# Synthesizing Training Data for Handwritten Music Recognition

Jiří Mayer(✉) and Pavel Pecina

Institute of Formal and Applied Linguistics, Charles University, Prague, Czech Republic
{mayer,pecina}@ufal.mff.cuni.cz

**Abstract.** Handwritten music recognition is a challenging task that could be of great use if mastered, e.g., to improve the accessibility of archival manuscripts or to ease music composition. Many modern machine learning techniques, however, cannot be easily applied to this task because of the limi'ted availability of high-quality training data. Annotating such data manually is expensive and thus not feasible at the necessary scale. This problem has already been tackled in other fields by training on automatically generated synthetic data. We bring this approach to handwritten music recognition and present a method to generate synthetic handwritten music images (limited to monophonic scores) and show that training on such data leads to state-of-the-art results.

**Keywords:** Handwritten music recognition · Synthetic training data generation

## 1 Introduction

Handwritten music recognition (HMR) is the automatic process of converting handwritten sheet music to a machine-readable format. Having the music in digital form lets us easily preserve, analyze, search through, modify, engrave, or play the music [5]. While printed music recognition is still far from a solved problem, HMR has additional difficulties to contend with, due to the vast variability in handwriting style [3] and limited availability of data for training [12]. In addition to the low variability of graphical style, recognition of printed music has easier access to synthetic training data that can be generated using engraving tools such as Lilypond[1] or Verovio[2]. This has already been reflected in the literature: for instance, DeepScores [26] consists of 300k synthetic images of printed music scores for performing symbol classification, image segmentation, and object detection; the PrIMuS dataset [6] provides more than 80k printed images of music staves with ground truth readily available for end-to-end HMR learning.

Acquisition of training data for HMR, however, requires an expensive and slow manual process [12], which involves either transcription of existing sheet music into a symbolic representation (producing annotations) or handwriting

[1] http://lilypond.org/.
[2] https://www.verovio.org/.

© Springer Nature Switzerland AG 2021

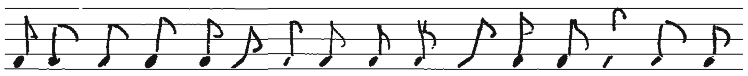


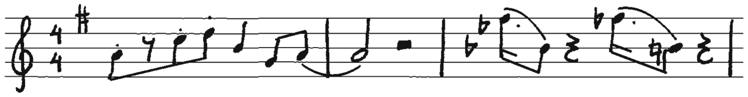
**Fig. 1.** Examples of eighth notes sampled from the MUSCIMA++ dataset. The handwriting style between writers varies greatly. Noteheads vary in shape and size, stems vary in slant and size, and sometimes the stem does not connect to the head.

music symbols on sheets of paper based on existing music representation (producing images). Both approaches require capturing the high variability of handwriting styles and therefore engagement of many writers (see Fig. 1).

The existing datasets for HMR are scarce. Most handwritten music datasets contain only individual symbols. The only resource containing entire staves is CVC-MUSCIMA [8], which comprises of 1,000 scanned sheets of music. While the variability of handwriting styles in this dataset is rather rich, the variability of music content is very limited (about 110 unique melody staves) and certainly not sufficient for end-to-end learning. This situation forces many researchers in the field to resort to data augmentation or transfer learning [2]. Despite all this, nobody has yet tried to create synthetic training data for HMR.

Synthetic data is data generated by a computer simulation of a real-world process. Training on such data is being used in machine learning where original (authentic) data is not available in sufficient amounts and the data generation process can be simulated by a computer program (e.g., image classification [20], natural scene text recognition [14], or handwritten text recognition [16]).

In this paper, we present *Mashcima* – an engine designed to generate realistic images of handwritten music for training HMR models. It exploits symbol masks extracted from the MUSCIMA++ dataset [12] (a richly annotated subset of CVC-MUSCIMA) that are rearranged and placed on one staff according to a music annotation in a newly proposed encoding adopted from the PrIMuS dataset [6] (see Fig. 2). *Mashcima* is highly configurable and customizable to alter the resulting visual style of the image. It can, for instance, mix handwriting of multiple people or generate images in the style of only one author. A non-trained human reported difficulties distinguishing a real-world sample from a well-synthesized one (see Fig. 3). The presented version of *Mashcima* generates only monophonic scores (music with one voice, without chords, and with musical symbols spanning only one staff). Polyphonic music presents challenges with regards to music representation and will be addressed in future versions.



```
clef.G-2 time.4 time.4 #4 e=-1 . er =e=1 . =e2 . q0 e=-2 =e-1 ( |
    ) h-1 hr | b0 b3 s=4 * ( ) =e0 qr b4 s=4 * ( ) N0 =e0 qr |
```

