Transactions of the International Society for Music Information Retrieval

Anonymous (2026). Structured Prediction for Piano Fingering with an Interval-Conditioned CRF, *Transactions of the International Society for Music Information Retrieval*, V(N), pp. xx–xx, DOI: https://doi.org/xx.xxxx/xxxx.xx

RESEARCH ARTICLE

Structured Prediction for Piano Fingering with an Interval-Conditioned CRF

Anonymous Author*

Abstract

Automatic piano fingering is a structured prediction problem: the same notes admit many valid fingerings, expert annotators disagree, and a good choice at one note often depends on notes several positions away. We present KeyGenius, a compact (4.6M-parameter) Transformer encoder paired with a linear-chain CRF whose transitions are conditioned on local interval context, pretrained on a large weakly-labelled corpus and fine-tuned on the PIG dataset. On PIG's canonical multi-annotator test set, the model is competitive with substantially larger ThumbSet-pretrained autoregressive baselines — ahead on soft-match (M_{soft} 87.9 vs 86.8) and slightly behind the best on the primary metric (M_{gen} 65.8 vs 66.8) — at a small fraction of their parameter count. We further report three results that qualify common assumptions in the task: structured CRF decoding does not significantly improve aggregate annotator agreement over independent per-note prediction, though it more than halves runs of a single finger held across moving pitch; hand-authored ergonomic decoding constraints reduce rather than improve agreement with expert annotators; and pairwise inter-annotator agreement is a weak "ceiling" for a model predicting consensus. Finally, we describe a deployed system in active use by piano teachers. Code is openly available.

Keywords: piano fingering, structured prediction, conditional random fields, symbolic music

1. Introduction

Choosing how to finger a passage is one of the first real interpretive decisions a pianist makes, and one of the easiest to get wrong. A fingering that feels adequate under the hand in isolation can strand the hand a bar later, force an awkward crossing, or quietly cap the tempo a passage can reach. Poor fingering learned early is not merely inefficient: because playing is a motor skill consolidated through repetition, an unhelpful fingering practised until automatic is far harder to unlearn than to have avoided, so the cost of a bad choice compounds over the time a student spends with a piece.

The difficulty is that good fingering is neither unique nor local. The same notes admit many defensible fingerings, and expert annotators routinely disagree; the "right" answer depends on tempo, phrasing, dynamics, and the interpretive effect a player is after. A repeated note may be played with the same finger, or deliberately with a changing finger to shape articulation and give the line depth – both correct, for different musical ends. And the choice at any one note is rarely justified by that note alone: a comfortable fingering often requires planning several notes ahead, setting up a thumb tuck or a hand shift before the moment it is needed. This forward-planning is exactly what beginners find hardest and what only becomes automatic with experience. It is also what makes the task a natural fit for structured, sequence-level modelling rather than independent per-note classification.

Prior work has largely approached automatic fingering with hidden Markov models (Nakamura et al., 2020) and, more recently, a conditional random field (Randolph et al., 2023), evaluated on the PIG dataset (Nakamura et al., 2020) using match rates against multiple human annotators. We revisit the task with a compact neural sequence model and, in the process, examine several assumptions the field has carried – in particular whether hand-authored ergonomic constraints help, and what the multi-annotator "ceiling" actually measures.

Contributions.

- We present a compact (4.6M-parameter) Transformer encoder with an interval-conditioned linear-chain CRF for automatic piano fingering, trained by ThumbSet pretraining followed by PIG

*Anonymous Affiliation

fine-tuning. The CRF does not significantly improve aggregate annotator agreement over independent per-note prediction (Wilcoxon $W = 210.0$, $p = 0.65$), though it more than halves runs of a single finger held across moving pitch (61 → 28). The component that does help agreement is a hard anatomical chord constraint at decode time, which is the best configuration on all three metrics while guaranteeing playable output.

