Skipping the Frame-Level: Event-Based Piano Transcription With Neural Semi-CRFs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Piano transcription systems are typically optimized to estimate pitch activity at each frame of audio. They are often followed by carefully designed heuristics and post-processing algorithms to estimate note events from the frame-level predictions. Recent methods have also framed piano transcription as a multi-task learning problem, where the activation of different stages of a note event are estimated independently. These practices are not well aligned with the desired outcome of the task, which is the specification of note intervals as holistic events, rather than the aggregation of disjoint observations. In this work, we propose a novel formulation of piano transcription, which is optimized to directly predict note events. Our method is based on Semi-Markov Conditional Random Fields (semi-CRF), which produce scores for non-overlapping continuous intervals rather than individual frames. When formulating piano transcription in this way, we eliminate the need to rely on disjoint frame-level estimates for different stages of a note event. We conduct experiments on the MAESTRO dataset and demonstrate that the proposed model surpasses the current state-of-the-art for piano transcription. Our results suggest that the semi-CRF output layer, while still quadratic in complexity, is a simple and fast solution for event-based prediction, and may lead to similar success in other areas which currently rely on frame-level estimates.

1 Introduction

The task of Automatic Music Transcription (AMT) aims to transcribe a music recording into some form of music notation [Benetos et al., 2018]. Examples of notation include MIDI event sequences, e.g, Hawthorne et al. [2018], Kong et al. [2020], Kim and Bello [2019], Kwon et al. [2020], or staff notation, e.g., Nakamura et al. [2018], Román et al. [2018, 2019]. In this work, we address the problem of transcribing piano music into a MIDI event sequence. MIDI transcription involves constructing a sequence of events, each specified by its onset and offset (beginning and ending) positions, with the constraint that two events of the same event type (certain pitches and pedals) cannot overlap. In addition to onsets and offsets, the velocity (a value that represents the intensity, which informs the loudness, of the key strike) associated with each event is often estimated.

In recent years, neural network based approaches have reached the state of the art for the problem of piano transcription. Contemporary piano music transcription models, e.g., Hawthorne et al. [2018], Kong et al. [2020], Kwon et al. [2020], operate at the frame-level and generate scores for different stages of a note (event), i.e., the onset, offset, and pitch activation, separately. In order to extract intervallic note (event)-level predictions, they use manually designed heuristics to combine the disjoint frame-level predictions. These include thresholding, peak picking [Hawthorne et al., 2018, Kong et al., 2020] or state transition based probabilistic inference [Kwon et al., 2020]. This two-stage practice requires manually crafted procedures and a manually designed set of states that model

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

the temporal evolution of a note. Furthermore, state-of-the-art systems like Hawthorne et al. [2018], Kong et al. [2020] utilize a cascade of neural network processing blocks to predict the activation of note components sequentially. These blocks take as input the original spectrum, and in some cases the activation signals previously predicted. This formulation of music transcription is complex and requires significant computation. In this work, we propose to generate note-level predictions in a more structured way, by using (0th-order) semi-markov conditional random fields (semi-CRF) [Sarawagi and Cohen, 2004] to score note events holistically instead of scoring and aggregating activations across individual frames. This method builds a distribution over a set of non-overlapping intervals for a certain event type (i.e., piano notes, pedals) and thus simplifies the formulation of piano transcription.

We introduce a specialized zeroth order Semi-CRF formulation and show that it can be implemented efficiently (see Table 4) for both training and inference when dealing with problems similar in complexity to piano transcription (400-1600 frames with around 90 labeling channels), without applying an upper bound on the interval length as commonly seen in the literature, e.g., Zhuo et al. [2016]. The Semi-CRF layer promotes a direct formulation for event (interval) prediction, eliminating the need for estimating activations at the frame-level followed by post processing.

We believe that this simple, fast and well-performing approach is also extensible to other similar tasks with intervals as the prediction target, such as polyphonic sound event detection and speaker diarization.

2 Related Works

Piano Transcription There have been many proposed approaches to piano transcription in the last several decades. Early works consist of simple signal processing methods such as spectral peak-picking [Klapuri et al., 2000, Bello et al., 2006], which estimates fundamental frequencies directly from the spectrum, or spectral decomposition [Smaragdis and Brown, 2003, O'Hanlon and Plumbley, 2014], where a short-time spectrum is factorized into spectral components with corresponding activations. Several parametric models have also been proposed to learn note-related parameters [Vincent et al., 2009, Emiya et al., 2009, Cheng et al., 2016]. Most of these aim to model notes explicitly, and tend to suffer from lack of generality. Other methods include machine learning techniques like Hidden Markov Models [Raphael, 2002, Böck and Schedl, 2012] or Support Vector Machines [Poliner and Ellis, 2006, Weninger et al., 2013]. Recently, neural network based approaches have made significant advances [Sigtia et al., 2016, Hawthorne et al., 2018, 2019, Kim and Bello, 2019] by implicitly modeling notes using large databases of MIDI performances. Some more recent methods attempt to fold note modeling into a DNN [Kelz et al., 2019, Kwon et al., 2020].