**Fig. 2.** An example of a synthetic image generated from the given annotation.

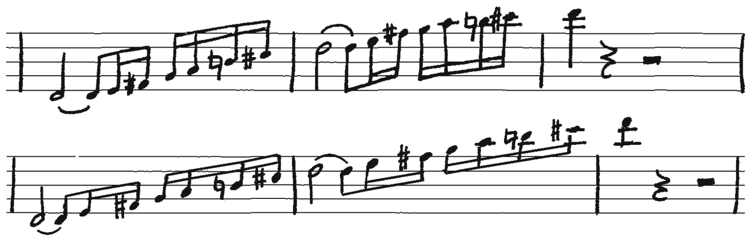


**Fig. 3.** Comparison of a real-world image (top) to a synthetic image (bottom) containing the same music. The top image is taken from the CVC-MUSCIMA dataset. The synthetic image is generated by *Mashcima* using symbols produced by a single writer.

The evaluation is conducted using a model based on a Convolutional Recurrent Neural Network (CRNN) with the Connectionist Temporal Classification (CTC) loss function, similar to the model proposed for printed music recognition by Calvo-Zaragoza and Rizo [6]. The model, trained on synthetic data produced by *Mashcima*, is evaluated on an unseen subset of MUSCIMA++ and a fully independent sample of real-world sheet music. A quantitative and qualitative comparison with previously published state-of-the-art results indicates the superior performance of our approach. The complete code of *Mashcima* and the experiments is available on GitHub.[3]

This text continues with an overview of related work (Sect. 2), followed by the proposal of the music encoding used by *Mashcima* (Sect. 3) and a detailed description of the *Mashcima* system (Sect. 4). Section 5 then presents our experiments, including details of the HMR model, training configurations, results, and analysis. Section 6 concludes the text and outlines our future work.

## 2 Related Work

Optical Music Recognition (OMR) aims at converting images of musical scores into a computer-readable form [5,9,22]. Traditionally, the research has mainly focused on printed music in the common Western music notation. Recognition of printed music is less challenging compared to handwritten music due to the enormous differences in the variability of printing/handwriting styles.

Until recently, most of the approaches to printed OMR employed the traditional pipeline consisting of several recognition steps (preprocessing, music object detection, notation assembly, and encoding) [5]. In 2017, van der Wel et al. [28] presented the first end-to-end OMR approach based on Convolutional Neural Networks (CNNs) and sequence-to-sequence models, although limited to monophonic music scores only. In 2018, Calvo-Zaragoza et al. [6] presented an end-to-end model (also limited to monophonic scores) based on CRNN [21] and CTC [11] that did not require alignment of graphical symbols and ground truth.

[3] https://github.com/Jirka-Mayer/Mashcima.

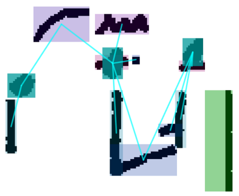


**Fig. 4.** Symbol masks and their relationships form a notation graph in MUSCIMA++.

The initial attempts to HMR focused on particular stages of the traditional OMR pipeline including preprocessing [4], staff removal [19], symbol detection and recognition [27]. In 2019, Baró et al. [2] published the first baseline for full HMR (not limited to monophonic music) based on CRNN and transfer learning. The model was pretrained on the PrIMuS dataset (printed notation) and fine-tuned on the MUSCIMA++ data (handwritten notation). Calvo-Zaragoza et al. [7] used CRNN models for HMR of mensural notation. The attention mechanism used in HMR of historical music was explored by Baró et al. [1]. Pacha et al. [18] attempted to reconstruct full notation graphs.

There are only two papers focusing on synthetic data generation for OMR: DeepScores [26] and PrIMuS [6], both synthesising printed scores.

## 2.1 Datasets

This section provides an overview of datasets exploited in our work. For a more complete overview, see the webpage by Alexander Pacha[4].

