ONLINE SYMBOLIC MUSIC ALIGNMENT WITH OFFLINE REINFORCEMENT LEARNING

Silvan David Peter

Institute of Computational Perception, Johannes Kepler University Linz, Austria

`silvan.peter@jku.at`

ABSTRACT

Symbolic Music Alignment is the process of matching performed MIDI notes to corresponding score notes. In this paper, we introduce a reinforcement learning (RL)-based *online* symbolic music alignment technique. The RL agent — an attention-based neural network — iteratively estimates the current score position from local score and performance contexts. For this symbolic alignment task, environment states can be sampled exhaustively and the reward is dense, rendering a formulation as a simplified offline RL problem straightforward. We evaluate the trained agent in three ways. First, in its capacity to identify correct score positions for sampled test contexts; second, as the core technique of a complete algorithm for symbolic online note-wise alignment; and finally, as a real-time symbolic score follower. We further investigate the pitch-based score and performance representations used as the agent's inputs. To this end, we develop a second model, a two-step Dynamic Time Warping (DTW)-based *offline* alignment algorithm leveraging the same input representation. The proposed model outperforms a state-of-the-art reference model of offline symbolic music alignment.

1. INTRODUCTION

Music alignment refers to matching at least two different versions of the same musical material. In this paper, we address symbolic music alignment, for our purposes defined as models that match individual notes of a performance recorded as MIDI file to individual notes of a score encoded as MusicXML file.

Alignment procedures can be separated into online or offline classes. If the alignment procedure is carried out with access to the full versions of the musical material, we refer to it as offline alignment. Conversely, if one version is only known up to the point currently to be matched, we refer to it as online.

We introduce a reinforcement learning (RL)-based *online* symbolic music alignment technique. It aligns symbolically encoded music or, more specifically, MIDI performances to their corresponding MusicXML scores by matching individual notes of each version. The RL agent – a small attention-based neural network – is trained to iteratively predict the current score position from limited score and past performance contexts. The current performance note and estimated score position are then processed to compute a symbolic note-wise matching.

RL terminology introduces another online versus offline differentiation. RL is termed online if the agent learns from data created by the agent's interaction with its environment during training. In our case, we use offline RL, that is, the agent is trained using a dataset of exhaustively sampled environment states and associated rewards, effectively turning agent training into a supervised learning problem. Once trained in an offline fashion, the agent can be used in online alignment.

The agent processes a purely *pitch-based representation* and timing information is only incorporated in a post-processing step. Before addressing the online problem, we investigate the same separation of pitch and time processing in an offline setting: we develop a two-step (first pitch, then time) Dynamic Time Warping (DTW) offline model and evaluate it against the state of the art in note-wise alignment in Section 3. The subsequent Section 4 addresses the RL-based online model reusing the input setup.

The rest of this paper is thus structured as follows: Section 2 introduces related work. Section 3 discusses offline symbolic music alignment. We develop as well as evaluate an offline symbolic music alignment algorithm based on two different applications of DTW, first on pitch information, then on onset times. Starting from these results, section 4 introduces a formulation of online alignment as reinforcement learning problem. In particular, we train an agent's value function in an offline setting. In section 5 we evaluate the trained agent in three ways: as a standalone score onset identification model, as an online symbolic alignment model (where the aim is the production of correct note-wise alignments), and in a score following scenario (where the aim is the precise temporal tracking the current score position). Finally, Section 6 concludes the paper with a critical appraisal of our models as well as recommendations for future research.

2. RELATED WORK

Symbolic music alignment has been a popular research area for many years. We begin our review of related work with online symbolic music alignment, then we progress

to offline symbolic music alignment, general music alignment, and finally applications of reinforcement learning.

Most often, online models have been presented in the context of score following, where the principal aim is to identify the current score position. Dannenberg [1] and Vercoe [2] pioneered this area of research in the mid 1980s. Recent works commonly use Dynamic Bayesian Networks to track the performance [3–6]. Recently, A. Anonymous compared both Hidden Markov Models (HMM) and On-Line Time Warping (OLTW) techniques. We use their OLTW model as comparison baseline for our online alignment technique. This model processes inputs represented as piano rolls, as is common for OLTW and DTW applications to symbolic music alignment in general.