Many methods separate piano transcription into frame-wise polyphonic pitch estimation and note tracking, thereby optimizing for frame-level predictions and quantizing estimated note intervals. Kong et al. [2020] address the problem by regressing between active frames to predict continuous onset and offset times, but still adopt the disjoint approach to assembling note predictions. In contrast, we directly estimate note intervals, while still performing regression to obtain continuous times. One proposed event-based method [Kameoka et al., 2007] performs harmonic template structured clustering to explain spectrogram observations as originating from distinct sources. Cogliati et al. [2016, 2017] similarly estimate the activation of time-domain note templates, but these templates lack generalization and must be re-acquired for every piano and acoustic environment.

Semi-Markov CRFs A semi-Markov conditional random field (semi-CRF, Sarawagi and Cohen [2004]) defines a conditional probabilistic distribution over sets of non-overlapping labeled segments within an input sequence. This formulation has been used for Chinese word segmentation [Liu et al., 2016, Kong et al., 2015], named entity recognition [Zhuo et al., 2016, Ye and Ling, 2018, Arora et al., 2019], character-level part-of-speech tagging [Kemos et al., 2019], chord recognition [Masada and Bunescu, 2017], etc. The time complexity for computing the partition function and inferring the most likely configuration are quadratic with respect to the length of the sequence if no restriction be made on the maximum length of an interval. Therefore, most works set an upper bound on the length, e.g., Kemos et al. [2019] .

The semi-CRF layer in this work differs from the standard formulation in following ways: 1. intervals/segments are allowed to overlap on endpoints (boundary) as each frame position is quantized

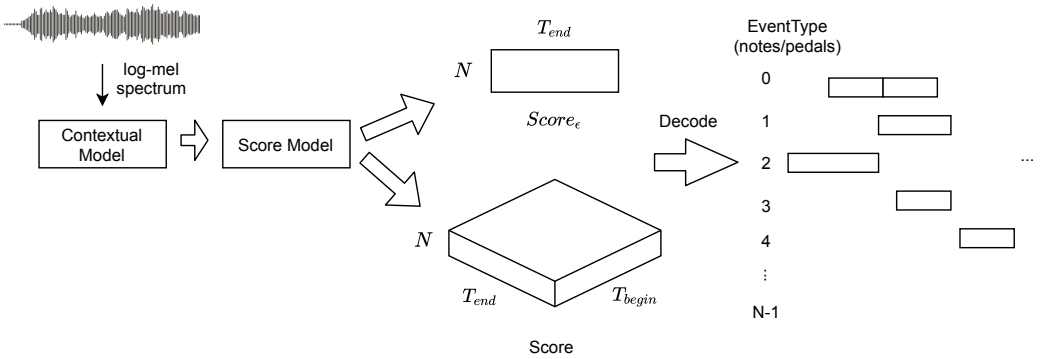


Figure 1: Overview of the proposed system.

from the original continuous time position; 2. a position (frame) are allowed to be empty, not belonging to any interval. Compared with tasks above-mentioned, the task of piano transcription has a longer input sequence with each event type (88 pitches and 1-3 pedals) defines a separate semi-CRF. Also note/pedal events (intervals) have a larger range of durations which makes it hard to decide a limit on duration for a speedup. After mildly optimizing the implementation for this simplistic semi-CRF layer, we found that it becomes efficient for the problem size of piano transcription without applying a limit on the event duration.

3 Proposed Semi-CRF Approach to Piano Transcription

The proposed system transcribes the input audio into a list of musical events (i.e., notes and pedals) in a segment-wise fashion. Taking an audio segment (e.g., 10s), the transcription process is illustrated in Figure 1. A log-mel spectrogram is first computed using short-time Fourier transform (STFT) as the input to a *contextual model* to aggregate audio features across time frames. Such features are then fed to a *score model* to calculate two kinds of scores. The first kind of score indicates whether an audio frame is part of an event for a certain event type, and the second kind of score evaluates the likelihood of an arbitrary time interval being such an event. Finally, a Viterbi algorithm is used to decode the most likely sequence of events for each event type as well as their attributes.

