MUSIC TRANSLATION: GENERATING PIANO ARRANGEMENTS IN DIFFERENT PLAYING LEVELS

Matan Gover
Simply
matan@hellosimply.com

Oded Zewi
Simply
oded@hellosimply.com

ABSTRACT

We present a novel task of "playing level conversion": generating a music arrangement in a target difficulty level, given another arrangement of the same musical piece in a different level. For this task, we create a parallel dataset of piano arrangements in two strictly well-defined playing levels, annotated at individual phrase resolution, taken from the song catalog of a piano learning app. In a series of experiments, we train models that successfully modify the playing level while preserving the musical 'essence'. We further show, via an ablation study, the contributions of specific data representation and augmentation techniques to the model's performance.

In order to evaluate the performance of our models, we conduct a human evaluation study with expert musicians. The evaluation shows that our best model creates arrangements that are almost as good as ground truth examples. Additionally, we propose MuTE, an automated evaluation metric for music translation tasks, and show that it correlates with human ratings. Demos are available online. [1]

1. INTRODUCTION

In this paper we tackle the task of generating piano arrangements for specific playing difficulty levels, conditioned on piano arrangements of the same music in a different playing level. The purpose driving this work is to significantly accelerate our rate of content creation for our piano learning app, Simply Piano. [2] In Simply Piano, we have a library of songs for beginner piano learners to practice. Our library contains arrangements in various playing levels, to match the skills acquired by our learners over their piano journey. Arrangements are prepared by expert musicians based on a pre-defined set of piano pedagogy guidelines. These guidelines are strict and are designed to maintain a uniform playing method and skill level in order to help learners familiarize themselves with the piano in a systematic way. Our aim is to be able to automatically generate multiple arrangements, spanning our range of playing levels, from a single human-generated arrangement at a given level.

Our contributions are two-fold. First, we introduce the task of playing level conversion. We share our dataset preparation method and discuss various experiments regarding choice of music representation and data augmentation methods. Second, we develop and share an automated evaluation metric that can be used for music translation or generation tasks where a reference is available. This automated evaluation metric provides a fast, low-effort estimation of human ratings.

2. RELATED WORK

The symbolic music generation field has advanced considerably in recent years. Specifically, the community has been fast in adopting and adapting capable sequence-to-sequence models, such as Transformer [1], which were originally developed for natural language processing tasks such as machine translation.

Some works focus on unconditional music generation, that is, generating music from scratch [2–4]. Many recent ones are based on Transformer and its variants [2, 4–10]. Other works tackle music generation conditioned on specific inputs: emotions [11], structure [12], theme [13], or musical attributes such as note density [14]. In [2, 15], an accompaniment is generated conditioned on a melody or chord progression.

While most papers work in the performance domain (training on MIDI performances), some work in the score domain as we do. Some use custom tokenized score representations [16, 17] while others operate on text-based notation formats like ABC [18, 19].

Works that are most closely related to this paper are those concerning symbolic music style transfer: generating a musical piece in a target style given the same piece in another style. Musical 'style' can refer to various attributes [20]: symbolic abstractions (score), expressive timing and dynamics (performance), and acoustic details (sound). Of relevance to us is only the first of the three.

Due to the lack of datasets with parallel examples in different domains, most works use unsupervised methods for learning style features. In [9], a Transformer with an encoder bottleneck is used to learn a global style representation. This style representation is then combined with melody representations and fed into a decoder to generate the same melody in a different style. [21] also learns a style

[1] https://www.matangover.com/music-translation/

[2] https://www.hellosimply.com/simply-piano

representation in an unsupervised manner using an autoencoder bottleneck together with a style classifier. [22] uses inductive biases in the encoder to disentangle chord and texture factors. In [23, 24], CycleGAN is used to transfer music between genres. Difficulty level classifiers such as [25–27] could also be used for generating weakly-labeled non-parallel training data for level translation.

In [28], style translation is performed with supervised learning using synthetic parallel data. To our knowledge, our work is the first to perform supervised domain transfer for symbolic music with real-world parallel data.

3. DATA PREPARATION AND REPRESENTATION

Our proprietary dataset of piano arrangements is taken from the song library of Simply Piano. For each song in the library, expert musicians have created arrangements in up to three levels: Essentials (easy), Intermediate, and Pre-Advanced (more difficult but still aimed at learners). Strict arrangement guidelines have been developed by our musicians to create an approachable and engaging learning path for users. We use a pedagogy system based on hand positions, where the aim is to initially minimize the player's need to shift their hands, and then gradually introduce new hand positions and musical concepts as users progress.

