182.45$ $\sigma = 113.13$

Table 1. Means μ and the standard deviations σ for the number of scores (n scores), notes (n notes), bars (n bars) and tempo across the three levels of difficulty in the Mikrokosmos dataset.

We propose the Mikrokosmos-difficulty dataset based on a subset of 146 pieces from the Mikrokosmos collection, discarding the 7 pieces composed for four hands performance. We group the pieces on three levels of difficulty according to the original classification in the the editions by the publishers Wiener and Henle Verlag. Correspondingly, the beginner level contains the pieces 1-66 (Volumes I and II), the moderate level contains the pieces 67-121 (Volumes III and IV), the professional level contains the pieces 122-153 (Volumes V and VI). The 6-volume-long collection is obtained from IMSLP in pdf format. The conversion to machine-readable symbolic representations in musicXML was done semi-automatically using a commercial Optical Music Recognition (OMR) software [17], and manual corrections. The statistics are presented in Table 1, where we can see that Mikrokosmos-difficulty dataset is biased in lengths and tempo. Therefore, analyzing the technique difficulty in this dataset requires robustness to variations in length and tempo.

4. METHODOLOGY

We present five piano technique-based feature representations to analyse the performance difficulty and an additional baseline using solely the notes. Moreover, we use these features as input for two machine learning classifiers. The two methods are interpretable, and we may derive important feedback for music education applications.

4.1. Feature representation of the piano technique

Automatic piano fingering systems aim to describe the movements of hands and fingers on the piano departing from the score. This task is related to piano technique and a proxy to modelling the difficulty of playing a score. The current approaches in piano fingering go from expert systems [18] to local search algorithms [19, 20] and, more recently, data-driven methods [11, 12]. Towards modelling difficulty, we derive piano technique features from two piano fingering approaches, a knowledge-driven system, Pianoplayer [20], and a data-driven system proposed by Nakamura et al. [12].

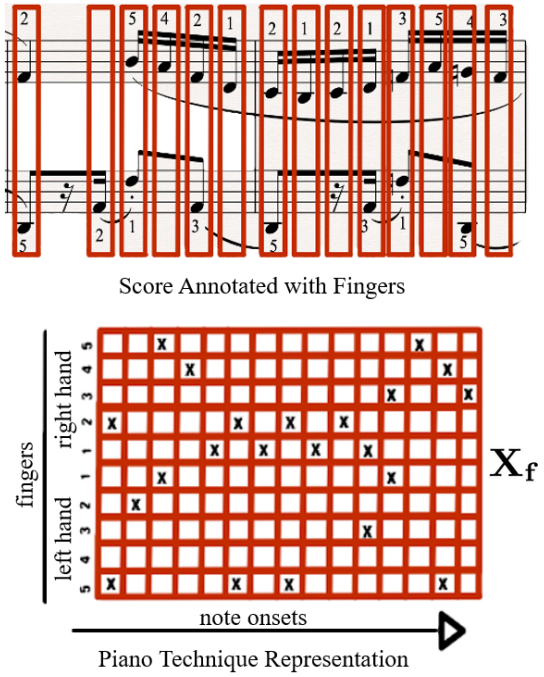

Fig. 1. An example of piano fingering computed for a score (top) with Pianoplayer and Nakamura et al. [12] for which we derive the features P-F, P-V, N-F, N-P, and then the corresponding matrices $\mathbf{X_f}$ (bottom).

Pianoplayer [20] is a local search algorithm grounded in dynamic programming and combinatorial optimisation, openly available under an MIT license. It computes which finger is playing each note by considering the following nine notes and the previous positions of the fingers. Pianoplayer carries out the optimisation according to hard-coded constraints and a cost function related to the finger velocity. From the cost function, we derive two features associated with each note: the finger which plays the note, P-F, and the velocity associated with the finger, P-V.

Nakamura et al. [12] train a Hidden Markov Model (HMM) on a 150 scores dataset annotated by more than four professional pianists – data, code and models are available for noncommercial activities.

In contrast to Pianoplayer which has hard-coded rules, this model learns implicitly the transition probabilities that model finger movements. We use the transition probability given by the HMM model to derive two features for a given note: the finger which plays the note, N-F, and the transition probability associated to the note, N-P.