- On the canonical PIG test set (30 pieces, 10 each Bach/Mozart/ Chopin), the model is competitive with substantially larger ThumbSet-pretrained autoregressive baselines (Ramoneda et al., 2022) at 4.6M parameters — ahead on M_{soft} (87.9 vs 86.8), slightly behind the best on M_{gen} (65.8 vs 66.8).
- We show ThumbSet pretraining gives a clean, leakage-free gain of +3.3 M_{gen} and +4.8 M_{soft} on the canonical split.
- We report a negative result: hand-authored ergonomic decoding constraints, encoding standard pedagogical heuristics, do not improve (and slightly reduce) agreement with expert annotators; the interval-conditioned CRF appears to already capture this structure from data. A hard anatomical chord rule, by contrast, fixes rare real playability violations at no measurable cost.
- We deploy the system as a public web application at keygenius.app — a rare demonstration of an MIR fingering model in real use. The system is in beta: OMR-derived input can be faulty, a failure mode we characterise rather than elide.

2. Related Work

Early automatic fingering methods were rule- or cost-based, encoding ergonomic constraints by hand (Parncutt et al., 1997). Because appropriate parameter values for such cost models are difficult to determine, later work turned to data-driven formulations. Yonebayashi et al. (2007) cast fingering as HMM decoding, and Nakamura et al. (2020) released the PIG dataset alongside a systematic study of first- through third-order and chord HMMs, introducing the M_{gen}, M_{high}, and M_{soft} metrics we adopt.

Neural approaches followed. Guan et al. (2021) applied a bidirectional LSTM with an input representation capturing relations between surrounding notes, and proposed a playability measure alongside match-rate evaluation. Guan et al. (2022) report improvements over the third-order HMM with a pitch-difference matching model. Most relevant to our setting, Ramoneda et al. (2022) released ThumbSet, a large crowdsourced but non-expert corpus, and trained autoregressive LSTM and graph-neural models on it, establishing the strongest published results on the canonical PIG test set. A separate line treats fingering as sequential decision-making: recent work formulates polyphonic fingering as model-free reinforcement learning with elite episode replay (Iman and Ahn, 2025). Randolph et al. (2023) model fingering with a first-order linear-chain CRF, and note limitations of match-rate metrics when a piece admits multiple score representations.

Our work differs in three respects. Architecturally, we pair a Transformer encoder with a CRF whose transition scores are conditioned on local interval context, rather than a fixed transition table, and decode exactly with Viterbi. Methodologically, we use the multi-annotator structure of PIG to characterise the human agreement ceiling, and we *test* rather than assume that hand-authored ergonomic constraints improve agreement – reporting a negative result where they do not. Finally, we report a deployed system in use by piano teachers, in addition to benchmark numbers.

3. Data

3.1 PIG

We use the PIG dataset (Nakamura et al., 2020), the standard benchmark for piano fingering. It comprises 150 pieces (opening fragments) drawn from 24 composers across the standard repertoire, with per-note finger labels (1–5) provided by experienced pianists – annotators who had graduated from a music college or played piano for more than twenty years. In total the dataset contains 309 annotation files (some pieces carry fingerings from several pianists) and roughly 100,000 labelled notes. Each file is a tab-separated list of notes with onset and offset times, pitch, and finger; the timing derives from score-editor synthesis rather than recorded human performance.

Following (Nakamura et al., 2020; Ramoneda et al., 2022), we use the canonical test set of 30 pieces – 10 each by Bach, Mozart, and Chopin, representing the Baroque, Classical, and Romantic periods – each annotated by four (Bach), six (Mozart), or five (Chopin) pianists. These multi-annotator pieces make inter-annotator agreement measurable, which we use both to evaluate the model and to characterise the human ceiling. All remaining pieces form the training pool.

3.2 ThumbSet

For pretraining we use ThumbSet (Ramoneda et al., 2022), a substantially larger but weakly and sparsely labelled corpus of user-uploaded scores. Roughly half of its notes carry a finger label; the rest are unlabelled. We use ThumbSet only to initialise the encoder and emission head (see Section 4); it is never used for evaluation.