Figure 1 illustrates the different difficulty levels. In this paper, we focus on two levels only: Essentials and Intermediate. Specifically, we tackle the task of translating from Intermediate to Essentials. The main difference between the two levels are in hand positions, rhythmic complexity, and harmonic complexity (amount of simultaneous notes). In Essentials, we only allow a small number of positions and keep position shifts to a minimum. We keep the number of chords small and emphasize the melody. We also limit the range of allowed pitches and rhythms: in Essentials we generally do not use tied notes or sixteenth notes. In both Essentials and Intermediate, we do not use tuplets and multiple indepdendent voices on the same staff.

When approaching the task of song level translation, initial experiments showed that translating entire songs at once is a difficult task: models we trained did not learn a meaningful mapping between levels. The reason is probably that song structure can vary greatly between levels in our dataset. That is, some levels of the same song omit certain phrases, while other levels include extra phrases, or change the phrase order. It seems that for full-song mapping, a larger (or cleaner) dataset is needed.

For this reason, we focus our work on translating individual musical phrases. Fortunately, we could make use of existing annotations for this purpose. Each of our arrangements is divided by musicians into phrases, based on the song structure. For example, a song could have the following phrases: Intro, Verse 1, Verse 2, Chorus, Verse 1, Ending. Phrase names stay consistent between different levels of the same song, allowing for minor variations such as 'Phrase 1' ⇔ 'Phrase 1a'.

We use the phrase boundary annotations to derive a dataset of parallel phrases from our library of arrangements. This is illustrated in Figure 2. We start from two

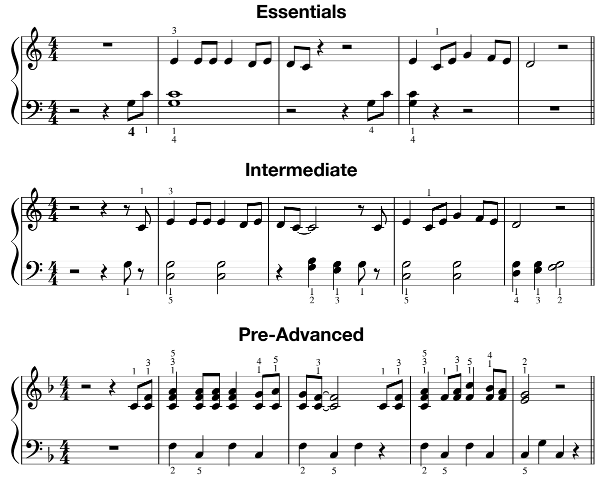

Figure 1. Comparison of three difficulty levels. From bottom to top: *Pre-Advanced* is fully harmonized with rhythmic bass in the left hand and chords in the right hand. *Intermediate* is transposed to the easier key of C major and contains a simplified accompaniment. The right hand chords are omitted, and the melody is split differently between the hands to accomodate for less hand shifts. In *Essentials*, the accompaniment is reduced to a minimum, and a tied note in the melody is removed to simplify the rhythm.

parallel arrangements of the same song in the source and target levels. We discard arrangements where the target and source have different time signatures. We then transpose the source arrangement to the same key as the target arrangement (see details below).

Following that, we match phrases from the source and target levels using heuristics that take into account phrase order, names, and durations. These heuristics were crucial for obtaining a dataset of sufficient size. To account for added or removed phrases, we compute a diff between the source and target phrase names. Phrases with exact name (and order) matches are then considered to be parallel if their duration difference is 2 measures or less. For phrases with no exact name matches, we diff the phrase durations instead, and consider phrases as parallel if their durations match and their names are sufficiently similar.

In some songs, different levels were written in different keys (e.g., Essentials in C major and Intermediate in D major), to fit the arrangement to the desired playing level. For our task, it was important that source and target phrases maintain the same key, so that the model could learn a consistent mapping. Since we didn't have key annotations for the dataset, we implemented a heuristic method for transposition estimation.

We initially tried existing methods for key estimation [29] for each of the phrases, however these weren't accurate enough. Since we only need to estimate the transposition (and not the actual key), we came up with a dedicated heuristic: convert each of the arrangements (source s and target t) into a "pitch-class piano-roll", a boolean matrix (P_s and P_t) with time and pitch-class dimensions, in which 1 signifies the given pitch-class is active at the given time.