The offline setting has seen more recent work [3, 5, 7–11]. Symbolic music alignment methods often perform very well, with error rates rarely exceeding 10%. Consequently, much recent work focused on the rare, indeterminate, or asynchronous events that make the errors difficult to identify and fix. Ornaments are one source of such events [9]. Another is left-right hand asynchrony in piano performance as discussed in Nakamura et al. [3]. Arbitrary skips and repeats further present a very difficult challenge for most algorithms, especially when runtime considerations are important [10]. This series of articles by Nakamura et al. [3, 9, 10] culminated in one of the most widely used automatic score-performance alignment tools and the current state of the art (SOTA) [11]. We use this model as reference for the evaluation of our offline algorithm.

Although beyond the scope of this article, no introduction of music alignment is complete without the mention of the large body of work concerning alignment of non-symbolic music formats, in particular audio. Wang [12], Arzt [13], and chapter three in Müller [14] present introductory discussions of audio alignment. As in our offline approach, applications of (non-standard) DTW are central to audio alignment [15–18]. Audio score following is commonly computed using On-Line Time Warping and variants of Hidden Markov Models [19–23].

To the best of our knowledge, Dorfer et al. [24] (later expanded upon by Henkel et al. [25]) are the only prior application of RL in a music alignment task, namely online audio to sheet music image alignment. For a general introduction to RL, we refer the reader to Sutton and Barto [26], for a discussion of the merits and disadvantages of offline RL to Levine et al. [27].

3. OFFLINE SYMBOLIC MUSIC ALIGNMENT

In this section, we introduce an offline symbolic music alignment based on two different DTW steps as well as an intermediate cleanup step. We close the section with an evaluation of our model against a state-of-the-art reference.

Symbolic music alignment produces *note alignments*, i.e., it matches individual notes of a performance recorded as MIDI file to individual notes of a score encoded as MusicXML file. Three types of note alignments exist: a match is tuple of a performance note and a score note, a deletion is a score note not played, and an insertion is a performed
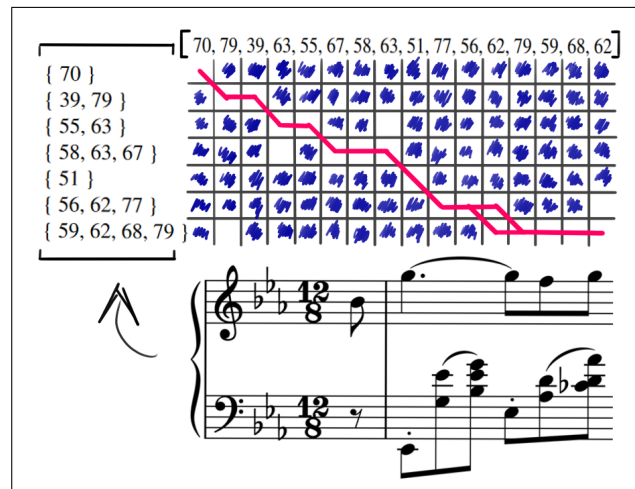

Figure 1. First half measure of Chopin Op. 9 No. 2 (bottom score), encoded as pitch set sequence (left) and warped to its performance, encoded as sequence of pitches as played (top). The matrix shows the corresponding pairwise distance (shaded is distance of 1, see equation 1), red lines indicate equivalent optimal warping paths.

note not notated.

Our proposed offline algorithm consists of the following steps: First, the performance and score are aligned using DTW on a purely pitch-based representation (Section 3.1). Then, remaining gaps are filled by complete sequences of a single pitch. Finally, individual notes are aligned using an application of DTW on their onset times (Section 3.2).

3.1 Pitch Sequence Warping

In this approach, we align sequences of performance notes, encoded as integer pitches $p_t \in \mathbb{I} := \{1...88\}$, with sequences of score onset notes, encoded as sets of integer pitches $s_t \in \mathcal{P}(\mathbb{I}) \setminus \{\varnothing\}$, with $\mathcal{P}(\mathbb{I})$ denoting the power set of the set $\mathbb{I}$. Since these sequence elements are of different types — integers and sets of integers — no standard local distance metric can be used. Instead we opt for a non-symmetric inclusion metric, with some abuse of the term.

$$m(p_t, s_t) = \begin{cases} 0 & \text{if } p_t \in s_t \\ 1 & \text{else} \end{cases} \quad (1)$$

Having defined the metric in Equation 1, two standard DTW paths are computed, one forward and one backward, i.e., using inverted sequences. Figure 1 shows the encoding as well as examplary DTW paths computed from cumulative pairwise distances. While the optimal DTW distance is unique, the DTW paths are not necessarily so. In our case, such ambiguity is often introduced by repeated pitches in neighboring score onsets, see e.g., the two adjacent (left, stacked vertically) notes of pitch D4/62 in Figure 1. To pinpoint non-robust path segments, we use the backward DTW path. Wherever the forward and backward paths disagree, they effectively bracket ambiguous parts from both sides, and we exclude all bracketed notes from the path.