CVC-MUSCIMA [8] is a dataset originally designed for two tasks: writer identification and staff removal. It contains 20 pages of music, each manually transcribed by 50 different writers (1,000 pages total). It contains ground truth for staff removal but not for music recognition. We manually annotated a small subset of the dataset to be used for evaluation in our experiments (see Sect. 5).

MUSCIMA++ [12] is a subset (140 pages) of CVC-MUSCIMA enriched with detailed information about the placement and relationships of individual music symbol primitives in the form of a notation graph (MuNG, see Fig. 4) which allows recovering pixel masks of music symbols (e.g., notes) and their positions with respect to staff lines. We harvest this dataset to collect samples of the pixel masks to be used for rendering synthetic images of handwritten scores.

PrIMuS [6] is a dataset of 87,678 music incipits taken from the RISM catalog[5] containing over 1.2M records from various musical sources. Each PrIMuS incipit is encoded in five machine-readable formats (including an agnostic encoding format) and engraved in a raster image (Fig. 5). This data was sourced for real-world music annotations to produce the synthetic data for our experiments.

[4] https://apacha.github.io/OMR-Datasets/.

[5] http://opac.rism.info/.

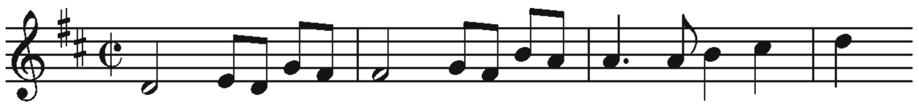


**Fig. 5.** An original image of an incipit from the PrIMuS dataset (rendered by Verovio).

## 3 Music Representation

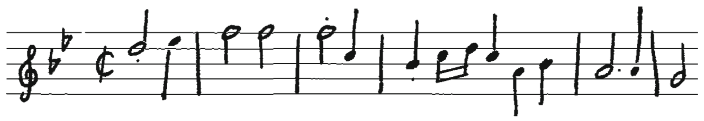
Several formats have been proposed for representing music in a machine-readable form, differing in nature and purpose: e.g., MIDI [24] is designed for music playback, MusicXML [10] is primarily for music notation editing and engraving, MEI [23] captures various notation semantics, MuNG [12] was recently proposed for precise annotation of symbols in (handwritten) scores and their relationships.

In this work, we employ our own encoding similar to the agnostic encoding used in the PrIMuS dataset [6]. The PrIMuS agnostic encoding represents each staff as a sequence of tags corresponding to graphical symbols in the score, following the relative left-to-right and top-down ordering. Each graphical symbol is represented as a tag (without any predefined musical meaning) and position in the staff (line/space). Note beams are vertically sliced and slurs are represented by opening and closing tags surrounding a subsequence of symbols.

Such an agnostic encoding presents several advantages for HMR: It is very simplistic and describes only the way a piece of music looks (not what it means) and does not enforce any implicit constraints, such as proper rhythm (e.g., number of beats per measure). This implies that machine learning models aiming to produce this graphical-level encoding as output avoid the issue of large-distance dependencies that arise when interpreting music notation, such as clefs influencing the meaning of notes on the entire line, and thus have an easier task. Once this agnostic encoding is recovered, the musical semantics can be computed deterministically. There is no explicit alignment to the corresponding image (musical symbols and their absolute positions), which allows easy and relatively fast annotation of music scores from scratch. A single annotation can represent multiple scores (containing the same music but with a different appearance, which is typical in handwritten music). This is also important for generating synthetic data where multiple diverse images can be rendered for a single input. Most importantly, the sequential nature of the agnostic encoding allows feasible training of neural network models (e.g., using CTC) and straightforward evaluation based on string edit distance (e.g., Symbol Error Rate, see Sect. 5).

The PrIMuS agnostic encoding, however, was not designed for direct manual annotation (typing by hand). The tags have rather long names, which hinders readability. To simplify and speed up the manual annotation of the evaluation data used in our experiments, we designed our own *Mashcima* encoding, although not principally different from the PrIMuS agnostic encoding (see Fig. 6).