3.1 CRF Formulation

Let $\mathcal{X} =< x_0, x_1, \ldots, x_{N-1} >$ be an audio segment containing a sequence of N time frames. Let $\mathcal{Y} = \{(i, j, \textit{eventType}), i \leq j\}$ be the set of musical events entirely contained within this segment, with time quantized to audio frames. Here i and j are the beginning and ending frame indices for each event, and *eventType* is the type of the event, e.g., a specific pitch of the 88 pitches of a piano, the sustain pedal usage. Events that extend outside the audio segment are not considered in this formulation, but will be handled in a post-processing step in the inference process (See Section XXX). In this formulation, we allow single-frame events where $i = j$. We also assume that for the same event type, two events A and B are disjoint, i.e., either $j_A \leq i_B$ or $j_B \leq i_A$. Finally, we associate each event with a set of attributes: 1) the non-quantized position of onsets and offsets at the sub-frame level, represented by a value from 0 to 1 indicating the relative position within a frame, and 2) velocity of the event, represented by a discrete value from 0 to 127.

We use $\mathcal{Y}_{\textit{eventType}}$ to denote the subset of events that contains only a specific event type. For each event type, we model the following conditional probability:

$$p_\theta(\mathcal{Y}_{\textit{eventType}}|\mathcal{X}) = \frac{1}{Z(eventType)} \exp\{ \sum_{(i,j,\textit{eventType}) \in \mathcal{Y}_{\textit{eventType}}} score(i, j, eventType) + \sum_{[i-1,i] \text{ not covered in } \mathcal{Y}_{\textit{eventType}}} score_\epsilon(i-1, i, \textit{eventType}) \quad \}, \tag{1}$$

where $score(i, j, eventType)$ assigns a score to indicate how likely the interval $[i, j]$ is an event of *eventType*; $score_\epsilon(i-1, i, \textit{eventType})$ assigns a score for the interval $[i-1, i]$ to indicate how likely it is not covered by any event of *eventType*; and $Z(\textit{eventType})$ is the normalization factor. Here, for notational convenience, we omit $\mathcal{X}$ for every term.

It is noted that the Eqn. (1) is computed in the log-domain. The exponent, i.e., the summation of all $score(i, j, \textit{eventType})$ and $score_\epsilon(i-1, i, \textit{eventType})$ corresponding to $\mathcal{Y}_{\textit{eventType}}$, is the unnormalized the likelihood. Table 1 provides an example of an interval sequence candidate for a given *eventType*, and illustrates how their corresponding unnormalized log-likelihoods are computed.

Intervals	[0,0], [2,4], [4,5]
Unormalized log-likelihood	$score(0,0) + score_\epsilon(0,1) + score_\epsilon(1,2)$ $+score(2,4) + score(4,5) + score_\epsilon(5,6)$

Table 1: Example interval sequence candidate of a specific *eventType*. Here we assume there are 7 audio frames <0,1,2,3,4,5,6> in total. The *eventType* is omitted in each term for clarity.

The computation of $\log Z(eventType)$ and its gradient w.r.t. to θ, $\nabla \log Z(eventType)$, is critical in training. Here we propose a *forward-backward* algorithm, as shown in Algorithm 1. In practice, we compute $\log Z$ and $\nabla \log Z$ for all eventType(s) in parallel. The forward stage and the backward stage in the forward-backward algorithm are batched to compute in a single pass, as their calculations are essentially the same, but with all the positions flipped. In order to make memory access at each step more contiguous, we use a storage layout for the $score$ tensor with shape $T_{end} \times T_{start} \times N_{batch}$, with the first dimension being the end position of the interval, the second dimension being the start position, and the third dimension being the indices inside a batch. We also found that when using a GPU, a substantial speedup can be achieved by using a custom gradient computation via the backward pass of the forward-backward algorithm for $\log Z$, compared with using automatic differentiation w.r.t. $\log Z$ in *PyTorch*. These simple optimizations make the semi-CRF layer efficient for the problem size of interest. An efficiency benchmark is provided in Table 4.

3.2 Training Objectives

For training, we use maximum likelihood estimation (MLE), where the conditional log-likelihood is defined to consolidate the conditional probability in Eq. (1) over all event types, assuming their conditional independence given $\mathcal{X}$:

$$\log p_\theta(\mathcal{Y}|\mathcal{X}) = \sum_{\textit{eventType}} \log p_\theta(\mathcal{Y}_{\textit{eventType}}|\mathcal{X}). \tag{2}$$

One may question the validity of this conditional independence assumption for piano transcription, this is a simple treatment similar to naive Bayes.