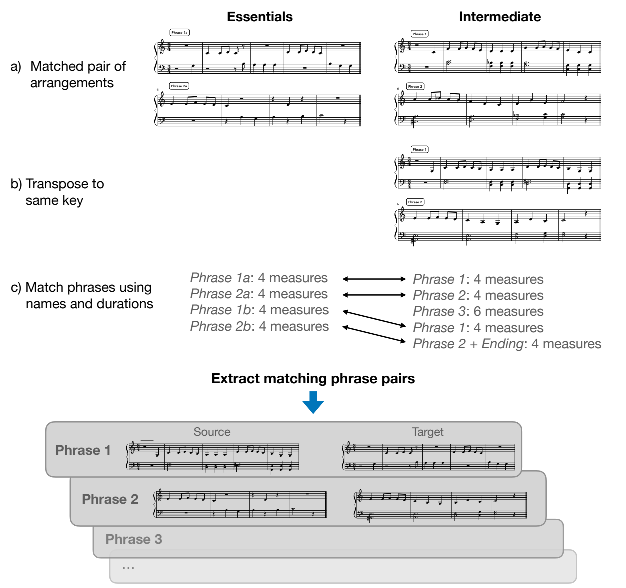

Figure 2. Data preparation: we start with two arrangements of the same song in different levels, transpose them to the same key, and extract matching phrase pairs.

We then compute the pitch-class overlap ($\sum P_s \cap P_t$) between the piano-rolls for each possible transposition (out of 12 possibilities) and take the one that gives the maximum overlap. We use this value to transpose the source level's phrases to match the target level's key.

While the transposition might slightly affect the source phrases' difficulty, the target's difficulty is kept intact. We found that including these examples improves the generated results, since it allows us to increase the dataset size considerably.

3.1 Music Representations

Sequence-to-sequence models such as Transformer operate on a stream of tokens. In order to use these models we must represent music as a sequence, even if it is polyphonic or consists of multiple tracks. Since symbolic music is multi-dimensional in nature, various ways to turn music into a token sequence have been proposed. MidiTok [30] gives a good summary of various representations.

It is crucial to choose a representation that fits the given task. We experimented with three representations for piano music: MIDI-like, Notes, and Notes+Hands. The MIDI-like representation uses note on and note off tokens, along with time shift tokens to signify the passing of time [2,31]. MIDI-like representations are commonly used, perhaps because it is easy to derive them directly from MIDI files. For our use case, the MIDI-like representation has two limitations: it forces the model to meticulously track active notes in order to later output corresponding note off tokens (potentially leading to syntax errors), and it does not encode the metrical structure of music, potentially leading to compound alignment errors if a single time-shift is wrong.

To counter these problems we employ the Notes representation, which uses three tokens for each note: offset, pitch, and duration. The offset token signifies the note's time offset from the beginning of the measure. A 'bar' token is output at the beginning of each measure. This is similar to REMI [6] (but without velocity tokens), in that it (partially) encodes the metrical structure of music.

Since our output is sheet music, we must output two separate staves for the right and left hands. Since the MIDI-like and Notes representations do not encode track information, we use a heuristic by which all notes on or above middle C are assigned to the right hand, and any note lower than middle C is given to the left hand. Chords (notes with identical onset and offset times) are always grouped to one hand. This heuristic generally matches our dataset's specific characteristics, but is quite coarse and leads to hand assignment errors.

To solve this, we test an additional representation, Notes+Hands, which is identical to the Notes representation but adds a 'hand' token to each note. As stated in [15], such a representation emphasizes the harmonic ('vertical') aspect of music. In the case of piano music, sorting notes by time (and only then by track) emphasizes the overall coherence of the two piano hands over the melodic coherence of each hand separately.

3.2 Data Augmentation

Since our dataset is small, even small models quickly overfit it, limiting our ability to scale model size, consequently limiting the amount of information the model can learn. Previous work has shown that data augmentation can turn an overfitting problem into an underfitting problem, allowing us to iteratively increase model size [32]. We followed this protocol by gradually adding data augmentation methods and increasing model size to improve the final results.

To match our translation task, we needed augmentations that meaningfully alter both source and target phrases, without corrupting the relationship between them. We implemented the following augmentation methods: adding empty measures at random locations; randomly cutting the beginning or end of phrases; removing some measures randomly; and rhythm augmentation (doubling the duration of each note) in some measures.

We purposefully do not include transposition in the list of augmentations, even though it is commonly used in other works. As described above, the arrangement style in our dataset is not transposition-invariant: many features depend on absolute pitch such as hand positions, fingering, key signatures, range limits and hand allocation.

We release our data augmentation code, [3] built on top of the muspy library [33]. We believe that these augmentation methods could be useful for other symbolic music generation tasks, especially when working with small datasets.