These excluded segments are then processed using a simple heuristic: The notes in bracketed segments are sep-

arated by pitch. If two pitch-wise sequences with matching number of notes (in the performance and the score) are found, they are aligned and the result is added to the path. If no matching sequence is found, the path is linearly interpolated. We finally compute a mapping from score time to performance time from this merged and cleaned path.

3.2 Onset Sequence Warping

The next goal is to derive note-wise alignments from the approximate score to performance mapping computed in Section 3.1. To this end, the performance and score are split into pitch-wise sequences for each pitch occurring in the union of score and performance pitches. The approximate score-time-to-performance-time mapping computed in the previous step is used to project all score onsets (in beats) to performance time points (in seconds). The last step aligns the performance onset sequence with the score onset sequence, now mapped to the same space.

This alignment is computed by a DTW path between the onset sequences, this time using a simple L_1 metric, and for each non-unique alignment, keeping the tuple with the lowest distance. A threshold of maximal distance is further built-in (and set to 5 seconds) to avoid spurious alignment of unrelated deletions and insertions.

3.3 Offline Model Evaluation

To test the full offline model (first pitch DTW 3.1, then onset DTW 3.2), we compute alignments on four datasets of high-quality note-wise alignment piano music. The datasets are the Vienna 4x22 Dataset [28], four excerpts performed by 22 performers each; the Zeilinger Dataset [29], nine piano sonatas by Ludwig van Beethoven performed by Clemens Zeilinger; the Magaloff Dataset [30], the near complete solo piano works by Frederic Chopin performed by Nikita Magaloff; and the Batik Dataset [31], twelve piano sonatas by Wolfgang Amadeus Mozart performed by Roland Batik. We compare our model against the reference by Nakamura et al. [11], post-processed to produce the same output format we employ.

To compare produced alignments to ground truth ones, we have to define a metric. Recall that note alignments consist of three types: matches (tuples of performance and score notes), deletions (unplayed score notes), and insertions (unnotated performed notes). We use an F-score metric for matches: A predicted match is counted as a true positive (TP) only if the same notes are matched in the ground truth alignment. A false positive (FP) is a predicted note label that isn't in the ground truth, a false negative (FN) is a ground truth note label that isn't predicted. The F-score is defined as the harmonic mean of precision (TP / (TP + FP)) and recall (TP / (TP + FN)).

Table 3.3 shows dataset-wise and globally averaged F-scores for matches. Our proposed model outperforms the reference on each dataset. A two-sided sign test on performance-wise rankings shows significantly ($\alpha = 0.01$) higher performance for our proposed model on all datasets except the Vienna 4x22 Dataset. On the Vienna 4x22 dataset, the models reach the same F-score of 1.0 for 38

Dataset	DTW Offline	Nakamura
Magaloff	$98.4 \pm 0.9\,\%$	$97.8 \pm 1.4\,\%$
Zeilinger	$99.3 \pm 0.9\,\%$	$98.8 \pm 1.2\,\%$
Batik	$99.4 \pm 0.7\,\%$	$98.5 \pm 2.1\,\%$
Vienna 4x22	$99.8 \pm 0.4\,\%$	$99.5 \pm 0.5\,\%$
Combined	$99.0 \pm 1.0\,\%$	$98.5 \pm 1.5\,\%$

Table 1. Dataset-wise averaged F-scores and standard deviations of each model.

performances and our proposed model has higher F-scores for the remaining 50 performances.

4. ONLINE ALIGNMENT AGENT

Having established the effectiveness of the separation into pitch-based and time-based input representations in the offline setting, we now introduce a formulation of RL-based online alignment. We continue with the model and training setup used to approximate the agent's value function.

Reinforcement learning is formalized as Markov Decision Process (MDP). An MDP consists of the following components: a state space $\mathcal{S}$, an action space $\mathcal{A}$, an index set $\mathcal{T}$, a reward function $\mathcal{R}$, transition probabilities $\mathcal{P}$, and a discount factor γ.

An agent is placed in an environment and perceives this environment and itself as being in a possible state $S_t \in \mathcal{S}(t \in \mathcal{T})$. The agent now takes an action $A_t \in \mathcal{A}$ and receives a reward R_{t+1} as well as a new state $S_{t+1} \in \mathcal{S}$. Repeating this process iteratively yields a sequence of states, actions, and rewards, called an episode: $S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1}, R_{t+2}, S_{t+2}, A_{t+2}, ...$ It is now the agent's task to infer actions from states that maximize long-term reward. Before we look at our formulation of this optimization problem, we discuss the online alignment's state and action space in more detail.