Given a piano score and the features associated with each note P-F, P-V, N-F, N-P, we then form the matrices $\mathbf{X_f}(i, j)$ containing all the notes for each of the four features, where $i = 1 \dots I$ and I is the total number of note onsets in the piece, and $j = 1 \dots 10$ are the fingers. In order to reduce the size of $\mathbf{X_f}$ and the computational complexity, we do not consider the note duration. In Figure 1 we present an example of how these matrices are formed. Note that the matrices $\mathbf{X_f}$ associated with P-F and N-F are binary matrices containing 1 if the finger j plays at note onset i, while the matrices $\mathbf{X_f}$ associated with P-V and N-P contain either the velocities or the transition probabilities associated with each note onset. As a baseline, we include a simple note onset representation K as a feature agnostic to piano technique. We form the matrix $\mathbf{X_n}(i, k)$ containing 1 if a note k is played at the onset i or 0 otherwise, where $k = 1 \dots 88$ are the notes that can be played on the piano.

4.2. Classification methods

We use the five features introduced in Section 4.1 as input for two machine learning methods, gradient-boosted trees and a GRU neural network with an attention mechanism. To that extent, we decided to use machine learning methods that offer some form of interpretability. This property may potentially help with understanding the difficult parts of a score. Correspondingly, the decision trees give us comprehensive explanations while the attention mechanism in the neural network points out which parts the network focuses on.

Gradient-boosted trees – *XGBoost*. We perform a two-step classification of a score into the three difficulty classes using a decision tree classifier, gradient boosted trees [21]. Since this method works solely on fixed-sized inputs, we split the input feature matrices $\mathbf{X_f}$ and $\mathbf{X_n}$ into windows of size w, representing short-term score segments. The resulting matrices $\hat{\mathbf{X}}_\mathbf{f}$ of size $(w \times 10)$ and $\hat{\mathbf{X}}_\mathbf{n}$ of size $(w \times 88)$ are used as input to the classifier. Next, the predictions corresponding to each segment are averaged across the whole score to obtain a global difficulty. Note that besides being interpretable, this method is more robust to noisy labels [21]. In our case, we deal with a weakly supervised scenario because we have annotations solely for the whole piece, and not for each segment. Moreover, the difficulty of the segments may vary across a piece.

GRU network with attention mechanism – *DeepGRU*. Towards modelling time dependencies in the score and dealing with variable size scores as inputs, we use a recurrent neural network classifier. We modify an existing architecture used in multivariate time series classification, *DeepGRU* [10]. The model lies in a set of stacked gated recurrent units (GRU), two fully connected (FC) layers and a global attention model. The final two FC layers, using ReLU activations, take the output of the attention module and use a softmax classifier to create the probability distribution of the class labels. The GRUs stacked layers are able to model the time dependencies while the attention mechanism selectively attends to specific note onsets which are more important in the difficulty decision. We use the attention layer to visualise and understand the important notes.

5. EXPERIMENTS

We evaluate the classification (balanced) accuracy of the machine learning models *XGBoost* and *DeepGRU* trained with proposed features P-F, P-V, N-F, N-P, K on the Mikrokosmos-difficulty dataset we introduce in Section 3. Towards assessing the impact of weak labels, we compare the 2-step classification, *XGBoost (avg)* with a single step classification, where the predictions are not averaged across all segments, *XGBoost (window)*. In addition, we rank the scores using the output probability of the classifiers. The ranking is calculated by multiplying each class probability by their class number in the last layer. We then compare this ranking with the original one proposed by Bartók and the one by the publisher Henle using Spearman's rank correlation.

5.1. Experimental setup

Because we want to keep the window size consistent with the 9 notes used as a temporal context by Pianoplayer, we use segments of window size $w = 9$. Hence, the input matrices are of size 9×10 in the case of $\hat{\mathbf{X}}_\mathbf{f}$ and 9×88 in the case of $\hat{\mathbf{X}}_\mathbf{n}$. Similarly, we set the stride $s = 1$ in order to generate the maximum overlap between the windows. Note that we perform an ablation study to evaluate the effect of the window size.