4. MODEL

We use a classic Transformer model [1], specifically the BART [34] encoder-decoder implementation from the hug-

[3] https://github.com/matangover/muspy-augment

gingface `transformers` library [35]. The specifics of this implementation (compared to classic Transformer) are that it uses learned position embeddings (we saw slightly better results with these compared to sinusoidal embeddings) and GeLU rather than ReLU as the activation function (this did not seem to make a difference in our experiments). As in the original Transformer, we use a shared weight matrix for the encoder, decoder, and output embedding layers.

We experimented with various model configurations: model dimension, number of layers, number of attention heads. We found that larger models ($d_{\text{model}} > 64$) quickly overfit our training dataset and do not achieve good performance on the validation dataset. The optimal model dimension was found to be 32–64 (with the feed-forward layer dimension always set to $4 \cdot d_{\text{model}}$ as in the original Transformer). The optimal number of layers was 3 to 5, depending on the amount of data augmentation. As discussed in Section 3.2, use of more data augmentation enables us to increase model size without overfitting the data.

We trained using the Adam optimizer [36] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, a learning rate of 0.003 and 1,000 warmup steps.

Our dataset contains a total of 5,543 phrase pairs taken from 1,191 songs. We split the data into train (5,241 phrases), validation (244 phrases), and test (58 phrases) splits. We split phrases by song (never including phrases from the same song in different splits) to avoid data leakage between splits. The validation set was used to stop model training after validation loss stopped decreasing. The test set was used for final evaluation (see Section 7).

During prediction, the model outputs a probability distribution for the next output token given all the input tokens and the previous output tokens. However, if at each decoding step we simply pick the most likely output (greedy decoding), we might end up with a non-optimal sequence [37]. We experimented with two decoding methods: beam search and sampling. We found that sampling produces more diverse results (due to its random nature) but beam search produces overall superior results for our task.

5. INTERFACE FOR INTERACTIVE USE

While our model can be used unattended to create arrangements in the target level, in practice we found that it is beneficial to keep musicians “in the loop” by allowing them to use the model interactively rather than in a ‘fire and forget’ fashion. We created an interface in which a musician can load a source arrangement, translate it phrase-by-phrase to the target level, and review the results side by side.

Furthermore, the auto-regressive nature of the model enables interesting use cases: a musician could manually modify some notes and ask the model to re-generate the subsequent music accordingly. Additionally, if we use random sampling decoding (see Section 4) we could generate multiple alternatives for each measure using different random seeds and offer the musician a choice. We could also offer knobs that control sampling parameters such as temperature (for controlling the output diversity vs. quality trade-off) or top-k/top-p thresholds (ways of eliminating unlikely outputs to increase the probability of choosing more likely predictions).

In this way, our model becomes part of the creative process, rather than replacing it. It becomes a tool that assists musicians in their job.

6. EVALUATION METRIC

Evaluation is often difficult for music generation due to the absence of pre-defined criteria and because output quality is subjective [38]. Standard practices are reporting perplexity (the likelihood of the ground truth test data given the trained model) and conducting human evaluation studies [2,4,9,15,31]. Some works also calculate scores based on distributions of certain musical features, either comparing generated data to ground truth distributions [9,15,39] or using self-similarity in the generated data itself [3].

In machine translation (of text), evaluation metrics such as the popular BLEU metric [40] have been developed to measure the correspondence between generated translations and ground truth references. BLEU has been shown to correlate with human ratings. We are not aware of a similar metric for music generation tasks. To this end, we propose MuTE (Music Translation Evaluation), an automated evaluation metric for symbolic music translation or generation tasks where a reference is available.

MuTE is a score between 0 and 1. Like BLEU and similar metrics, it is designed to reflect the correspondence between a machine-generated example and a human-generated reference. BLEU measures word n-gram precision (the portion of correct predictions out of all predictions) and deliberately omits recall because of the desire to allow for multiple reference translations of the same phrase. In our case, we only have a single reference for each phrase, hence we can easily use both precision and recall, and combine them using the harmonic mean to give the commonly used F1 score [41].

MuTE works by treating the model as a multi-label classifier that predicts at every time step what pitches are active. To compute the MuTE score, we convert the reference and target music to piano-rolls (where the time unit is set to a certain small metrical unit – in our case, a sixteenth note), and calculate the F1 score over pitches. We do not compute a global score for the entire piece, since that would mean disregarding note order and timing. Instead, we treat each time-slice of the piano-rolls as an individual sample, compute an F1 score for each time-slice, and average those scores. For time-slices where both the reference and target are silent (and hence precision and recall calculation would result in division by zero), we set the score to 1.