4.1 Alignment as Reinforcement Learning

The state information S_t comprises both the current score context as well as the most recent past performance. Specifically, the score context is represented as a window of the pitch set sequence introduced in Section 3.1. The window centers the last predicted score onset position and spans seven score onsets to the past as well as eight score onsets to the future for a total windowed sequence of 16 pitch sets. The performance context is only derived from past performance notes to enable real-time application. It consists of the eight most recent notes in the performance pitch sequence. Whenever less score or performance context is available, e.g., at the very beginning or end of a piece, the windows are shortened accordingly.

At each state S_t the agent aims to match the most recent performance note to its most likely score onset. There are 16 actions at S_t; select one score onset as matching onset position. Having decided on a next score onset, the agent receives a reward R_{t+1} which is set to one if the score onset is correctly aligned, zero otherwise. The environment transition probabilities $\mathcal{P}$ determine a new state S_{t+1}: the

performance context of the new state is determined by an actual performance, i.e., the agent is presented a new state based on the estimated next position in the score and a new incoming performance note.

4.2 Simplified Deep Q-learning

The agent's behavior is captured by its policy $\pi(A|S)$, the distribution of actions taken by the agent in state S. Although it is possible to optimize the policy directly, we instead adapt a value-function-based formulation, or more specifically, deep Q-learning [32, 33]. Q-learning aims to optimize a state-action value function $Q : \mathcal{S} \times \mathcal{A} \longrightarrow \mathbb{R}$, an estimate of the expected cumulative discounted future reward, also called return, given a state and an action. In deep Q-learning the value function $Q(S, A, \theta)$ contains trainable parameters θ which are fitted to the experienced reward distribution. A typical optimization loss l looks like this:

$$l = (R_{t+1} + \gamma \max_{A_{t+1}} Q(S_{t+1}, A_{t+1}, \theta) - Q(S_t, A_t, \theta))^2$$

(2)

where the discount factor $\gamma \in [0, 1]$ determines the weighting of future rewards. For the alignment case we can make several simplifications. We opt for a completely myopic agent, i.e., $\gamma = 0$. The argument for this is that the optimal action to take for each incoming performance note is determined by the correct score onset which can in turn be specified by the immediate reward. >For a discussion of the implications of this modeling choice see section 6. Setting $\gamma = 0$ removes the value function at subsequent states from the loss. Using that $\mathcal{R} = \{0, 1\}$, we make a second reformulation and replace this squared error loss by a binary classification: For each state-action tuple (S, A) the agent predicts the probabilities of the reward being 1 or 0, optimized with a cross-entropy loss. Note that this formulation still optimizes a state-action value function Q and not a policy $\pi(A|S)$, i.e., the probabilities of rewards are computed for each possible action (that is, per score onset) and do not sum to one over all actions. There are several ways of deriving a policy from a value function; two are discussed in section 4.5.

4.3 Value Function Model

To approximate $Q(S, A, \theta)$, we use an attention-based Transformer Neural Network. The input of the network consists of a sequence of tokens encoding the performance, a delimiter token, the score, and an ending token. We encode 88 pitches of the piano keyboard, adding extra tokens for "no_pitch", "delimiter", and "end" in a 64-dimensional embedding space. The performance pitches are straightforward to embed, however, the score onset pitch sets require more processing. For our data, more than 99% of score onsets can be represented with pitch sets with no more than seven different pitches, we thus limit our pitch sets to this length (with a subset of seven taken randomly at onsets with more pitches). Pitch sets with fewer than seven pitches are filled up with a pitch corresponding to the
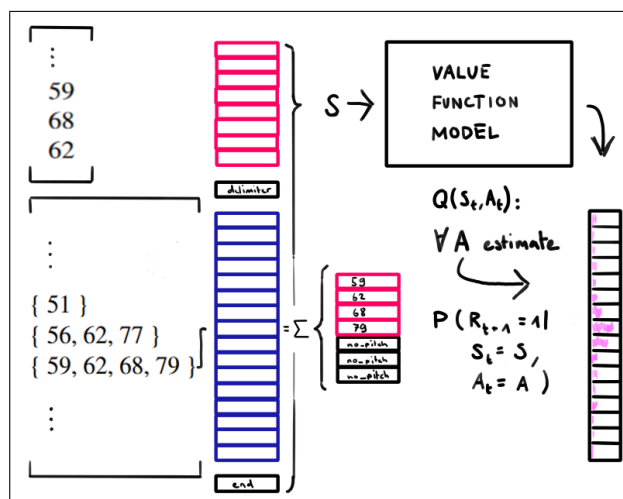