In addition to the presence log-likelihood of events defined in Eqn. 2, we also learn to predict three attributes for each interval:

$$\log p_\theta(\textit{attributes}|e) = \log p_\theta(\textit{velocity}|e) + \log p_\theta(\textit{onset refine}|e) + \log p_\theta(\textit{offset refine}|e). \tag{3}$$

Here we use e to denote an event. We parameterize these terms with the following distributions:

$$\begin{aligned} \textit{velocity}|e &\sim \text{Softmax}(\mu(e)), \\ \textit{onset/offset refine}|e &\sim \text{ContinuousBernoulli}(\lambda(e)), \end{aligned} \tag{4}$$

where $\mu(e)$ and $\lambda(e)$ are parameters produced by neural networks that take features of the interval as the input. The final objective is defined as

$$\mathfrak{L} = -(\log p_\theta(\mathcal{Y}|\mathcal{X}) + \sum_{e \in \mathcal{Y}} \log p_\theta(\textit{attributes}|e)). \tag{5}$$

More discriminative and cost-sensitive losses such as max-margin and softmax margin can be used as drop-in replacements, but we leave this investigation to future work.

Algorithm 1 Forward-backward algorithm for $\log Z$ and $\nabla logZ$ for a specific event type.

Input: function $score(i, j)$, function $score_\epsilon(i-1, i)$
Output: $\log Z$ and $\nabla \log Z$

Forward stage:
Initialize the forward variable: $v(0) \leftarrow \log(\exp(\ score(0,0)\ ) + 1)$
for all $j = 1, \ldots, N-1$ **do**

$$v(j) \leftarrow \log\left(\exp\{v(j-1) + score_\epsilon(j-1, j)\ \} + \sum_{k<j} \exp\{v(k) + score(k, j)\}\right)$$

$$v(j) \leftarrow v(j) + \log(1 + \exp\{score(j, j)\ \}\ )$$

end for

Readout the log partition function:
$\log Z \leftarrow v(N-1)$

Backward stage:
Initialize the backward variable: $q(N-1) \leftarrow \log(\exp(\ score(N-1, N-1)\ ) + 1)$
for all $j = N-2, \ldots, 0$ **do**

$$q(j) \leftarrow \log\left(\exp\{q(j+1) + score_\epsilon(j, j+1)\ \} + \sum_{k>j} \exp\{q(k) + score(j, k)\}\right)$$

$$q(j) \leftarrow q(j) + \log(1 + \exp\{score(j, j)\ \}\ )$$

end for

Read out the posterior marginals as derivatives:
for all i=0,..., N-1 **do**
$p(i, i) \leftarrow \exp\{v(i) + q(i) + score(i, i) - 2\log(\exp(score(i, i) + 1)) - \log Z\}$
end for
for all i<j **do**
$p(i, j) \leftarrow \exp(v(i) + q(j) + score(i, j) - \log Z)$
end for
for all i=1,..., N-1 **do**
$p_\epsilon(i-1, i) \leftarrow \exp(v(i-1) + q(i) + score_\epsilon(i-1, i) - \log Z)$
end for
$\frac{\partial \log Z}{\partial score(i,j)} = p(i, j)$, $\frac{\partial \log Z}{\partial score_\epsilon(i-1,i)} = p_\epsilon(i-1, i)$

3.3 Inference

We use dynamic programming (Viterbi) to infer the most likely interval sequence for every *eventType* independently. This process is shown in Algorithm 2.

When processing longer audio recordings, our system transcribes audio segment by segment. Because the system is designed to ignore all events that extends outside a segment, these audio segments need overlap with each other. In this work, segments are 10s long and the overlap is 5s. In order to properly handle notes near the boundary of a segment, we modify the algorithm such that it is forced to take the result from the overlapping portions into account. Segment-wise decoding is performed backwards and backtracking begins from the position immediately after the last event of the same *eventType* in the overlapping region from the most recently processed segment (if any).