Tag names in the *Mashcima* encoding are considerably shorter and visually similar to the musical symbols they represent. Notes are represented by a one-letter symbol signaling their duration and a pitch number. The pitch `0`



```
clef.G-2 b0 b3 time.C/ h2 . q3 | h4 h4 | h4 . q1 |
q0 . s=1 =s2 q1 q-1 q0 | h-1 * q-1 | h-2
```

```
clef.G-L2 accidental.flat-L3 accidental.flat-S4 metersign.C/-L3 note.half-L4 dot-S3
note.quarter-S4 barline-L1 note.half-L5 note.half-L5 barline-L1 note.half-L5 dot-S5
note.quarter-S3 barline-L1 note.quarter-L3 dot-S2 note.beamedRight2-S3
note.beamedLeft2-L4 note.quarter-S3 note.quarter-S2 note.quarter-L3
barline-L1 note.half-S2 dot-S2 note.quarter-S2 barline-L1 note.half-L2
```

**Fig. 6.** Image of an incipit from the PrIMuS dataset produced by *Mashcima*. Below the image are the *Mashcima* encoding and the PrIMuS agnostic encoding of the image.

corresponds to notes on the center staff line. This makes the pitch space symmetrical and the annotation less prone to errors. Even pitches correspond to notes on staff lines while odd pitches correspond to spaces. Rests are represented by the letter `r` instead of a pitch number, slurs are denoted by round brackets.

Up until this point, the two encodings are equivalent. The differences are the following: *Mashcima* distinguishes between duration dots `*`, `**` and staccato dots `.` while the PrIMuS agnostic supports only duration dots. The PrIMuS agnostic encodes multimeasure rest digits and time signature digits the same way; our encoding treats them differently (but does not support numeric multimeasure rests, only symbolic ones). *Mashcima* also introduces the `?` token for unsupported symbols. The current version of the synthesizer does not support grace notes, fermatas, and tuplets (they will be added in future versions).

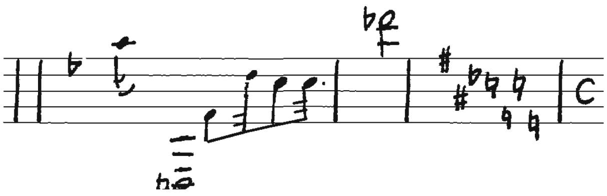
The complete description of the *Mashcima* encoding is available on GitHub, including a (deterministic) tool to convert the PrIMuS agnostic encoding to the *Mashcima* encoding which is used in our experiments.

## 4 Synthetic Data Generation

The aim of *Mashcima* is to produce synthetic images mimicking handwritten music scores to train a system for recognition of such real-world images. This is achieved by reusing images of individual music symbols extracted from scans of real (authentic) handwritten scores and rearranging them onto a blank image given a prescription specified in the *Mashcima* encoding. We describe all stages of the process: 1) acquisition of music symbol masks, 2) obtaining ground-truth annotations, 3) symbol placement on a staff, and 4) image rendering.

### 4.1 Acquisition of Music Symbol Masks

*Mashcima* collects authentic instances of music symbols from MUSCIMA++ in the form of pixel masks (image pixels belonging to the given symbol) in binarized images of handwritten scores. The masks are obtained for the entire symbols



```
| | b3 s6 N-12 w-12 e=-3 =t=2 =e=1 =t1 * |
b9 h9 | #4 #-1 b2 N1 N-3 N1 N-4 | time.C
```

**Fig. 7.** An image synthesized from a pseudo-random annotation (denoted as *random*).

from the masks of their components (e.g., an eighth-note is extracted with its notehead, stem, and flag) through the music notation graphs and stored in a repository to be sampled from during later stages of the process. A total of 38,669 symbol instances were collected. Each symbol instance is then assigned anchor points (center of notehead, center of accidental, stem top) to control positioning on the rendered image. The acquisition can be constrained to the symbols of one writer to mimic the handwriting of a single person.

### 4.2 Obtaining Ground-Truth Annotations

*Mashcima* renders images based on music annotations in the *Mashcima* encoding. Two principled ways of obtaining such annotations are supported.

The first resort is to reuse existing annotations of real music. We leverage the converter from PrIMuS agnostic encoding to *Mashcima* encoding and use the PrIMuS dataset [6] as a source of such annotations. The entire dataset contains 87,678 music incipits, but only 64,127 of them (approx. 73%) can be successfully converted (due to symbols not supported by the current version of *Mashcima*, i.e., tuplets and multi-measure rests). Since PrIMuS contains no staccato dots, yet they are fairly abundant during evaluation, we converted about one half of the duration dots to staccato dots. Although this damages the musical meaning of (some) annotations, it allows recognition of staccato dots which would not be possible otherwise (adding training data without musical meaning actually helps the model, see Sect. 5). Apart from PrIMuS, additional annotations can be obtained, e.g., from RISM (the source of PrIMuS) and possibly other sources.