Figure 2. Setup of the value function model: states are encoded as contiguous token sequence of past performance (red) and current score (blue) contexts. Pitch set embeddings are summed over individual pitch embeddings. The model is set up as token classifier as each score onset in the context corresponds to a possible action (= "select this onset as next score onset") and is classified according to its expected reward class. The vector on the right shows the reward probability for each action (pink).

"no_pitch" token. To create score onset embedding with no more than 64 dimensions independent of the number of pitches at any onset, the pitch set tokens are summed up. Figure 2 shows the setup of the value function model. The model is set up with eight heads, and six layers, layer normalization, and a single feedforward head for binary classification, making for a total of 157250 parameters.

4.4 Training

The training is set up as token classification problem; i.e., for each token in the sequence, the probability of receiving a reward is estimated. The aligned piano datasets from Section 3.3 are used again. For our offline RL setting, a dataset of states is created before training. We extract local score and performance contexts from the aligned data, shifting the performance window such that true next score onsets fall from leftmost (current position minus seven) to rightmost (current position plus eight) to cover the possible states exhaustively. During training, batches of states are sampled randomly, not in sequence. To aid generalization, we further augment the data by random pitch shifting of all notes in the state within +/- one octave. We use an ADAM optimizer with a learning rate with warm-up followed by square root decay. The batch size is set to 8192, the models are trained for 50 epochs.

4.5 Online Models

Using a trained value function model, we derive two complete models. First, a simple score follower model ("Greedy Agent Model") that outputs only greedily estimated score onsets for incoming performance notes. Sec-
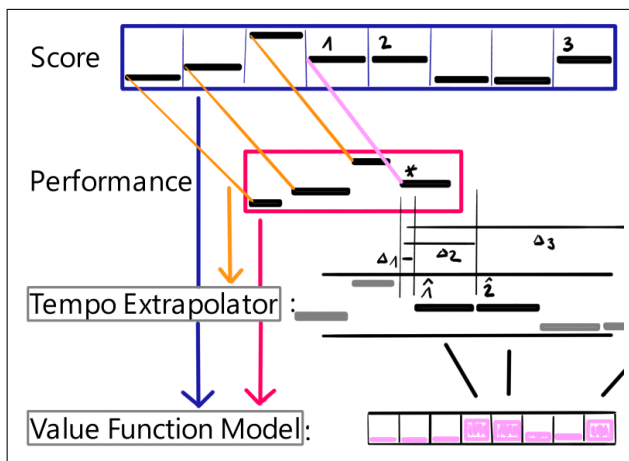
Figure 3. Schematic overview of the Online Alignment Model with a monophonic piece and 8 onset context. Score (blue, top) and performance (red) contexts are inputs to the Value Function Model which outputs value estimates (pink, bottom). The top three onsets (1,2,3) are passed to a tempo extrapolator, along with existing alignments (yellow). The tempo extrapolator predicts three onsets $(\hat{1}, \hat{2})$ for the the candidate onsets. The one with lowest distance (Δ_1) to the newest performance note (*) is aligned (pink).

ond, a note-level alignment model ("Online Alignment Model") that produces both note alignments as well as an estimate of the next score onset for the score following setting. The Greedy Agent Model consists of an agent following a greedy policy based on the trained value function model, i.e., an agent picking the action A with maximal estimated reward $Q(S, A)$.

For the Online Alignment Model a few additional steps are taken. See Figure 3 for an overview of the alignment loop. In this model, an action is selected from the top three value estimates for a given score and performance context. To pick one of these three actions, onset time information is incorporated. A simple local tempo estimator approximates an expected performed onset time for each of the three possible score onsets using linear extrapolation of beat periods computed from previously aligned notes. This process takes on the role of the second onset-wise DTW step in the offline model (see Section 3.2): to match notes that are closest together according to an approximate score-to-performance mapping.

There are two further heuristics worth mentioning. If the current performed pitch is not available at any of the three highest ranking score positions, the performed note is counted as an insertion, and the current score position is unchanged. Furthermore, we decrease the number of calls made to the agent in a real-time setting by directly aligning pitches that are trivially missing at the current score onset.

