Detecting Hands in Piano MIDI Data

Aristotelis Hadjakos, Simon Waloschek, Alexander Leemhuis
{a.hadjakos, s.waloschek}@cemfi.de, alexander.leemhuis@gmx.de
Center of Music and Film Informatics, Detmold, Germany
Detmold University of Music
OWL University of Applied Sciences and Arts

ABSTRACT

When a pianist is playing on a MIDI keyboard, the computer does not know with which hand a key was pressed. With the help of a *Recurrent Neural Network* (RNN), we assign played MIDI notes to one of the two hands. We compare our new approach with an existing heuristic algorithm and show that RNNs perform better. The solution is real-time capable and can be used via Open Sound Control (OSC) from any programming environment. A non real-time capable variant provides slightly higher accuracy. Our solution can be used in music notation software to assign the left or right hand to the upper or lower staff automatically. Another application is live playing, where different synthesizer sounds can be mapped to the left and right hand.

CCS CONCEPTS

• **Computing methodologies** → *Neural networks*; • **Applied computing** → **Performing arts**.

KEYWORDS

piano, music, dataset, neural networks

1 INTRODUCTION

Automatic music transcription is an important topic in Music Information Retrieval. The goal is to create a symbolic representation from a sound recording. Especially for piano music, many promising solutions have been proposed in recent years [2, 5, 8, 11, 13, 21]. But simply transcribing recordings to MIDI events represents only the first step towards a useful piano score. Afterwards, the notes must be quantized, divided into beats, and distributed to the lower and upper staff. We introduce a method to determine which hand was used by the piano player to play a note. The method only uses MIDI data and does not need any additional hardware like cameras over the keyboard. It is especially useful for two applications:

- Interactive notation: Many notation solutions allow the user to record a piece using a MIDI interface. This is especially well suited for piano music. But then, among other things like quantization and correction of playing errors, the assignment of the individual notes to the upper or lower staff must be made. Usually the left hand is written in the lower staff and the right hand in the upper staff. So far, most systems allocate notes by splitting at the middle C: deeper notes are assigned to the lower staff; all others to the top. This approach is highly inaccurate as soon as one hand crosses this split point, which is why the users then have to manually correct the assignment.
- Live playing: The assignment of notes to hands can also be interesting for live playing with synthesizers or other virtual instruments. It opens up the possibility to assign different sounds to the left and right hand. Thus, new expression dimensions are created without the need for additional hardware.

This paper provides three contributions:

- We show that the assignment of notes to hand with an RNN is possible. We achieve accuracies of 93.25% in real-time operation and 94.47% in non-real-time operation.
- We are extending an existing Kalman filter based approach [10] to bidirectional non real-time operation. With 92.17% accuracy, this is about 1.4 percentage points more accurate than the existing real-time capable solution.
- We have created an extensive dataset for training the RNN. It contains live piano music, with right and left hand encoded as separate MIDI tracks. In addition to the outlined use cases, the data set may be of interest to other projects, especially in musical performance research.

Dataset and code are publicly available on Github under an open source license: https://github.com/cemfi/hannds

2 RELATED WORK

Kilian and Hoos [14] proposed a method that separates a piece into different voices for notation. Chords may occur in one voice. The method allows to select the number of available voices. Therefore, it can be used to find a left and a right part of a MIDI performance if one chooses the number of voices = 2. To separate voices, Kilan's and Hoos's method divides the piece into a sequence of slices with overlapping notes and determines the voice separation by minimizing a cost function using stochastic local search. While this approach is applicable for notation, it can not be used in real time because the slice of overlapping notes can not be instantly determined when a note is played. In addition, the stochastic local search algorithm operates on the entire piece. Gorodnichy & Yogeswaran [9] use a camera mounted over the keyboard to detect which hand and which finger has played a note. Oka & Hashimoto use a similar setup for this, but use a depth camera instead [18]. Akbari & Cheng [1] transcribe whole pieces of music with a camera image of the keyboard.

MuC'19 Workshops, Hamburg, Deutschland

https://doi.org/10.18420/muc2019-ws-578

In 1997, Parncutt et al. [19] proposed a piano fingering prediction system based on ergonomic constraints of the hand. More recently, HMM-based solutions have been proposed [17, 24]. To perform fingering prediciton, it is necessary to assign each note to one hand. Nakamura et al. [17] report the accuracy of predicting the accuracy of hand detection with their merged-output HMM. The reported error rates range from 1.4% to 18.7% depending on the piece. A comparison with our results reported in Section 4 is problematic since the results depend heavily on the analyzed pieces.