We use the gradient-boosted trees implementation in the XGBoost library [9] for training the *XGBoost (window)*. We use a random search over the training set (5-fold cross-validation) to tune seven hyper-parameters. In this case, we pick the parameters corresponding to the model with the best balanced accuracy on the validation set. The DeepGRU model is trained with the original parameters [10]: 20 epochs, an Adam optimizer with a learning rate of 0.002, a batch-size of 64 samples and using negative log likelihood loss as the criterion.

In both classification methods, the 80% of data is used for training and 20% for testing. We repeat the experiments for 50 different initial random seeds which control the initialisation of the machine learning models and the train and test splits. We report means and standard deviations for the considered metrics across the 50 seeds.

5.2. Results

The effect of different feature representations on the difficulty classification accuracy is presented in Table 2. The results have to be interpreted considering the large standard deviations across the 50 seeds, which the small size of our dataset may cause.

Table 2. 3-class classification accuracy (%) for different features and machine learning models.

	XGBoost (window)		*XGBoost (avg)*		*DeepGRU*	
	train	test	train	test	train	test
K	77 ± 9	51 ± 7	90 ± 8	65 ± 10	93 ± 6	64 ± 8
N − F	71 ± 5	51 ± 5	83 ± 5	62 ± 8	85 ± 5	72 ± 8
P − F	74 ± 7	52 ± 4	86 ± 6	64 ± 8	87 ± 4	73 ± 6
N − P	80 ± 6	57 ± 5	92 ± 6	67 ± 8	83 ± 5	71 ± 8
P − V	88 ± 5	62 ± 6	96 ± 4	68 ± 10	80 ± 5	$\mathbf{78 \pm 6}$

We observe that *XGBoost (avg)* yields better results than *XGBoost (window)*. Thus, averaging the results across the whole piece is better than considering stand-alone segments. We see that using transition probabilities, N-P, and velocity information, P-V, increases the performance compared to using the fingers solely, P-F, N-F. Comparing the two finger modelling methods, we see that Pianoplayer fingerings, P-, perform better than the Nakamura fingerings, N-. To our best knowledge, it is the first time Pianoplayer is

explored in a research paper, and the results demonstrate its performance is comparable with the established Nakamura's algorithm.

The importance of finger modelling in piano difficulty analysis may be seen in the drastic improvement for the *DeepGRU* case, where all the fingering strategies improved the classification accuracy by a large margin compared to features generated directly from sheet music without finger modelling (K).

The ranking performance using Bartók and Henle difficulty rankings is presented in Table 3. Even though the models are trained using solely 3-class labels, the difficulty rankings we derive evaluated against the rankings provided by the composer (Bartók) and the publisher (Henle) achieve high correlation.

Table 3. Spearman's rank correlation coefficient between the rankings derived from the output probabilities of the classifiers and the Bartók and Henle rankings

	XGBoost (avg)		*DeepGRU*	
	Bartók	Henle	Bartók	Henle
K	$.74 \pm .09$	$.75 \pm .08$	$.71 \pm .08$	$.71 \pm .09$
N − F	$.70 \pm .08$	$.66 \pm .08$	$.82 \pm .06$	$.81 \pm .06$
P − F	$.69 \pm .09$	$.64 \pm .11$	$.80 \pm .06$	$.80 \pm .06$
N − P	$.78 \pm .07$	$.73 \pm .09$	$.84 \pm .06$	$.83 \pm .06$
P − V	$.81 \pm .08$	$.78 \pm .10$	$\mathbf{.86 \pm .05}$	$\mathbf{.86 \pm .05}$

The ablation study of the window size for the *XGBoost* method is shown in Table 4. Considering the high standard deviation, we concluded that the change in window size is not statistically significant. For this reason, we decided to stick with the window size of the Pianoplayer method, which is $w = 9$.