5. ONLINE EVALUATION

In the following, we evaluate the agent and the proposed online alignment model. In section 5.1, we address a greedy agent's capacity to identify correct score positions for sampled test contexts. In section 5.2, the Online Align-

ment Model is evaluated with respect to correct note-wise alignment. Finally, we use both the Greedy Agent Model as well as the Online Alignment Model as real-time symbolic score followers in an experiment in section 5.3.

5.1 Agent Evaluation

Top0	Top1	Top2
94.5 ± 0.8 %	96.6 ± 0.5 %	97.6 ± 0.4 %

Table 2. Average topK score onset hit rate and standard deviation across the five test folds.

For direct value function evaluation, we assume a greedy policy for each testing state S. That is, the agent picks the action A with the highest estimated value $Q(S, A)$. We evaluate this action (= chosen score onset) via the distance from the ground truth score onset. Specifically, we compute three metrics: the number of states where this action corresponds exactly to the true score onset ("Top0"), the number of times this action picks a score onset in the neighborhood of $\pm$ one score onset of the true location ("Top1"), and the number of times this action picks a score onset in the neighborhood of $\pm$ two score onsets of the true location ("Top2"), each normalized by the total number of test states. We use five-fold cross-validation on the same combined datasets used in section 3.3, and report mean and standard deviation values across testing folds. The fold splitting is carried out piece-wise with roughly the same number of score onsets in each fold.

Table 2 shows the results. Greedy action selects the correct score onset with more than 94 % probability on unseen pieces. Furthermore, for more than half of the remaining errant actions, the greedy action is not further than two onsets from the correct one.

5.2 Online Note-wise Alignment

Piece	OAM	DTW Offline	Nakamura
B. Op. 53 3rd. m.	99.0 %	99.4 %	98.2 %
C. Op. 9 No. 1	97.6 %	98.4 %	98.8 %
C. Op. 9 No. 2	97.4 %	99.1 %	97.6 %
C. Op. 10 No. 11	90.3 %	96.3 %	94.3 %
C. Op. 60	95.1 %	97.9 %	94.7 %

Table 3. Piece-wise F-scores of each model. OAM = Online Alignment Model, DTW Offline = model of section 3.3, Nakamura = reference SOTA model [11].

To evaluate the Online Alignment Model's performance, we perform alignments for five selected performances: Nocturnes Op. 9 No. 1 and 2, Etude Op. 10 No. 11, Nocturne Op. 15 No. 2, the Barcarole Op. 60 by F. Chopin, and the third movement of the Sonata Op. 53 (Waldstein) by L. v. Beethoven. The value function model used in this section was trained on all data except these five pieces for 100 epochs, the rest of the training and model setting remains the same. The same metrics of section 3.3

Model	Async	≤ 25ms	≤ 50ms	≤ 100ms
OLTW	60.6 ms	38.0 %	63.3 %	86.7 %
GAM	36.0 ms	89.0 %	91.4 %	94.6 %
OAM	15.7 ms	91.4 %	93.8 %	96.6 %

Table 4. Asynchrony of the models in score follower setting. Column "Async" presents the median asynchrony. Columns 3, 4, 5 present the percentage of onset estimates with lower asynchrony than 25ms, 50ms and 100ms, respectively.

apply, namely the F-score of retrieved matched note tuples. For comparison we also add piece-wise F-scores of our proposed offline model as well as the model by Nakamura et al. Table 3 shows the F-scores of note alignments. As expected, the proposed online alignment performs worse than offline methods, albeit with small difference. Notably, all models show the lowest performance on Chopin's Op. 10 No. 11.

5.3 Score Following

In the score following setting, the core metric is the accurate prediction of the current position. We thus compute asynchrony values in milliseconds which give the absolute time between any onset in a performance and the onset in the same performance that corresponds to the estimated score onset. The data used for this experiment consists of the same five pieces used in the previous section 5.2 with the same value function model training. Three models are compared in this setting: The Online Alignment Model (OAM) previously evaluated in terms of note alignment F-scores in Table 3, the Greedy Agent Model (GAM), and an On-Line Time Warping (OLTW) Model. This latter OLTW model performed best in a recent music score following comparison by Cancino-Chacón et al. [34] and is added as a reference. However, this model does not predict note alignments, hence a comparison in terms of note alignment F-score as in Section 5.2 is not possible.

Both the GAM and the OAM outperform the reference model in all metrics in Table 4. Most of the lower performance of the GAM is due to the fact that for Chopin's Op. 10 No. 11, this agent loses track of the performance close to the end when a full measure is deleted. All subsequent performance notes are estimated very wrongly. The online alignment model on the other hand follows all test performances robustly until the end.