We also need to account for cases in which the target differs from the reference in its duration – if, for example, the model ‘skips’ some measures of the input. We design a procedure that is intended to match the way a human would judge such cases, by detecting skipped or added measures and adjusting the comparison accordingly.

For this, we precede the scoring step with a measure-level alignment procedure. We perform the alignment using dynamic time warping [42]. The feature vectors for the alignment are each measure's pitch-class piano-roll (see Section 3). We found that using pitch-class piano-rolls gives a more robust alignment compared to regular piano-rolls. The distance between each two feature vectors is computed using the Hamming distance (the proportion of elements that disagree between the two vectors). We use the computed alignment path to align the reference and target's (regular) piano-rolls. The aligned piano-rolls are used to compute the F1 score, while masking out the misaligned segments to assign them a score of 0, thus penalizing the model for any alignment errors.

One additional element of MuTE that pertains to our use case is the desire to reflect the separation between the two hands: we would like to penalize wrong hand allocation of notes. For this reason, we use 'track-specific' MuTE: we calculate a separate MuTE score for each hand and average them to get the final score. We found the mean-of-hands metric to correlate better with human ratings than the vanilla MuTE score calculated over both tracks combined.

We note that the measure-based alignment method is specifically suitable for our use case because we use a measure-based representation for our models, and we noticed that models sometimes omit or repeat some measures. For other use cases, the alignment could be computed over individual time steps or beats, or skipped altogether.

Figure 3 illustrates the calculation of the MuTE score. In Section 7 below, we show that the MuTE score correlates with human ratings. While simple and effective, the MuTE score has caveats. First, it does not differentiate between sustained and repeated notes. For example, a half note is considered identical to two consecutive quarter notes of the same pitch. Second, it does not allow for minor variations that might be acceptable to a human rater, such as octave changes and rhythmic variations. Third, the hand-specific version does not account for notes that are missing in one hand but present in the other hand. We plan to address these shortcomings in a future version.

We release a Python library for computing the MuTE score [4] and we welcome researchers to adopt it or adapt it for their use cases as needed.

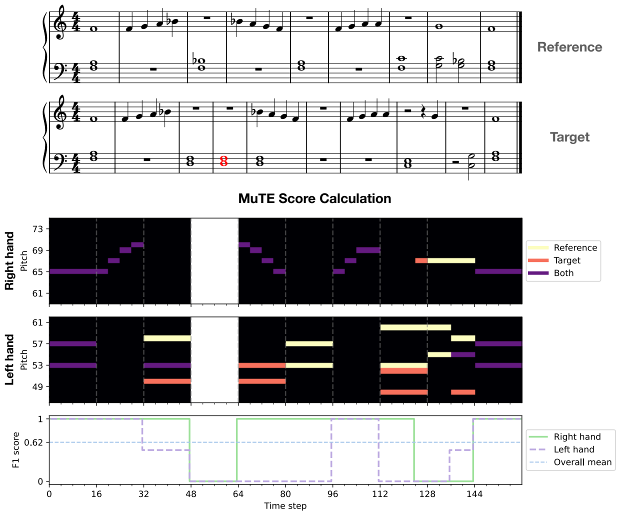

Figure 3. Illustration of the MuTE score calculation. The target score has an added measure (marked in red) with regard to the reference. The misaligned region is masked (in white) on the piano-rolls. The per-hand sample-wise pitch F1 score is calculated from the aligned piano-rolls, and scores are averaged to get the final MuTE score.

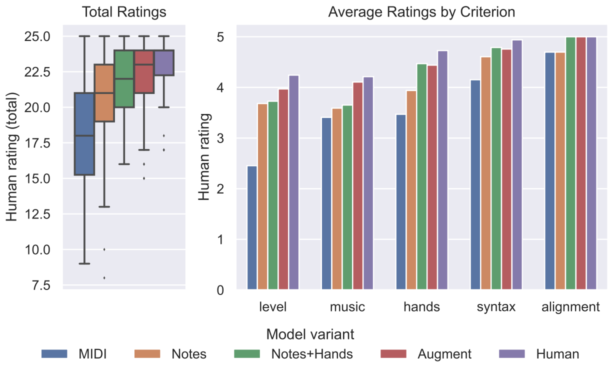

Figure 4. Scores for each model version, as rated by human experts. Our best model (Augment) achieves ratings almost as high as human generated examples (Human).