3.3 Split and Evaluation Protocol

The 30 canonical test pieces are held out of training entirely, in both the pretraining and fine-tuning phases. From the remaining ~120 pieces we carve a validation set (10% of pieces) used only for early stopping and checkpoint selection; the test set is never con-

sulted during training or model selection. Every piece is assigned to exactly one split at the piece level (a piece's multiple annotator files are never divided across splits), and a run-time assertion checks the actually-loaded training set against the 30 test pieces at the start of every fine-tuning run, rather than trusting a static ID list, so any mismatch between the nominal split and what is read from disk is caught immediately rather than silently inflating results. For pretraining, we additionally exclude 1,366 ThumbSet windows that are alternate transcriptions of the same underlying compositions as four of the 30 canonical test pieces (178,737 windows before exclusion, 177,371 after) — a composition-level overlap that a file-identity check alone would miss. We quantify the effect of this exclusion in Section 7. This protocol follows the canonical test set used by prior work (Nakamura et al., 2020; Ramoneda et al., 2022), though the comparison is not perfectly controlled; see Section 7. One limitation, noted in Section 7: the validation set does double duty for model selection, and we report on the canonical test set without a further held-out split beyond it.

4. Method

4.1 Task Formulation

We formulate piano fingering prediction as per-note sequence labeling. Given a symbolic score, each note is assigned a finger label 1–5, predicted independently for each hand: a piece's notes are split by hand into two note sequences, each decoded as its own linear-chain CRF, so a finger-transition score is only ever computed within one hand's sequence and never across hands. A single set of model weights is shared across both hands; hand identity is provided as an input feature rather than via separate parameters. Within a chord (notes sharing an onset), notes are ordered by ascending pitch before being fed to the model, so the sequence the encoder and CRF observe is always low-to-high within simultaneities.

4.2 Feature Representation

Each note is represented by 14 continuous features, a learned pitch-class embedding, and a learned hand embedding, concatenated into a 26-dimensional input. The continuous features fall into four groups: (i) the note's own pitch (normalized to the 88-key range) and log-scaled duration; (ii) timing and interval context relative to both neighbors — log inter-onset intervals, and both the signed and unsigned pitch interval to the previous and next note; (iii) binary chord-membership flags indicating whether the previous or next note shares this note's onset; and (iv) coarse physical/ergonomic indicators — whether this or the preceding note is a black key, and whether the melodic move to the previous note is stepwise (≤ 2 semitones) or a leap (≥ 5 semitones). Pitch class (0–11) is embedded into 8 dimensions and hand (left/right) into 4, giving a total input width of $14 + 8 + 4 = 26$.

4.3 Model Architecture

The 26-dimensional per-note input is linearly projected to $d_{\text{model}} = 256$, combined with a fixed sinusoidal positional encoding, and passed through a multi-scale local-context convolution block — three parallel `Conv1d` branches (kernel widths 3, 5, 7) with GELU activations, concatenated and projected back to d_{model} with a residual connection — before entering an 8-layer pre-norm Transformer encoder (8 attention heads, feed-forward width 512, dropout 0.2, GELU activation). A linear head maps the encoder output to 5 emission scores per note, which feed a linear-chain CRF (Section 4.4). The convolutional block precedes the Transformer specifically to give the model a short local receptive field for scale and arpeggio patterns before global self-attention mixes the full sequence. The complete model has 4.62M parameters.

4.4 Interval-Conditioned CRF

The CRF's base transition matrix is a freely learned 5×5 parameter, uniform-initialized in $[-0.5, 0.5]$ and trained end-to-end via CRF negative log-likelihood; no anatomical or ergonomic prior is built into its initialization or loss, so whatever transition structure it learns comes entirely from PIG's annotated sequences. Transition scores are additionally conditioned, at every step, on the signed pitch interval and chord-membership flag relating each pair of consecutive notes: a small two-layer MLP (hidden width 16) maps this 2-dimensional context to a per-step 5×5 adjustment added to the base matrix. The MLP's output layer is zero-initialized, so the conditioned model is architecturally identical to a plain static-transition CRF at initialization and can only diverge from it if doing so reduces training loss — making the conditioned CRF a strict superset of the earlier static design rather than a different starting point. Learned start- and end-of-sequence bias vectors complete the model. Decoding is exact Viterbi; because the transition matrix is now a function of local context, decoding recomputes the relevant 5×5 matrix at each step rather than reusing one fixed matrix throughout.

4.5 Physical Decode Constraints

We evaluate two hand-authored constraints applied at decode time, neither learned from data. The first is a hard anatomical rule: notes within a chord must receive distinct fingers ordered monotonically with pitch, since fingers cannot cross under a simultaneously sounded chord. The second is a set of three soft penalties nudging the decoder toward wider finger spreads on wider intervals, away from same-finger reuse across small melodic steps, and away from placing the thumb on a black key in stepwise motion.