6. DISCUSSION AND CONCLUSION

In this paper, we introduce two models, an offline alignment model based on dual DTW steps, and an online alignment model based on an RL agent trained in an offline fashion. Both models perform competitively; with the offline model surpassing the relevant state of the art.

In the setup of the RL training we made some simplifications that warrant further discussion. Specifically, we set the discount factor γ to zero and train using a dataset of sampled states. In section 4.2, we claim that the optimal action for each step is determined by the correct score onset. While this is true for the states in the dataset and if optimality is defined by accuracy in note-wise alignment, it might not be for out-of-distribution states or if the focus of the agent is shifted to robustness, i.e., following the entire performance even at the cost of some misaligned notes.

For offline RL, a crucial issue is distributional drift [27]; i.e., the fact that the agent learns from states that follow a different distribution that the states it would encounter in an online setting. Even though we can sample the state space exhaustively for the training set, out-of-distribution states are expected in test sets consisting of previously unseen pieces and performances. Furthermore, the relative frequency of training samples does not necessarily correspond to the states an online agent is likely to see during training, where all target locations have the same frequency. Specifically, for an agent that already learned to predict the score onset with some accuracy, the targets at the limits of the context are going to be less frequent than the center ones. In other words, a non-myopic online agent is likely to behave more conservatively, avoiding large skips as they do not occur that frequently in actual performances.

On the other hand, the offline RL formulation successfully leverages prior knowledge about the task and — more importantly — stabilizes the gradient, rendering the training of a complex value function approximator feasible. Future work includes shifting this trade-off back towards online RL, for example with online RL training after initial offline training.

The RL agent learns to align purely on pitch information. Including onset or even duration information is likely to increase the accuracy of following at the cost of requiring a more expressive model which in turn affects inference speed — a hard bottleneck for real-time application. In fact, running the value estimation for every incoming performance note (such as the score follower "GAM" in Table 4) uses up to a minute of computation time for the 7273 notes in the performance of Beethoven's Op. 53 Mvt. 3 (Roughly 10 ms per note). A further increase is liable to affect real-time score following in fast passages.

In terms of post-processing steps, both our offline and online models are comparatively crude, making little use of score information such as ornaments. As Nakamura et al. [11] correctly remark, their post-processing step is in principle able to improve upon any prior more error-prone alignment. Further research is needed to know whether the offline model can be improved in this way.

To conclude, we developed and evaluated two models of symbolic music alignment which both outperform relevant prior work. To the best of our knowledge, this RL-based online alignment model is one of the first applications of not only trainable but effectively *trained* models to symbolic music alignment.

7. REPRODUCIBILITY

Our alignment models are available at: `https://github.com/sildater/parangonar`

8. ACKNOWLEDGEMENTS

9. REFERENCES

[1] R. B. Dannenberg, "An On-Line Algorithm for Real-Time Accompaniment," in *Proceedings of the International Computer Music Conference (ICMC)*, vol. 84, 1984, pp. 193–198.

[2] B. Vercoe, "The Synthetic Performer in the Context of Live Performance," in *Proceedings of the International Computer Music Conference (ICMC)*, 1984, pp. 199–200.

[3] E. Nakamura, N. Ono, Y. Saito, and S. Sagayama, "Merged-Output Hidden Markov Model for Score Following of MIDI Performance with Ornaments, Desynchronized Voices, Repeats and Skips," in *International Conference on Mathematics and Computing*, 2014.

[4] C. Raphael and Y. Gu, "Orchestral Accompaniment for a Reproducing Piano," in *International Conference on Mathematics and Computing*, 2009.

[5] E. Nakamura, P. Cuvillier, A. Cont, N. Ono, and S. Sagayama, "Autoregressive hidden semi-markov model of symbolic music performance for score following," in *16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[6] A. Maezawa, H. G. Okuno, T. Ogata, and M. Goto, "Polyphonic audio-to-score alignment based on bayesian latent harmonic allocation hidden markov model," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2011, pp. 185–188.

[7] B. Gingras and S. McAdams, "Improved Score-Performance Matching using Both Structural and Temporal Information from MIDI Recordings," *Journal of New Music Research*, vol. 40, no. 1, pp. 43–57, 2011.

[8] C.-T. Chen, J.-S. R. Jang, and W. Liou, "Improved Score-Performance Alignment Algorithms on Polyphonic Music," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 1365–1369.