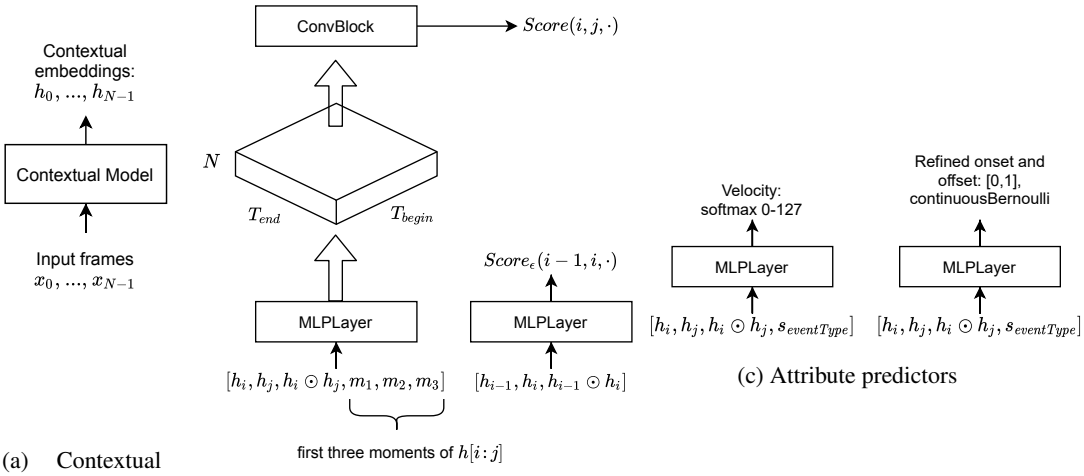
After intervals (events) are extracted, feature vectors of certain intervals are used to predict attributes, namely, velocity and refined onset/offset times. These attributes are then combined with the intervals to form an MIDI event tuple $(begin, end, pitch, velocity)$ for the final output.

Algorithm 2 Viterbi (MAP) decoding of a specific event type within an audio segment

Input: function $score(i, j)$, function $score_\epsilon(i-1, i)$, backtracking starting position t
Output: a set of intervals $\mathcal{Y}$
$v(N-1) \leftarrow \max(score(N-1, N-1), 0)$
for all $j \in N-2, \ldots, 0$ **do**
$v(j) \leftarrow \max \begin{cases} v(j+1) + score_\epsilon(j, j+1) \text{ - } \textit{skip if inactive} \\ \max_{k>j}\{v(k) + score(j, k) \text{ - } \textit{if an interval} \end{cases}$
$v(j) \leftarrow v(j) + \max(score(j, j), 0)$ - *single frame case*
end for
Perform backtracking starting from position t to get $\mathcal{Y}$

3.4 Model Architectures

In this section, we present details of the three neural architectures of the proposed approach: 1) contextual model, 2) score models, and 3) attribute predictors. As a recap, we first apply a contextual model to transform the input audio frames into a sequence of contextual embeddings. These contextual embeddings are then used by the score models to form feature vectors for intervals and compute score values $score(\cdot)$ and $score_\epsilon(\cdot)$), in a similar fashion as Cross and Huang [2016], Kitaev and Klein [2018], Liu et al. [2016]. After events are extracted, contextual embeddings are also used by the attribute predictors to predict attributes including the velocity and refined onset/offset times associated with each event.



(a) Contextual model

(b) Score models

(c) Attribute predictors

Figure 2: Model Architecture

3.4.1 Contextual Model

We first apply a windowing function to each frame in the input audio and convert them to a log-mel spectrogram. We use a sampling rate of 44100 Hz, a hop size of 1024 samples, and a frame size of 4096 samples. Following Hawthorne et al. [2018], Kong et al. [2020], we use a 229-band log-mel spectrogram with a frequency range from 30 hz to 8000hz. Then we apply four conv blocks with output sizes 48, 64, 92, 128. Each conv block contains two 2-d convolutional layers, each followed by batch normalization and GELU activation function [Hendrycks and Gimpel, 2016]. At the end of each block, the output is pooled by 2 along the frequency dimension using average pooling. The channel and frequency dimensions are flattened together and fed into a two-layer GRU with hidden size 256. This contextual model is largely the same as the one used in Hawthorne et al. [2018] and Kong et al. [2020], however we only utilize one instance of the model, rather than stacking it in parallel for different frame-level prediction targets.

3.4.2 $Score(i, j, \cdot)$ and $Score_\epsilon(i-1, i, \cdot)$

As shown in Fig. 2b, the features used for scoring an interval $[i, j]$ include contextual embeddings at two endpoints, i.e., $\mathbf{h}_i$, $\mathbf{h}_j$, their elementwise multiplication $\mathbf{h}_i \odot \mathbf{h}_j$, and the first three moments for the contextual embeddings within the interval $[i, j]$. These features are concatenated and fed into a three-layer feed-forward network (*MLPLayer*), to get a raw intervalic score tenor with shape $T_{end} \times T_{begin} \times N$, where the three dimensions are the ending position of an interval, the beginning position of an interval, and *eventType* channels, respectively. The output size of this network is equal to the number of event types and the hidden size is equal to four times the output size. On top of the raw score tensor, we apply a simple conv block with two convolutional layers with kernel size 3, with the hope that the block can aggregate neighboring information in the space of interval space in order to increase the expressivity. We also experimented with scaling the intervalic score by the length of the interval.