Table 4. Evaluation considering different window size for *XGBoost (avg)*. Results are presented in terms of 3-class classification accuracy (%), Spearman rank correlation coefficient for Bartók and Henle.

window size	3-class acc	Bartók rank	Henle rank
$w = 1$	69 ± 12	$.81 \pm .08$	$.77 \pm .10$
$w = 3$	69 ± 10	$.81 \pm .08$	$.78 \pm .10$
$w = 5$	69 ± 10	$.81 \pm .09$	$.78 \pm .10$
$w = 9$	68 ± 10	$.81 \pm .08$	$.78 \pm .10$
$w = 13$	69 ± 10	$\mathbf{.83 \pm .07}$	$\mathbf{.82 \pm .07}$
$w = 19$	$\mathbf{70 \pm 9}$	$.82 \pm .06$	$.80 \pm .07$

5.3. Local Difficulty Feedback

As a practical application to music learning, giving feedback related to the local difficulty of a piece allows students and teachers to focus and improve on the most difficult passages. Because *XGBoost* and *DeepGRU* are interpretable, we derive local difficulty feedback for both of the methods. We created an online tool to explore the two different types of feedback for the Mikrokosmos-difficulty dataset [3].

Window-based feedback. The difficulty of the windows pertaining to a score may vary considerably. In the case of the 2-step *XGBoost (window)* classification, we obtain an output probability for each window. Considering the maximum overlapping for the windows for a stride of $s = 1$, we derive a probability for each note onset. Therefore, this feedback is an indicator of the local difficulty corresponding to that note onset. An example of visualisation of the local difficulty may be seen in Figure 2. The difficulty levels in the Mikrokosmos-difficulty dataset are displayed over the score with three different colours We colour level 1 notes with green, level 2 notes with yellow and level 3 with red. In the excerpt shown in Figure 2 we note how the difficulty changes within the same piece. We observe that the part we highlight as A.1 has asymmetric changes between the two hands while the A.2 part is a simple arpeggio. Consequently, A.1 is marked in red while A.2 is yellow.

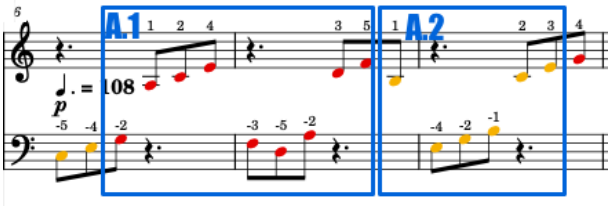

Fig. 2. Mikrokosmos Sz. 107 no. 142. Béla Bartók excerpt. Local difficulty feedback from *XGBoost (window)* with P-V as input feature. The score is from the test set of a random seed.

Attention-based feedback. This method grounds on the weights of the attention layer in the *DeepGRU* model. Attention tells us the notes on which the model is focusing. The most interesting aspect of the present feedback is dealing with repetitions that occur in the short term. We can see a particular example in Figure 3.In this plot the attention weights corresponding to each note control the intensity of the color. The pairs A.1-A.2 and B.1-B.2 are performed with the same physical movement. Therefore, in the second pattern, A.2 and B.2 ,respectively, the method pays less attention. The attention reasoning is similar to how music students should study. When dealing with two similar patterns, students should study the first one very well and repeat the second one by imitating the first one.

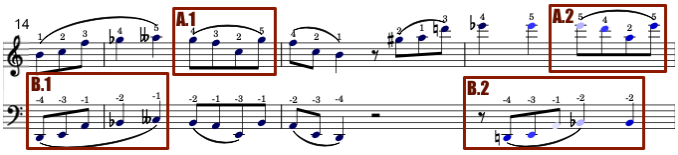

Fig. 3. Mikrokosmos Sz. 107 no. 109. Béla Bartók excerpt. Local difficulty feedback from *DeepGRU* approach with P-V as input feature. The score is from the test set of a random seed.

6. CONCLUSIONS AND FUTURE WORK

In this work, we proposed a methodology to classify piano scores based on piano technique features. Namely, we derive piano fingering features from two approaches [12, 20], and a simple baseline based on using solely the notes in the score. Towards training and testing machine learning methods for this task, we distributed the Mikrokosmos-difficulty dataset. We evaluated two interpretable machine learning methods on the proposed dataset to show that piano technique features are good predictors for the difficulty of performing a piece. In addition to the openly available source code, models and data, we also provide an online demo which visualizes the segmental difficulty, which is crucial in educational feedback. We acknowledge that the small size of the dataset may lead to overfitting biases and large standard deviations for the bootstrapped experiments. In future, we plan to extend this work with larger score dataset with difficulty annotations and other instruments.