[9] E. Nakamura, N. Ono, S. Sagayama, and K. Watanabe, "A Stochastic Temporal Model of Polyphonic MIDI Performance with Ornaments," *Journal of New Music Research*, vol. 44, no. 4, pp. 287–304, 2015.

[10] E. Nakamura, T. Nakamura, Y. Saito, N. Ono, and S. Sagayama, "Outer-product hidden markov model and polyphonic midi score following," *Journal of New Music Research*, vol. 43, no. 2, pp. 183–201, 2014.

[11] E. Nakamura, K. Yoshii, and H. Katayose, "Performance Error Detection and Post-Processing for Fast and Accurate Symbolic Music Alignment," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2017, pp. 347–353.

[12] S. Wang, "Computational Methods for the Alignment and Score-Informed Transcription of Piano Music," Ph.D. dissertation, Queen Mary University of London, London, UK, 2017.

[13] A. Arzt, "Flexible and robust music tracking," Ph.D. dissertation, Johannes Kepler University Linz, Linz, Austria, 2016.

[14] M. Müller, *Fundamentals of Music Processing – Audio, Analysis, Algorithms, Applications*. Springer, 2015.

[15] T. Prätzlich, J. Driedger, and M. Müller, "Memory-restricted multiscale dynamic time warping," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 569–573.

[16] M. Müller, Y. Özer, M. Krause, T. Prätzlich, and J. Driedger, "Sync toolbox: A python package for efficient, robust, and accurate music synchronization," *Journal of Open Source Software*, p. 3434, 2021.

[17] C. J. Tralie and E. Dempsey, "Exact, parallelizable dynamic time warping alignment with linear memory," in *21st International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[18] S. Ewert, M. Muller, and P. Grosche, "High resolution audio synchronization using chroma onset features," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2009.

[19] S. Dixon, "An On-Line Time Warping Algorithm for Tracking Musical Performances," in *International Joint Conference on Artificial Intelligence*, 2005, pp. 1727–1728.

[20] A. Cont, "ANTESCOFO: Anticipatory Synchronization and Control of Interactive Parameters in Computer Music." in *Proceedings of the International Computer Music Conference (ICMC)*, 2008, pp. 33–40.

[21] C. Raphael, "Music Plus One and Machine Learning," in *International Conference on International Conference on Machine Learning*, 2010, pp. 21–28.

[22] Z. Duan and B. Pardo, "A State Space Model for Online Polyphonic Audio-Score Alignment," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2011, pp. 197–200.

[23] A. Arzt and G. Widmer, "Real-Time Music Tracking Using Multiple Performances as a Reference," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[24] M. Dorfer, F. Henkel, and G. Widmer, "Learning to Listen, Read, and Follow: Score Following as a Reinforcement Learning Game," in *Proceedings of the 19th International Society for Music Information Retrieval Conference*. Paris, France: ISMIR, Sep. 2018, pp. 784–791. [Online]. Available: https://doi.org/10.5281/zenodo.1492535

[25] F. Henkel, S. Balke, M. Dorfer, and G. Widmer, "Score following as a multi-modal reinforcement learning problem," *Transactions of the International Society for Music Information Retrieval*, vol. 2, no. 1, 2019.

[26] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[27] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[28] W. Goebl. (1999) The Vienna 4x22 Piano Corpus. [Online]. Available: http://repo.mdw.ac.at/projects/IWK/the_vienna_4x22_piano_corpus/index.html

[29] C. E. Cancino-Chacón, T. Gadermaier, G. Widmer, and M. Grachten, "An Evaluation of Linear and Non-linear Models of Expressive Dynamics in Classical Piano and Symphonic Music," *Machine Learning*, vol. 106, no. 6, pp. 887–909, 2017.

[30] S. Flossmann, W. Goebl, M. Grachten, B. Niedermayer, and G. Widmer, "The Magaloff Project: An Interim Report," *Journal of New Music Research*, vol. 39, no. 4, pp. 363–377, 2010.

[31] G. Widmer and A. Tobudic, "Playing mozart by analogy: Learning multi-level timing and dynamics strategies," *Journal of New Music Research*, vol. 32, no. 3, pp. 259–268, 2003.

[32] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, pp. 279–292, 1992.

[33] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[34] C. Cancino-Chacón, S. Peter, P. Hu, E. Karystinaios, F. Henkel, F. Foscarin, N. Varga, and G. Widmer, "The accompanion: Combining reactivity, robustness, and musical expressivity in an automatic piano accompanist," in *International Joint Conference on Artificial Intelligence*, 2023.