For the 1-length inactivity score, $score_\epsilon(i-1, i)$, only contextual embeddings of two consecutive frames are used, which are then similarly fed into a three-layer feed-forward network with the output size equal to the number of event types.

3.4.3 Attribute Predictor

As shown in Fig. 2c, for predicting attributes associated with each event, we use features of endpoints concatenated with an *eventType* embedding vector (for fully specifying the event). These features are fed into separate three-layer feed-forward networks. The velocity prediction network has a hidden sizes of 512, and produces a127-dimensional vector.The refined onset and offset time prediction network has hidden sizes 512 and 128, and produces a 2-dimensional vector (1 dimension for the onset, and 1 for the offset).

4 Experiments

4.1 Dataset

We conduct our experiments using the MAESTRO v2.0.0 dataset [Hawthorne et al., 2019], which contains around 200 hours of MIDI-synchronized (3ms precision) virtuoso piano performance recordings. The recordings were collected across several years of the *International Piano-e-Competition*, and were recorded on *Yamaha Disklavier pianos*. All recordings are sampled at 44.1 kHz, except for files from the 2017 and 2018 competitions, which are sampled at 48 kHz. We only downsample audio files from these two years (to 44.1 kHz). The durations of notes are extended whenever the sustain pedal is active, which is a common practice for piano transcription training and evaluation. We follow this convention so that our system is comparable to other works.

4.2 Training

We use a batch size of 12 and Adabelief [Zhuang et al., 2020] optimizer with a weight decay of 1e-4. We use oneCycle [Smith and Topin, 2019] learning rate scheduler with maximum learning rate set to 6e-4 for 120k iterations. The learning rate is increased gradually for 30% of iterations and then gradually annealed to zero. We automatically determine the value for gradient clipping, which is a strategy similar to Seetharaman et al. [2020], by using the 0.8 quantile of the recent 10k gradient norm history. We apply dropout with rate 0.1 on the attribute predictors and the score model.

4.3 Evaluation Metrics

We follow the standard piano transcription evaluation procedure, validating note predictions using various additive conditions, as well as considering the frame-level polyphonic pitch overlap that such note predictions yield. The most basic metric considers a note prediction correct if the estimated pitch is within half a semitone and the estimated onset with 50 ms of the respective ground-truth values. The next incremental metric additionally requires that a note prediction have an offset prediction within the greater tolerance among 50 ms of the ground-truth or 20% of the note duration. The final incremental metric additionally requires a correct velocity prediction, defined in Hawthorne et al. [2018]. The velocity is matched with a tolerance of 0.1 (normalized velocity).

While irrelevant for our method, a frame-level pitch activation score derived from the note predictions and averaged across all frames of a track, is also offered for comparison. For the frame-level metrics, pitch activations are quantized to a time precision of 1ms [1]

Each of these scores are averaged across all pieces within the test set. We also compute similar metrics, minus the velocity variation, for the sustain pedal activity, as in Kong et al. [2020].

4.4 Main Results

	Frame			Note Onset			Note w/ Offset			Note w/ Offset & Vel.		
Method	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Hawthorne et al. [2019]	86.83	89.21	87.80	97.88	92.26	94.93	82.09	77.44	79.65	78.37	73.94	76.05
Kong et al. [2020]	90.11	**90.44**	90.17	98.16	**95.46**	**96.77**	85.65	83.32	84.45	84.18	81.92	83.02
Proposed no postConv.	93.78	88.22	90.81	98.73	93.71	96.12	90.41	85.87	88.04	89.35	84.89	87.02
Proposed w/ len scaling	93.73	88.46	**90.92**	98.81	93.97	96.29	90.76	**86.36**	88.47	89.76	**85.43**	87.51
Proposed	**94.06**	87.96	90.81	**98.97**	93.86	96.31	**90.92**	86.28	**88.50**	**89.91**	85.35	**87.53**

Table 2: Piano transcription note results for the proposed methods and various related works.

	Frame			Onset			Onset & Offset		
Method	P	R	F_1	P	R	F_1	P	R	F_1
Kong et al. [2020]	94.15	**94.29**	**94.11**	77.43	**78.19**	**77.71**	73.56	**74.21**	**73.81**
Proposed no postConv.	**95.05**	86.46	89.93	80.54	72.16	75.82	76.83	69.00	72.44
Proposed w/ len scaling	95.02	87.09	90.32	80.80	72.92	76.39	77.27	69.86	73.12
Proposed	94.97	86.04	89.69	**81.09**	72.42	76.23	**77.64**	69.46	73.06

Table 3: Sustain pedal detection results for the proposed methods and various related works.