There are many approaches to separate the voices contained in the piece from MIDI data. Among others there are rule-based solutions [3, 15, 23], solutions based on Hidden Markov Models [16] and solutions based on information theory [7]. Particularly relevant to us is the use of neural networks for voice separation, such as the approach of de Valk & Weyde [6]. They used a neural network for voice separation in MIDI representations of Fugues and Inventions by J.S. Bach. For each note a feature vector consisting of 33 handcrafted entries is given to the neural network.

3 HAND ASSIGNMENT

We realized hand assignment in two different ways:

- with an RNN that makes a prediction with a binary output for each MIDI event (Section 3.1) and
- with an extension of a Kalman filter based approach, in which the hand positions are modeled as Gaussian distributions (Section 3.2).

3.1 Recurrent Neural Network

To collect training data, piano students of a music conservatory played various pieces on two keyboards simultaneously. The keyboards were placed on top of each other. Pianists played with their right hand on the upper keyboard and their left on the lower one. The parallel MIDI streams were recorded synchronously with a digital audio workstation. We recorded 122 MIDI files with a total of 222,714 notes from different musical eras. Some of the pieces were played prima vista while others were well rehearsed. The dataset contains music ranging from the early 18th century up to modern jazz pieces.

We used a very simple input feature vector with four entries. The feature vector encoded MIDI pitch and MIDI velocity as a number ranging from 0 to 127, the time difference to the previous event in seconds and binary note-on / note-off information. Alternatively we also used the same format introduced by the Magenta project for their Performance RNN [22]. This feature vector consists of 388 entries in total:

- 2×128 entries indicate whether a note-on and note-off event occurred at the current time.
- a 1-hot-encoding with 100 entries indicates the time difference between successive events. Encoded time intervals are between 10ms and 1s.
- The MIDI velocity is quantized in 32 bins and also provided as 1-hot-encoding.

The RNN consists of 2 hidden layers with 70 neurons each. We use Gated Recurrent Units (GRU) [4] or Long Short-Term Memory (LSTM) [12] units. The 1-dimensional output is calculated as a linear combination of the output of the top layer followed by a sigmoid nonlinearity, see Fig. 1. At each time step, the RNN provides a 1-dimensional output indicating whether the left (output = 0) or the right hand (output = 1) has played the note. It is possible that the network (erroneously) predicts that the note-on event originated from one hand and the note-off event from the other. The RNN is trained with cross-entropy loss. We use the Adam optimizer with standard hyperparameter settings.

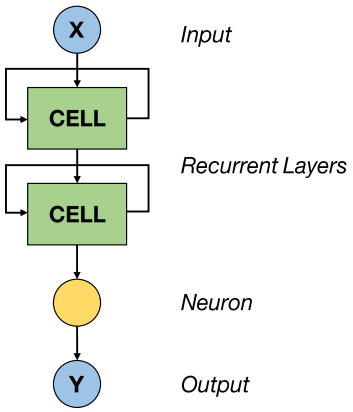

Figure 1: Architecture of the neural network. The cells are either LSTM or GRU units.

For non-real-time operation, it is possible to use the RNN bidirectionally. While the forward RNN processes the MIDI data in the normal time course, the backward RNN handles the MIDI sequence from back to front. The 70-dimensional outputs of the two RNNs are combined in each time step. The combined 140-dimensional vector is the input for the 1-dimensional output neuron. As our evaluation shows, the bidirectional approach achieves better accuracy.

3.2 Kalman filter

One of the authors [10] developed an algorithm for detecting hands based on a Kalman filter. This algorithm is described in the following Section 3.2.1. Section 3.2.2 introduces a new bidirectional extension of the algorithm.

3.2.1 Existing approach. Hand assignment is accomplished through two mechanisms: the identification of "unique notes" and the relation of a played note with respect to estimated hand positions.

While MIDI data is arriving, the computer recognizes "unique notes". These are notes where it seems implausible that they were played with one hand alone because they are too far apart. If two notes more than one eleventh apart sound simultaneously, they are recognized as unique notes. The lower note is then assigned to the left hand and the upper note to the right hand. If a note is not a unique, it is assigned to a hand based on the distance of the note to the estimated hand positions.