On the 30 canonical test pieces, unconstrained decoding violates the hard chord rule on 22 of 1,674

chords (1.3%), affecting 94 notes. Enforcing the rule removes these violations and is simultaneously the best-performing configuration on all three metrics (M_{gen} 65.8, M_{soft} 87.9, M_{high} 71.9), so playability is obtained at no cost — and a small benefit. We therefore retain it.

The soft penalties behave differently. They leave M_{gen} and M_{high} essentially unchanged (65.7 and 71.8) while giving back the M_{soft} gain (87.5 vs 87.9). We interpret this in light of the interval-conditioned CRF: the learned transitions already capture, from data, the interval-dependent finger-spread structure that the soft rules encode by hand, so the hand-authored version adds no signal and narrows the set of fingerings the model will agree with. We report this as a negative result and exclude the soft penalties from the final model. The contrast is informative: a constraint expressing a hard anatomical fact helps, while constraints expressing softer pedagogical preferences do not.

4.6 Training Procedure

Training proceeds in two phases, both using the architecture and loss above. **Pretraining** uses ThumbSet (Section 3) with a masked focal loss ($\gamma = 2.0$) computed only over notes carrying a real finger label; unlabeled notes still pass through the encoder to provide context but contribute no loss, and the CRF's transition parameters are left untouched in this phase, since ThumbSet's sparse labeling cannot support coherent transition learning. We use AdamW with learning rate 3×10^{-4} and batch size 256 — substantially larger than the fine-tuning batch size, since ThumbSet is roughly 600× the size of PIG. Pretraining early-stops on ThumbSet's held-out labeled-match rate (patience 8) against a ceiling of 100 epochs. **Fine-tuning** trains the full model — encoder and CRF jointly — on PIG under the canonical split (Section 3), initialized from the pretrained checkpoint. The loss is $3.0\,\mathscr{L}_{\text{CRF}} + 1.0\,\mathscr{L}_{\text{focal}}$ ($\gamma = 2.0$); the CRF term is weighted more heavily to push the model toward learning finger *transitions* rather than only marginal finger frequency. We use AdamW (learning rate 1×10^{-4}, weight decay 1×10^{-4}), gradient clipping at 1.0, a 5-epoch linear warmup followed by cosine decay, and early stopping (patience 12) against a ceiling of 80 epochs. Pitch-transposition augmentation (a uniformly sampled shift in ±6 semitones, training split only) follows the standard assumption that fingering choice is approximately invariant to transposition. Both phases split at the piece level, never at the file or window level, and fine-tuning holds the 30 canonical test pieces out entirely. All experiments use a fixed seed (1337). The reported checkpoint reached its best validation match rate (75.09%) at epoch 10 of fine-tuning, stopping at epoch 23; fine-tuning wall-clock time was 4 minutes 16 seconds on a single NVIDIA A100-SXM4-40GB GPU.

5. Experiments

5.1 Metrics

Following Nakamura et al. (2020), we report three note-level match-rate statistics against PIG's multi-annotator ground truth, plus a human agreement ceiling constructed the same way over annotator pairs. Let $\mathscr{A}_p$ be the set of annotators for piece p, N the number of notes, $\hat{y}(n)$ the model's predicted finger for note n, and $y_a(n)$ annotator a's label for note n.

M_{gen} (general match rate, the primary metric) is the model's match rate against each individual annotator, averaged over annotators:

$$M_{\text{gen}} = \frac{1}{|\mathscr{A}_p|} \sum_{a \in \mathscr{A}_p} \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}\big[\hat{y}(n) = y_a(n)\big].$$

M_{soft} counts a note correct if the prediction matches *any* annotator's label for that note (a set-membership match against the union across annotators); since agreeing with any one of several annotators is easier than agreeing with a fixed one, $M_{\text{soft}} \geq M_{\text{gen}}$ always. M_{high} is the match rate against whichever single annotator the model agrees with *most* per piece — a maximum over annotators, not an average, and therefore upward-biased relative to M_{gen}.