7. RESULTS

We report the results of our experiments on converting Intermediate level arrangements to Essentials, and compare our models' outputs to matching reference arrangements created by expert musicians. For evaluation we conduct a human evaluation study, calculate MuTE scores (see Section 6) and report the correlation between human ratings and MuTE scores. Furthermore, we run several ablation studies to study the effect of our data augmentation techniques and the choice of music representation.

7.1 Human Evaluation Study

For this study we selected 11 songs from the test set (none of these songs' phrases appeared at training). From each song we extracted a single phrase and translated it from Intermediate level to Essentials, using 4 variants of the model that differ in representation and augmentation. We also extracted the matching phrases from the ground truth Essentials arrangement for comparison.

The first 3 model variants differ only in the music representation they use: MIDI-like, Notes, and Notes+Hands (see details in Section 3.1). The fourth model variant uses the Notes+Hands representation and adds data augmentation (see Section 3.2) with randomly varying parameters to about half of the training examples.

This resulted in 55 examples, which were then rated by 6 expert musicians who are familiar with the Essentials level guidelines. The examples were randomly ordered and all identifiers of model version and ground truth were removed. Musicians were told all 5 examples are different model versions and were not aware of the details of the ver-

[4] https://github.com/matangover/mute

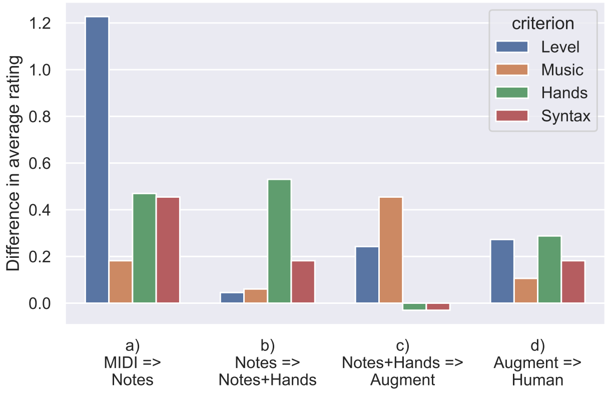

Figure 5. Difference in average rating, per criterion, between model variants. a) Moving from MIDI to Notes improved across all criteria, mostly Level. b) Adding hand information improved hand assignment. c) Augmentation improved preserving musical content. d) No particular criterion stands out in the difference from our best model to the human-generated ground truth.

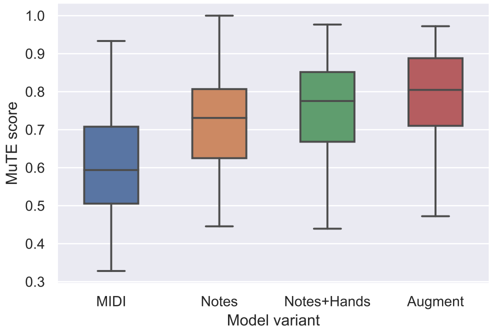

Figure 6. Evaluation of our model variants on the test set using the MuTE score.

sions or of the presence of human-generated ground truth. Ratings between 1–5 were collected in 5 criteria:

1. Meeting Essentials level guidelines (Level)
2. Preserving musical content (Music)
3. Correctly assigning hands according to allowed hand positions (Hands)
4. Avoiding syntactic style errors such as crossover voices (Syntax)
5. Maintaining structure: avoiding missing or extra bars (Alignment)

The maximum possible total score is 25. Figure 4 shows the distributions of scores for each model variant and for the ground truth. These results show that the changes between model variants improved overall output quality. The best model (Augment) achieves ratings almost as high as the ground truth (Human).

In Figure 5 we show the gains in each criterion from the changes in each model variant. Some differences are easily explainable, for example, adding hand information improved the correct assignment of hands. Interestingly, the difference between our best model to the ground truth cannot be attributed to any specific criterion. We left out the Alignment criterion from the figure, as ratings under 5 were very rare. The few ratings below 5 were given almost only to the MIDI model variant, which is more prone to alignment errors due to the lack of *bar* tokens.

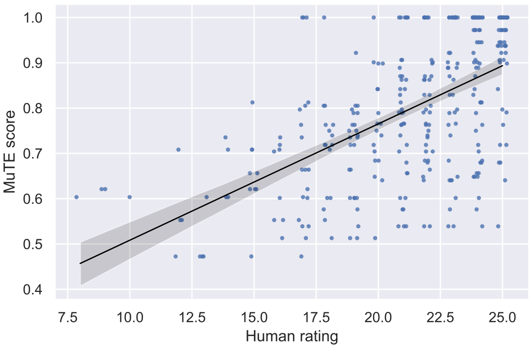

Figure 7. A scatter-plot showing the relationship between human ratings and MuTE scores. The black line shows a linear regression model fit and the shaded area shows the 95% confidence interval for the regression estimate.