The position of the hands is estimated with a Kalman filter for each hand. Note-on events are passed to the Kalman filter of the assigned hand. A received note-on event is interpreted as an approximate measurement of hand position. The variance σ_m^2 expresses the measurement uncertainty. When a note-on message is received, the variance is calculated, which expresses the uncertainty in the

	Bidirectional	Simple MIDI	Magenta MIDI	Kalman
RNN using GRU		**93.25% (0.09%)**	92.64% (0.07%)	
RNN using LSTM		93.03% (0.07%)	92.58% (0.07%)	
Kalman filter				90.79%
RNN using GRU	✓	**94.47% (0.06%)**	93.89% (0.05%)	
RNN using LSTM	✓	94.20% (0.02%)	93.65% (0.03%)	
Kalman filter	✓			92.17%

Table 1: Validation results for all proposed approaches. Best results are emphasized, standard deviation in parentheses.

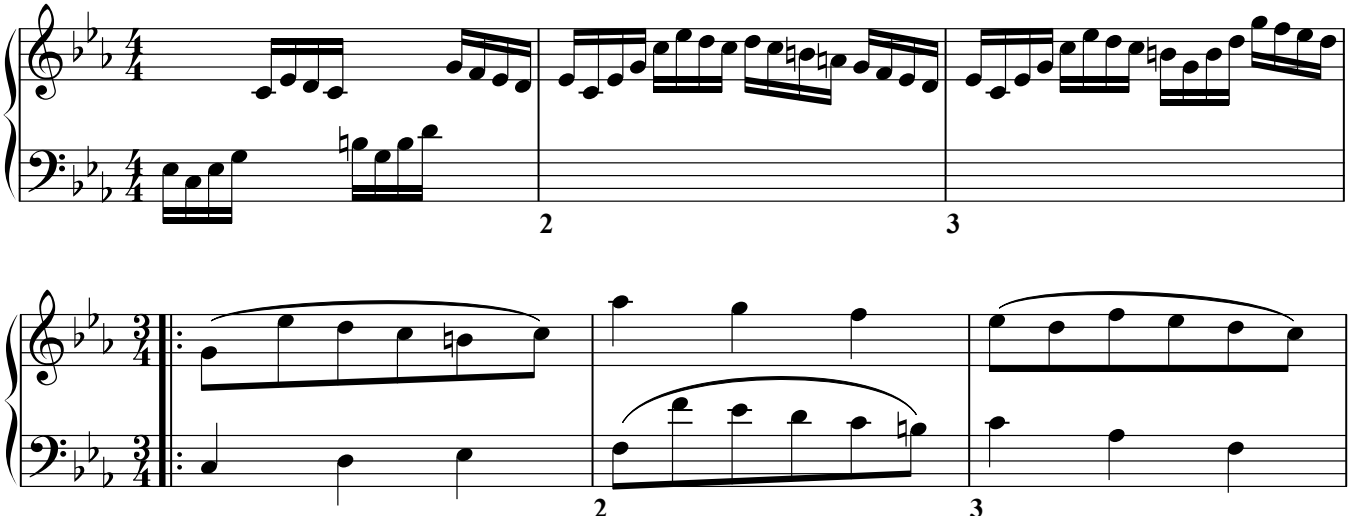

Figure 2: The Solfeggietto c minor (H 220, Wq. 117: 2) from the C.P.E. Bach (above) and the minuet from the French Suite no. 2 in c minor (BWV 813) by J.S. Bach (below).

position before inserting the measurement $\sigma_p^2(t_2^-)$. Let t_1 be the time the last note was assigned to the Kalman filter and t_2 be the time the new note was received. The uncertainty of the position before taking the new measurement $\sigma_p^2(t_2^-)$ is then based on the time difference between the two notes $t_2 - t_1$, a constant term σ_s^2 and updates the previous one Uncertainty after inclusion of the measurement $\sigma_p^2(t_1^+)$.

$$\sigma_p^2(t_2^-) = \sigma_p^2(t_1^+) + (t_2 - t_1) \cdot \sigma_s^2 \tag{1}$$

The uncertainty of the position after incorporating the measurement $\sigma_p^2(t_2^+)$ is updated based on the uncertainty of the position before incorporating the measurement $\sigma_p^2(t_2^-)$ and the constant term σ_m^2 that expresses measurement uncertainty.