We compare the proposed system to the state-of-the-art methods for piano transcription on the Maestro v2.0.0 test split in Table 2 and 3. One thing to mention is that the reported systems only use one convolution-RNN block as the contextual model, while Hawthorne et al. [2019] and Kong et al. [2020] stacked several blocks of similar size. Our proposed system achieves a note with offset F1 of 88.5% and note with Offset and velocity F1 of 87.53%.

Kong et al. [2020] has the highest note onset F1, as it directly optimized for onset predictions. For the pedal results, our method slightly underperforms Kong et al. [2020] on the event level metrics, which uses a separate branch specifically for predicting pedals.

We note that our system is generally higher in precision, which suggests that we may further optimize for F1 measures by tuning the threshold of predictions. This can be done by subtracting the threshold on posterior marginals computed from the forward-backward algorithm and then using the posterior marginals as the score input to the Viterbi algorithm to obtain a solution under non-overlapping constraint.

We also note that our system transcribes piano pieces faster than baseline systems. This is mainly due to reduced model complexity and the lack of post-processing steps involved within conditional branches in the other methods.

4.5 Computational time of layers with quadratic time complexity

The proposed model has a $\mathcal{O}(N^2|E|)$ time complexity, where N is the length of the input sequence, i.e., the number of frames, and $|E|$ is the number of *eventType*(s). Also, computations of the semi-CRF layer involves sequential computation, which may be traditionally considered slow to be practically used for a long sequence if not restricted further. Here we benchmark the computational time of those components with quadratic time complexity for a reasonable length of input sequence used for the music transcription task, shown in Table 4. With 1024 hopsize under sampling rate 44100, 400, 800, 1600 corresponds to an audio segment of 9.29s, 18.58s, and 37.15s, respectively. For these lengths, we see that the computational time does not contribute much compared with the computational time of remaining parts of the model. We implemented algorithms in *pytorch* and we believe that further speedup can be achieved with a native C++/CUDA implementation.

[1] This can be more precisely thought of as activation-level metrics for each pitch.

nFrames	400	800	1600
Forward only	0.05 (0.2)	0.09 (0.80)	0.23(3.08)
ForwardBackward	0.08 (0.58)	0.12 (2.32)	0.27 (7.12)
Viterbi	0.23 (0.16)	0.27 (0.36)	0.54 (0.88)
Pairwise scores	0.06 (0.84)	0.25 (3.14)	0.99 (11.76)

Table 4: Computational time (s) of quadratic layers on a computer with Intel(R) Core(TM) i7-7800X CPU @ 3.50GHz and Nvidia GTX 1080TI. $|E| = 90$. Running time purely on CPU are shown in parenthesis.

5 Conclusion

In this work, we propose an piano transcription system that is designed to directly predict note events. Our method uses semi-Markov Conditional Random Fields as the output layer where a set of non-overlapping events for a key/pedal is directly used the prediction target. By doing so, we eliminate the need to rely on disjoint frame-level estimates for different stages of a note event. We believe that this simple, fast, and well-performing approach can be extensible to other similar tasks like polyphonic sound event detection.

References

Ravneet Arora, Chen-Tse Tsai, Ketevan Tsereteli, Prabhanjan Kambadur, and Yi Yang. A semi-Markov structured support vector machine model for high-precision named entity recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5862–5866, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1587. URL https://www.aclweb.org/anthology/P19-1587.

Juan Pablo Bello, Laurent Daudet, and Mark B Sandler. Automatic piano transcription using frequency and time-domain information. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(6):2242–2251, 2006.

Emmanouil Benetos, Simon Dixon, Zhiyao Duan, and Sebastian Ewert. Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, 36(1):20–30, 2018.

Sebastian Böck and Markus Schedl. Polyphonic piano note transcription with recurrent neural networks. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 121–124. IEEE, 2012.

Tian Cheng, Matthias Mauch, Emmanouil Benetos, Simon Dixon, et al. An attack/decay model for piano transcription. ISMIR, 2016.

Andrea Cogliati, Zhiyao Duan, and Brendt Wohlberg. Context-dependent piano music transcription with convolutional sparse coding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(12):2218–2230, 2016.

Andrea Cogliati, Zhiyao Duan, and Brendt Wohlberg. Piano transcription with convolutional sparse lateral inhibition. *IEEE Signal Processing Letters*, 24(4):392–396, 2017.

James Cross and Liang Huang. Span-based constituency parsing with a structure-label system and provably optimal dynamic oracles. In *Proceedings of EMNLP*, 2016.