The human agreement ceiling is constructed identically, substituting one annotator's labels for the model's predictions: mean pairwise agreement across all annotator pairs is the ceiling analog of M_{gen}, and the best-agreeing pair per piece is the ceiling analog of M_{high}. Because one is a mean and the other a maximum, M_{high} should be compared only against the best-agreeing-pair ceiling, not the mean-pairwise ceiling.

5.2 Main Results

Table 1 compares KeyGenius against published results on PIG's canonical test set, and Figure 1 shows the per-piece relationship between model agreement and human agreement.

5.3 Effect of Pretraining

To isolate the contribution of ThumbSet pretraining, we train the same architecture on PIG alone under the identical split (Table 2).

5.4 Decoding Ablation

Table 3 isolates the effect of each decoding component: independent per-note prediction, the learned CRF, and the two families of hand-authored constraint described in Section 4.5.

6. Deployment

KeyGenius is deployed as a public web application at keygenius.app. The deployed pipeline accepts MusicXML, MIDI, or a photographed or scanned score image. Image input is first converted to MusicXML via optical music recognition: `homr`, a transformer-based

Model	M_{gen}	M_{soft}	M_{high}
HMM (1st order) (Nakamura et al., 2020)	61.8	81.1	67.7
HMM (3rd order) (Nakamura et al., 2020)	63.6	83.1	69.4
LSTM (Ramoneda et al., 2022)	62.7	82.7	67.3
ArLSTM (ThumbSet) (Ramoneda et al., 2022)	65.3	85.5	71.7
ArGNN (ThumbSet) (Ramoneda et al., 2022)	**66.8**	86.8	**72.6**
KeyGenius (ours)	65.8	**87.9**	71.9
Human ceiling (mean pairwise)	71.4	–	79.1

Table 1: Results on PIG's canonical multi-annotator test pieces (both hands), metric names following (Nakamura et al., 2020). Baseline figures are as reported by (Ramoneda et al., 2022). We caution that the comparison is close but not perfectly controlled: their protocol uses a validation split of their own construction and a different implementation of the evaluation metrics, so small differences should not be over-interpreted. KeyGenius, at 4.6M parameters, is competitive with the larger ThumbSet-pretrained autoregressive models — ahead on M_{soft}, slightly behind the best (ArGNN) on M_{gen} and M_{high}. All models remain below the human agreement ceiling on M_{gen}.

Training	M_{gen}	M_{soft}	M_{high}
PIG only	62.5	83.1	66.7
ThumbSet → PIG	**65.8**	**87.9**	**71.9**

Table 2: ThumbSet pretraining gives a clean, leakage-free gain on the canonical test set (+3.3 M_{gen}, +4.8 M_{soft}). Both rows use the same architecture and split. Notably, our PIG-only model (62.5) exceeds the PIG-only autoregressive GNN of (Ramoneda et al., 2022) (60.5, their Table 3); the baselines' overall edge comes from their ThumbSet and autoregressive recipe rather than architecture alone.

OMR engine, is tried first; `oemer`, an older U-Net-plus-SVM system, serves as a fallback; and Audiveris is offered as a manual option for clean, high-resolution (≥300 dpi) scans, requiring the user to run it separately and supply the resulting MusicXML. Model training never uses OMR output — training is always on symbolic data directly; OMR is purely an inference-time convenience for users without a symbolic source file. Symbolic input, from any source, is parsed with `music21` into per-hand note sequences, converted to the feature representation of Section 4, and passed through the trained model and CRF decoder to obtain a predicted finger and a confidence score — the CRF's posterior marginal probability for the chosen finger — for every note. Predicted fingerings are written back into the output MusicXML as genuine `music21.articulations.Fingering` objects attached to their notes, rather than as pixel coordinates overlaid on a rendered image, so downstream notation software places them correctly by construction. Notes whose confidence falls below a threshold (0.6 by default) are flagged for manual review rather than presented with unwarranted certainty. We note a limitation stated in the OMR module's own documentation: recognition quality degrades on dense polyphonic piano writing, and the system is accordingly labeled beta for image input.

7. Discussion and Limitations

Our results place KeyGenius competitive with substantially larger ThumbSet-pretrained baselines on the canonical PIG test set — ahead on M_{soft} but slightly behind the best model on M_{gen} — at a small fraction of their parameter count. Several caveats bound how that should be read.