$$\sigma_p^2(t_2^+) = \sigma_p^2(t_2^-) - \frac{\sigma_p^2(t_2^-)}{\sigma_p^2(t_2^-) + \sigma_m^2}\sigma_p^2(t_2^-) \tag{2}$$

Let n be the pitch of the note received at t_2. Then the new position $p(t_2)$ is estimated based on the old position $p(t_1)$, the uncertainty of the position before incorporating the measurement $\sigma_p^2(t_2^-)$, and the pitch of the received note n.

$$p(t_2) = p(t_1) + \frac{\sigma_p^2(t_2^-)}{\sigma_p^2(t_2^-) + \sigma_m^2}(n - p(t_1)) \tag{3}$$

3.2.2 Bidirectional extension. In addition to the normal time course, a second Kalman filter is operated backwards. Furthermore, the roles of note-on and note-off events are changed. The Kalman filter operates as in the normal direction only that a note "begins" with a note-off event and "ends" with a note-on event. Other than that the assignment for the backward pass is identical to the procedure described in Section 3.2.1. This results in an alternative assignment, which may assign a note differently than the Kalman filter in the forward pass.

But how can possibly contradictory assignment be meaningfully combined with simultaneous use of forward and backward direction? This is possible through a maximum likelihood approach. For the assignment we can look at the 4 likelihoods, which are determined using the MIDI pitch of the note in question, the respective predicted position (mean value) and the standard deviation of the four following Gaussian distributions: (1) right hand, forward pass; (2) left hand, forward pass; (3) right hand, backward pass; (4) left hand, backward pass. If the note evaluated at (1) or (3) produces the largest likelihood, the note is assigned to the right hand, otherwise it is assigned to the left.

4 RESULTS & DISCUSSION

We evaluated the RNN with 10-fold cross-validation. For this purpose, 10 percent of the pieces were held out for the evaluation while

the RNN was trained with the other pieces. This ensures that the evaluation data is completely disjoint. If we had cut pieces in the middle, repetitions and parallel variations could interfere with the results. The bidirectional variant of the RNN achieves better scores with 93.1 to 94.6%. The real-time forward variant provides an accuracy of 92.8 to 93.3%. For the features under consideration, the format "Simple MIDI" performs slightly better than the Magenta format and GRU units perform slightly better than LSTM units.

The results of the Kalman-based approach on the same data are slightly worse compared to the RNN. Again, the bidirectional approach with 92.17% accuracy gives the best results. The real-time forward approach provides an accuracy of 90.79% percent. Tab. 1 gives an overview over all results.

Both the RNN and the Kalman-based approach show large variances in recognition accuracy from piece to piece. As an example we will consider the Solfeggietto c minor (H 220, Wq. 117: 2) by C.P.E. Bach and the Minuet from the French Suite no. 2 in c minor (BWV 813) by J.S. Bach (see Fig. 2). The first bars of the Minuet allow only one sensible division of notes to hands. Things are quite different with the Solfegietto: the monophonic melody is usually played alternately on both hands. But quite different solutions are very well conceivable. For example, it would be possible to play everything with the right hand. It is therefore hard to determine which sound was played with which hand. The accuracy of hand assignment in the Solfeggietto lies between 60 and 80% while it works with 100% accuracy in the Menuet.

5 CONCLUSION AND FUTURE WORK

We have shown that it is possible to detect which hand a pianist has played a note based on MIDI data that is fed into a RNN. We have developed both a real-time capable forward variant and a non real-time capable bidirectional variant that provides a better accuracy. While the achievable accuracy depends on the piece, we achieve an overall accuracy of 94.47% in non real-time and 93.25% in real-time conditions. We have extended an existing real-time capable Kalman filter based approach for non real-time bidirectional processing, which increases its accuracy by about 1.4 percentage points to 92.17%.

We have implemented an OSC-based service for the real-time capable hand assignment variants. For this purpose the MIDI information is sent from a programming environment such as Max/MSP or pureData to the Python-based hand assignment service as an OSC message. After the analysis, the service returns an OSC message indicating the result.