7.2 Automated evaluation

We calculate MuTE scores (see Section 6) for the entire test set (58 phrases from 12 songs). The results are shown in Figure 6, and confirm the human evaluation results. Additionally, for the 55 examples rated by human musicians, we compare the MuTE scores to the human ratings (the MuTE score of the ground truth compared to itself is always 1). As shown in Figure 7, we find that MuTE scores are correlated with human ratings with a Pearson correlation coefficient of 0.56 ($p = 4 \cdot 10^{-29}$). While the correlation is not perfect, it shows that MuTE can be used as an estimator for human ratings. Furthermore, MuTE correlates better with the total human rating than with any of the individual criteria's ratings: the correlation coefficients with the indvidiual criteria are 0.27 (Alignment), 0.37 (Hands), 0.47 (Level), 0.37 (Music), and 0.29 (Syntax). The similarity of figures 6 and 4 further shows that MuTE scores can be used to rank models with similar results to human rankings, while being cheaper and faster to compute.

8. CONCLUSION

We presented the task of playing level conversion: converting piano arrangements from one difficulty level to another. We ran several experiments focused on simplifying arrangements to an easier level, and showed the importance of data representation and augmentation. Our best model creates arrangements that achieve human ratings almost as high as reference arrangements composed by expert musicians. We designed the MuTE evaluation metric and showed that it correlates with human ratings. For the benefit of the community, we share our data augmentation and evaluation code.

9. ACKNOWLEDGEMENTS

Thanks to the musicians at Simply for creating the arrangements used to train our models, and for helping perform the evaluation. Thanks to Eran Aharonson for valuable discussions and feedback.

10. REFERENCES

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[2] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. Shazeer, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, “Music Transformer: Generating music with long-term structure,” in *International Conference on Learning Representations*, 2018.

[3] G. Mittal, J. Engel, C. Hawthorne, and I. Simon, “Symbolic music generation with diffusion models,” in *Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[4] J. Liu, Y. Dong, Z. Cheng, X. Zhang, X. Li, F. Yu, and M. Sun, “Symphony generation with permutation invariant language model,” *arXiv preprint arXiv:2205.05448*, 2022.

[5] C. Donahue, H. H. Mao, Y. E. Li, G. W. Cottrell, and J. McAuley, “LakhNES: Improving multi-instrumental music generation with cross-domain pre-training,” in *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

[6] Y.-S. Huang and Y.-H. Yang, “Pop Music Transformer: Beat-based modeling and generation of expressive pop piano compositions,” in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020.

[7] W.-Y. Hsiao, J.-Y. Liu, Y.-C. Yeh, and Y.-H. Yang, “Compound Word Transformer: Learning to compose full-song music over dynamic directed hypergraphs,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[8] A. Muhamed, L. Li, X. Shi, S. Yaddanapudi, W. Chi, D. Jackson, R. Suresh, Z. Lipton, and A. J. Smola, “Transformer-GAN: symbolic music generation using a learned loss,” in *4th Workshop on Machine Learning for Creativity and Design at NeurIPS*, 2020.

[9] K. Choi, C. Hawthorne, I. Simon, M. Dinculescu, and J. Engel, “Encoding musical style with Transformer autoencoders,” in *International Conference on Machine Learning*, 2020, pp. 1899–1908.

[10] C. Payne, “MuseNet,” *OpenAI Blog*, 2019. [Online]. Available: https://openai.com/blog/musenet

[11] S. Sulun, M. E. P. Davies, and P. Viana, “Symbolic music generation conditioned on continuous-valued emotions,” *IEEE Access*, vol. 10, 2022.

[12] K. Chen, W. Zhang, S. Dubnov, G. Xia, and W. Li, “The effect of explicit structure encoding of deep neural networks for symbolic music generation,” in *2019 International Workshop on Multilayer Music Representation and Processing (MMRP)*, 2019.

[13] Y.-J. Shih, S.-L. Wu, F. Zalkow, M. Muller, and Y.-H. Yang, “Theme Transformer: Symbolic music generation with theme-conditioned transformer,” *IEEE Transactions on Multimedia*, 2022.

[14] J. Ens and P. Pasquier, “MMM: Exploring conditional multi-track music generation with the Transformer,” *arXiv preprint arXiv:2008.06048*, 2020.