The margin is small and against reported numbers. On the primary metric we sit one point *below* the best published model (M_{gen} 65.8 vs 66.8), and the baseline figures are those reported in prior work (Ramoneda et al., 2022) rather than re-run under our exact pipeline. We therefore claim competitiveness, not improvement; a fully controlled comparison would rerun the baseline models through our evaluation code, which we leave to future work.

Additional training data. Our full model is pretrained on ThumbSet before fine-tuning on PIG, whereas the baselines train on PIG alone. The strictly matched, PIG-only variant of our model (Table 2) sits slightly *below* the baselines, and the gain over it comes from pretraining rather than from the architecture in isolation. We regard this as an honest strength of the approach – weak, abundant labels help – rather than evidence that the model is intrinsically stronger.

Composition-level overlap has negligible effect. Because ThumbSet is drawn from user-uploaded scores, some of its windows are alternate transcriptions of compositions that also appear in the canonical test set – an overlap invisible to a file-identity check. Ex-

Decoding	M_{gen}	M_{soft}	M_{high}
Independent argmax (no CRF)	65.5	87.5	71.4
Interval-conditioned CRF	65.7	**87.9**	71.8
+ hard chord rule	**65.8**	**87.9**	**71.9**
+ soft ergonomic nudges	65.7	87.5	71.8

Table 3: Decoding ablation on the 30 canonical test pieces. The CRF's change in aggregate agreement over independent per-note argmax is small and not statistically significant (Wilcoxon signed-rank $W = 210.0$, $p = 0.65$; CRF wins on 17/30 pieces). Adding the hard anatomical chord rule is the best configuration on all three metrics while also removing the 22 chord violations (94 notes, 1.3% of 1,674 chords) that unconstrained decoding produces. The hand-authored soft ergonomic nudges give back the M_{soft} gain and are excluded from the final model.

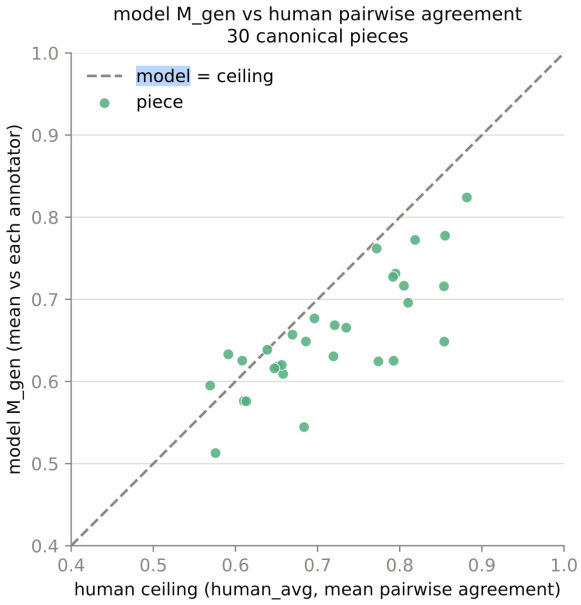

Figure 1: Per-piece model agreement against human pairwise agreement on the 30 canonical test pieces. Each point is one piece; the dashed line marks equality. Model performance tracks inter-annotator agreement across pieces — both are lower on pieces where annotators themselves disagree — while sitting below it on most pieces.

cluding these 1,366 windows and re-running the full pretrain, fine-tune, and evaluation chain changes results marginally and not systematically (M_{gen} 65.8 → 65.8, M_{soft} 87.4 → 87.9, M_{high} 70.9 → 71.9). We report the excluded configuration throughout, and note the comparison as evidence that this form of leakage was immaterial at this scale rather than as a correction to previously reported numbers.

On "beating the human ceiling." A tempting framing, which we avoid, is that the model exceeds human inter-annotator agreement. On the primary metric the model sits *below* the pairwise human ceiling (65.8 vs 71.8). More fundamentally, pairwise inter-annotator agreement is a lower bound on the difficulty of matching any *individual* annotator, not a bound on matching a consensus; a model that predicts the consensus fingering can exceed pairwise agreement without being "superhuman," since aggregating several annotators is closer to each of them than any two are to each other. We therefore treat inter-annotator agreement as context, not as a bar the model has cleared.