Our RNN could be adapted for the prediction of fingerings. While the input would remain unchanged, the fingers in the output layer could be represented using a 1-hot-encoding. The training data needed for this could be created with the Dactylize system [20]. In the Dactylize system, the finger-to-sound mapping is determined by an electronic measuring system, with the keys of a MIDI piano covered with metal tape and the fingers wrapped with an electrically conductive material.

REFERENCES

[1] Mohammad Akbari and Howard Cheng. 2015. Real-time piano music transcription based on computer vision. *IEEE Transactions on Multimedia* 17, 12 (2015), 2113–2121.

[2] Tian Cheng, Matthias Mauch, Emmanouil Benetos, Simon Dixon, et al. 2016. An attack/decay model for piano transcription. Proc. of the conference of the International Society for Music Information Retrieval (ISMIR).

[3] E. Chew and X. Wu. 2004. Separating voices in polyphonic music: A contig mapping approach. In *Computer Music Modeling and Retrieval: Second International Symposium*. Springer, 1–20.

[4] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

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

[6] R de Volk and Tillman Weyde. 2018. Deep neural networks with voice entry estimation heuristics for voice separation in symbolic music representations. In *Proc. of the conference of the International Society for Music Information Retrieval (ISMIR)*.

[7] Michele Della Ventura. 2018. Voice Separation in Polyphonic Music: Information Theory Approach. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*. Springer, 638–646.

[8] Sebastian Ewert and Mark Sandler. 2016. Piano transcription in the studio using an extensible alternating directions framework. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24, 11 (2016), 1983–1997.

[9] Dmitry O Gorodnichy and Arjun Yogeswaran. 2006. Detection and tracking of pianist hands and fingers. In *The 3rd Canadian Conference on Computer and Robot Vision (CRV'06)*. IEEE, 63–63.

[10] Aristotelis Hadjakos, François Lefebvre-Albaret, and IRIT Toulouse. 2009. Three methods for pianist hand assignment. In *6th Sound and Music Computing Conference*. 321–326.

[11] Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse Engel, Sageev Oore, and Douglas Eck. 2018. Onsets and frames: Dual-objective piano transcription. In *Proc. of the conference of the International Society for Music Information Retrieval (ISMIR)*. 50–56.

[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[13] Rainer Kelz, Matthias Dorfer, Filip Korzeniowski, Sebastian Böck, Andreas Arzt, and Gerhard Widmer. 2016. On the potential of simple framewise approaches to piano transcription. *arXiv preprint arXiv:1612.05153* (2016).

[14] J. Kilian and H.H. Hoos. 2002. Voice separation—a local optimisation approach. In *Proc. of the International Conference on Music Information Retrieval (ISMIR)*.

[15] S.T. Madsen and G. Widmer. 2006. Separating voices in MIDI. In *Proceedings of the 9th International Conference in Music Perception and Cognition (ICMPC2006)*. 22–26.

[16] Andrew McLeod and Mark Steedman. 2016. HMM-based voice separation of MIDI performance. *Journal of New Music Research* 45, 1 (2016), 17–26.

[17] Eita Nakamura, Nobutaka Ono, and Shigeki Sagayama. 2014. Merged-Output HMM for Piano Fingering of Both Hands.. In *Proc. of the conference of the International Society for Music Information Retrieval (ISMIR)*. 531–536.

[18] Akiya Oka and Manabu Hashimoto. 2013. Marker-less piano fingering recognition using sequential depth images. In *The 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision*. IEEE, 1–4.

[19] Richard Parncutt, John A Sloboda, Eric F Clarke, Matti Raekallio, and Peter Desain. 1997. An ergonomic model of keyboard fingering for melodic fragments. *Music Perception: An Interdisciplinary Journal* 14, 4 (1997), 341–382.

[20] David A Randolph and Barbara Di Eugenio. 2016. Dactylize: automatically collecting piano fingering data from performance. In *Proceedings of the Extended Abstracts for the Late-Breaking Demo Session of the 17th International Society for Music Information Retrieval Conference*.

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

[22] Ian Simon and Sageev Oore. 2017. Performance RNN: Generating Music with Expressive Timing and Dynamics. https://magenta.tensorflow.org/performance-rnn.

[23] David Temperley. 2004. *The cognition of basic musical structures*. MIT press.

[24] Yuichiro Yonebayashi, Hirokazu Kameoka, and Shigeki Sagayama. 2007. Automatic Decision of Piano Fingering Based on a Hidden Markov Models.. In *IJCAI*. 2915–2921.