[3] At: `https://musiccritic.upf.edu/mikrokosmos`

7. REFERENCES

[1] Deepanway Ghosal and Maheshkumar H Kolekar, "Music genre recognition using deep neural networks and transfer learning.," in *Interspeech*, 2018, pp. 2087–2091.

[2] Christof Weiss and Meinard Müller, "Tonal complexity features for style classification of classical music," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 688–692.

[3] Satoru Fukayama and Masataka Goto, "Music emotion recognition with adaptive aggregation of gaussian process regressors," in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2016, pp. 71–75.

[4] Yi-Hsuan Yang and Homer H Chen, "Ranking-based emotion recognition for music organization and retrieval," *IEEE Transactions on audio, speech, and language processing*, vol. 19, no. 4, pp. 762–774, 2010.

[5] Shih-Chuan Chiu and Min-Syan Chen, "A study on difficulty level recognition of piano sheet music," in *2012 IEEE International Symposium on Multimedia*. IEEE, 2012, pp. 17–23.

[6] Pedro Ramoneda, "Computational methods to study piano music in education context," Master's Thesis, Universitat Pompeu Fabra, 2020.

[7] Heinrich Neuhaus, *The art of piano playing*, Kahn and Averill, 2008.

[8] Véronique Sébastien, Henri Ralambondrainy, Olivier Sébastien, and Noël Conruyt, "Score analyzer: Automatically determining scores difficulty level for instrumental e-learning," in *Proceedings of 13th International Society for Music Information Retrieval Conference*, Porto, 2012, ISMIR 2021.

[9] Tianqi Chen and Carlos Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[10] Mehran Maghoumi and Joseph J LaViola, "Deepgru: Deep gesture recognition utility," in *International Symposium on Visual Computing*. Springer, 2019, pp. 16–31.

[11] Eita Nakamura, Nobutaka Ono, and Shigeki Sagayama, "Merged-output hmm for piano fingering of both hands.," in *Proceedings of the 13th International Society for Music Information Retrieval Conference*, Taipei, 2014, ISMIR 2014, pp. 531–536.

[12] Eita Nakamura, Yasuyuki Saito, and Kazuyoshi Yoshii, "Statistical learning and estimation of piano fingering," *Information Sciences*, vol. 517, pp. 68–85, 2020.

[13] Eita Nakamura, Emmanouil Benetos, Kazuyoshi Yoshii, and Simon Dixon, "Towards complete polyphonic music transcription: Integrating multi-pitch detection and rhythm quantization," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 101–105.

[14] Eita Nakamura, Kazuyoshi Yoshii, and Shigeki Sagayama, "Rhythm transcription of polyphonic piano music based on merged-output hmm for multiple voices," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 4, pp. 794–806, 2017.

[15] Eita Nakamura and Shigeki Sagayama, "Automatic piano reduction from ensemble scores based on merged-output hidden markov model," in *ICMC*, 2015.

[16] Eita Nakamura and Kazuyoshi Yoshii, "Statistical piano reduction controlling performance difficulty," *APSIPA Transactions on Signal and Information Processing*, vol. 7, 2018.

[17] Neratron, Sibelius Software division, Avid Technology, "Photoscore," v2020.

[18] Richard Parncutt, John A Sloboda, Eric F Clarke, Matti Raekallio, and Peter Desain, "An Ergonomie Model of Keyboard Fingering for Melodic Fragments Sibelius Academy of Music , Helsinski," *Music perception: An Interdisciplinary Journal*, vol. 14, no. 4, pp. 341–382, 1997.

[19] Matteo Balliauw, Dorien Herremans, Daniel Palhazi Cuervo, and Kenneth Sörensen, "A variable neighborhood search algorithm to generate piano fingerings for polyphonic sheet music," *International Transactions in Operational Research*, vol. 24, no. 3, pp. 509–535, 2017.

[20] "PianoPlayer automatic piano fingering generator.," `https://github.com/marcomusy/pianoplayer`, Accessed: 2021-02-18.

[21] Anabel Gómez-Ríos, Julián Luengo, and Francisco Herrera, "A study on the noise label influence in boosting algorithms: Adaboost, gbm and xgboost," in *International Conference on Hybrid Artificial Intelligence Systems*. Springer, 2017, pp. 268–280.