Structured decoding contributes modestly. The CRF improves over independent per-note prediction by a small, statistically insignificant margin on aggregate match rate. Its clearer effect is on output structure: runs of a single finger held across moving pitch fall from 61 to 28. Hand-authored ergonomic constraints did not help and were dropped; only the hard anatomical chord rule, which removes real violations at no cost, is retained.

Comparison scope. We compare against the HMM and CRF baselines on matched metrics, but the literature also contains neural models we have not run head-to-head, including a BiLSTM (Guan et al., 2021) and a pitch-difference matching model (Guan et al., 2022) reported to exceed the third-order HMM. Our claim is therefore competitiveness with the HMM and CRF baselines on this split, not a general state-of-the-art result; a controlled comparison against the neural models, and re-running all baselines through our own evaluation code, is the clear next step. We also omit the recombination metric M_{rec} used by (Nakamura et al., 2020), reporting the three match-rate statistics most consistently available across prior work.

Evaluation protocol. The validation set does double duty for model selection and we report on the canonical 30 without a further held-out split beyond it – standard in this subfield given the size of PIG, but worth stating. Finally, match-rate metrics on PIG have known subtleties when a piece admits multiple score representations (Randolph et al., 2023), which we inherit.

8. Conclusion

We presented KeyGenius, a compact Transformer encoder with an interval-conditioned CRF for automatic piano fingering, trained by weakly-labelled pretraining followed by fine-tuning on PIG. On the canonical PIG test set it is competitive with the classical HMM and

CRF baselines on matched match-rate metrics, with the gain over a PIG-only variant attributable to pretraining rather than the architecture in isolation. Along the way we report several results that qualify common assumptions: structured CRF decoding does not significantly improve aggregate annotator agreement over independent per-note prediction, though it more than halves runs of a single finger held across moving pitch; hand-authored ergonomic decoding constraints do not help and were dropped, while a hard anatomical chord rule removes real violations at no cost; and pairwise inter-annotator agreement, often treated as a ceiling, is a weak bound for a model predicting consensus. The system is deployed and in use by piano teachers. Controlled comparison against recent neural baselines on a shared evaluation harness is the natural next step.

9. Reproducibility

Code for the model, training, and evaluation is publicly available at `https://github.com/HACkerrr121/KeyGenius`. All experiments use seed 1337 on a single NVIDIA A100-SXM4-40GB GPU; checkpoint metadata sidecars record the hardware and wall-clock time for each run. The PIG and ThumbSet datasets are used under their respective terms and credited to their creators.

10. Competing interests

The authors have no competing interests to declare.

References

Guan, H., Yan, Z., and Hsu, T. (2021). Automatic piano fingering estimation using recurrent neural networks. In *Nordic Sound and Music Computing Conference (Nordic SMC)*.

Guan, X., Zhao, H., and Li, Q. (2022). Estimation of playable piano fingering by pitch-difference fingering match model. *EURASIP Journal on Audio, Speech, and Music Processing*, 2022(1):7.

Iman, A. P. and Ahn, C. W. (2025). Elite episode replay memory for polyphonic piano fingering estimation. *Mathematics*, 13(15):2485.

Nakamura, E., Saito, Y., and Yoshii, K. (2020). Statistical learning and estimation of piano fingering. *Information Sciences*, 517:68–85.

Parncutt, R., Sloboda, J. A., Clarke, E. F., Raekallio, M., and Desain, P. (1997). An ergonomic model of keyboard fingering for melodic fragments. *Music Perception: An Interdisciplinary Journal*, 14(4):341–382.

Ramoneda, P., Jeong, D., Nakamura, E., Serra, X., and Miron, M. (2022). Automatic piano fingering from partially annotated scores using autoregressive neural networks. In *Proceedings of the 30th ACM International Conference on Multimedia (MM '22)*.

Randolph, D. A., Di Eugenio, B., and Badgerow, J. (2023). Modeling piano fingering decisions with conditional random fields. Technical report, University of Illinois Chicago.

Yonebayashi, Y., Kameoka, H., and Sagayama, S. (2007). Automatic decision of piano fingering based on a hidden Markov models. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 7, pages 2915–2921.