[15] Y. Ren, J. He, X. Tan, T. Qin, Z. Zhao, and T.-Y. Liu, “PopMAG: Pop music accompaniment generation,” in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020.

[16] M. Suzuki, “Score Transformer: Transcribing quantized MIDI into comprehensive musical score,” in *Late-Breaking Demo Session of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[17] ——, “Piano fingering estimation and completion with Transformers,” in *Late-Breaking Demo Session of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[18] C. Geerlings and A. Meroño-Peñuela, “Interacting with GPT-2 to generate controlled and believable musical sequences in ABC notation,” in *Proceedings of the 1st Workshop on NLP for Music and Audio (NLP4MusA)*, 2020, pp. 49–53.

[19] B. Sturm, J. F. Santos, O. Ben-Tal, and I. Korshunova, “Music transcription modelling and composition using deep learning,” in *1st Conference on Computer Simulation of Musical Creativity*, 2016.

[20] S. Dai, Z. Zhang, and G. G. Xia, “Music style transfer: A position paper,” in *Proceeding of the International Workshop on Musical Metacreation (MUME)*, 2018.

[21] G. Brunner, A. Konrad, Y. Wang, and R. Wattenhofer, “MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer,” in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[22] Z. Wang, D. Wang, Y. Zhang, and G. Xia, “Learning interpretable representation for controllable polyphonic music generation,” in *Proceedings of the 21nd International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[23] G. Brunner, Y. Wang, R. Wattenhofer, and S. Zhao, “Symbolic music genre transfer with CycleGAN,” in *IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*, 2018, pp. 786–793.

[24] G. Brunner, M. Moayeri, O. Richter, R. Wattenhofer, and C. Zhang, "Neural symbolic music genre transfer insights," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2019, pp. 437–445.

[25] S.-C. Chiu and M.-S. Chen, "A study on difficulty level recognition of piano sheet music," in *2012 IEEE International Symposium on Multimedia*, 2012, pp. 17–23.

[26] Y. S. Ghatas, M. B. Fayek, and M. M. Hadhoud, "Generic symbolic music labeling pipeline," *IEEE Access*, vol. 10, pp. 76 233–76 242, 2022.

[27] P. Ramoneda, N. C. Tamer, V. Eremenko, X. Serra, and M. Miron, "Score difficulty analysis for piano performance education based on fingering," in *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 201–205.

[28] O. Cífka, U. Şimşekli, and G. Richard, "Supervised symbolic music style translation using synthetic data," in *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

[29] D. Temperley, "What's key for key? The Krumhansl-Schmuckler key-finding algorithm reconsidered," *Music Perception*, vol. 17, no. 1, pp. 65–100, 1999.

[30] N. Fradet, J.-P. Briot, F. Chhel, A. E. F. Seghrouchni, and N. Gutowski, "MidiTok: A Python package for MIDI file tokenization," in *Late-Breaking Demo Session of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[31] S. Oore, I. Simon, S. Dieleman, D. Eck, and K. Simonyan, "This time with feeling: Learning expressive musical performance," *Neural Computing and Applications*, 2018.

[32] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Interspeech*, 2019.

[33] H.-W. Dong, K. Chen, J. McAuley, and T. Berg-Kirkpatrick, "MusPy: A toolkit for symbolic music generation," in *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[34] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7871–7880.

[35] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020, pp. 38–45.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR)*, 2015.

[37] D. Ippolito, R. Kriz, J. Sedoc, M. Kustikova, and C. Callison-Burch, "Comparison of diverse decoding methods from conditional language models," in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL*, 2019, pp. 3752–3762.

[38] L.-C. Yang and A. Lerch, "On the evaluation of generative models in music," *Neural Computing and Applications*, vol. 32, no. 9, pp. 4773–4784, 2020.

[39] S. Wu and Y. Yang, "The Jazz Transformer on the front line: Exploring the shortcomings of AI-composed music through quantitative measures," in *Proceedings of the 21th International Society for Music Information Retrieval Conference (ISMIR)*, 2020, pp. 142–149.

[40] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: A method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 2002, pp. 311–318.

[41] C. J. Van Rijsbergen, "Foundation of evaluation," *Journal of documentation*, vol. 30, no. 4, pp. 365–373, 1974.

[42] S. Dixon and G. Widmer, "MATCH: A music alignment tool chest," in *Proceedings of the 6th International Society for Music Information Retrieval Conference (ISMIR)*, 2005, pp. 492–497.