Valentin Emiya, Roland Badeau, and Bertrand David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1643–1654, 2009.

Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse Engel, Sageev Oore, and Douglas Eck. Onsets and frames: Dual-objective piano transcription. In *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, 2018*, 2018. URL https://arxiv.org/abs/1710.11153.

Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. Enabling factorized piano music modeling and generation with the MAESTRO dataset. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=r1lYRjC9F7.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Hirokazu Kameoka, Takuya Nishimoto, and Shigeki Sagayama. A multipitch analyzer based on harmonic temporal structured clustering. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(3):982–994, 2007.

Rainer Kelz, Sebastian Böck, and Gerhard Widmer. Deep polyphonic adsr piano note transcription. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 246–250. IEEE, 2019.

Apostolos Kemos, Heike Adel, and Hinrich Schütze. Neural semi-markov conditional random fields for robust character-based part-of-speech tagging. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2736–2743, 2019.

Jong Wook Kim and Juan Pablo Bello. Adversarial learning for improved onsets and frames music transcription. In *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019*, pages 670–677, 2019.

Nikita Kitaev and Dan Klein. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, 2018.

Anssi Klapuri, Tuomas Virtanen, and Jan-Markus Holm. Robust multipitch estimation for the analysis and manipulation of polyphonic musical signals. In *Proc. COST-G6 Conference on Digital Audio Effects*, pages 233–236, 2000.

Lingpeng Kong, Chris Dyer, and Noah A Smith. Segmental recurrent neural networks. *arXiv preprint arXiv:1511.06018*, 2015.

Qiuqiang Kong, Bochen Li, Xuchen Song, Yuan Wan, and Yuxuan Wang. High-resolution piano transcription with pedals by regressing onsets and offsets times. *arXiv preprint arXiv:2010.01815*, 2020.

Taegyun Kwon, Dasaem Jeong, and Juhan Nam. Polyphonic piano transcription using autoregressive multi-state note model. In *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, 2018*, 2020. URL `https://arxiv.org/abs/1710.11153`.

Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, and Ting Liu. Exploring segment representations for neural segmentation models. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2880–2886, 2016.

Kristen Masada and Razvan C Bunescu. Chord recognition in symbolic music using semi-markov conditional random fields. In *ISMIR*, pages 272–278, 2017.

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)*, pages 101–105. IEEE, 2018.

Ken O'Hanlon and Mark D Plumbley. Polyphonic piano transcription using non-negative matrix factorisation with group sparsity. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3112–3116. IEEE, 2014.

Graham E Poliner and Daniel PW Ellis. A discriminative model for polyphonic piano transcription. *EURASIP Journal on Advances in Signal Processing*, 2007:1–9, 2006.

Christopher Raphael. Automatic transcription of piano music. In *ISMIR*, 2002.

Miguel A Román, Antonio Pertusa, and Jorge Calvo-Zaragoza. An end-to-end framework for audio-to-score music transcription on monophonic excerpts. In *ISMIR*, pages 34–41, 2018.

Miguel A Román, Antonio Pertusa, and Jorge Calvo-Zaragoza. A holistic approach to polyphonic music transcription with neural networks. In *ISMIR*, 2019.

Sunita Sarawagi and William W Cohen. Semi-markov conditional random fields for information extraction. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, pages 1185–1192, 2004.

Prem Seetharaman, Gordon Wichern, Bryan Pardo, and Jonathan Le Roux. Autoclip: Adaptive gradient clipping for source separation networks. In *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2020.

Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An end-to-end neural network for polyphonic piano music transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):927–939, 2016.

Paris Smaragdis and Judith C Brown. Non-negative matrix factorization for polyphonic music transcription. In *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No. 03TH8684)*, pages 177–180. IEEE, 2003.

Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, volume 11006, page 1100612. International Society for Optics and Photonics, 2019.

Emmanuel Vincent, Nancy Bertin, and Roland Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):528–537, 2009.

Felix Weninger, Christian Kirst, Björn Schuller, and Hans-Joachim Bungartz. A discriminative approach to polyphonic piano note transcription using supervised non-negative matrix factorization. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6–10. IEEE, 2013.

Zhixiu Ye and Zhen-Hua Ling. Hybrid semi-markov crf for neural sequence labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 235–240, 2018.

Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *Conference on Neural Information Processing Systems*, 2020.

Jingwei Zhuo, Yong Cao, Jun Zhu, Bo Zhang, and Zaiqing Nie. Segment-level sequence modeling using gated recursive semi-markov conditional random fields. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1413–1423, 2016.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Details to reproduce are mentioned in the main text. Code will be released upon publication
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] Too expensive to train so many times
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]