The second option is to generate the annotations by a stochastic process with only minimal constraints for valid token sequences (e.g., beams have to be connected, slur tags have to be paired). This approach is not expected to produce valid music, but this is not disqualifying. The amount of data that can be generated this way is basically unlimited and training on such noisy data can eventually improve the robustness of the model and its overall performance.

In our approach, most values (e.g., pitch) are drawn independently from a uniform distribution. For every image, the algorithm generates 5–15 token groups, where each group may be a key signature, a simple note, a beamed group, a barline, etc. The group distribution is chosen so that larger groups

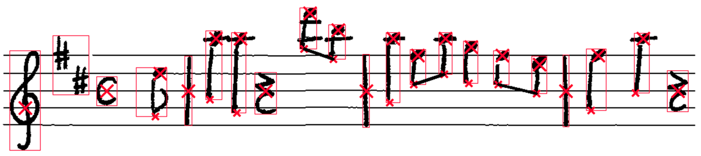


**Fig. 8.** Groups with bounding boxes. Large crosses mark symbol origins, smaller crosses mark where a stem ends and a beam attaches. A key signature has no anchor point.

(e.g., beamed groups) are less likely to appear. Each note (within a group) gets a random pitch and a set of ornaments and accidentals. Lastly, slurs are added (without crossing each other). An example of the result is displayed in Fig. 7.

## 4.3 Symbol Placement

Symbol placement begins by positioning an empty staff drawn from the symbol repository to an empty image. The staff is aligned horizontally, so the horizontal dimension of the image can serve directly as a temporal dimension and the vertical one for pitch. A table that maps pitch values to vertical pixel offset is precomputed for each staff/image. Symbols are then positioned by their anchor points onto the proper pitch offset with no variation. The random symbol selection already varies the vertical position well enough.

The input token sequence is then clustered into token groups of symbols that act as a single unit (e.g., a note with its accidentals and ornaments is a single group), however, many tokens are left alone in their group (rests, barlines). A mask is randomly chosen from the symbol repository for every symbol in the group. Stem orientation is determined based on the note pitch (with randomization around the center). All symbols within a group are positioned relative to each other (accidentals are placed in front of the notes, duration dots behind, staccato dots below). Ornaments are positioned with some random noise. The groups are then placed from left to right onto the staff with randomized padding.

When the placement is settled, the stem length for beamed notes is adjusted by scaling the stem masks vertically. This is the only place where masks are distorted. The beam is then positioned as a simple straight line of a fixed width (no masks used). Similarly, the slur positions are determined and then drawn as parabolic arcs. Each note has three anchor points where a slur can end: in front, after, and below.

## 4.4 Image Rendering

With all symbols positioned and adjusted, they are all drawn onto the staff image (without any distortion and scaling). The order of drawing does not matter since the image is binarized. An example of a final image with symbol bounding boxes and anchor points is shown in Fig. 8. The synthesizer is able to draw three staves onto one image, which further helps with mimicking a real cropped image (barlines can be rendered across multiple staves, see Fig. 9).

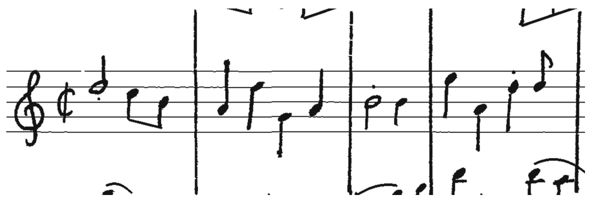


**Fig. 9.** Three staves synthesized in one image mimicking the look of a cropped image.

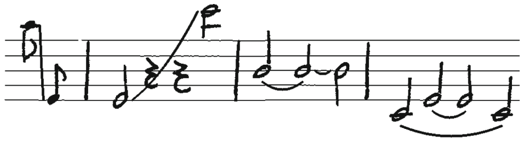
**Limitations.** The approach of simple positioning of masks cannot be applied to beams and slurs, so the system here fails to mimic the real world accurately (Fig. 10). There is also a lot of complexity regarding slur placement, that is not fully implemented in the current version of the synthesizer. The evaluation presented in the next section reveals that many errors are indeed related to slurs.

## 5 Experiments

This section presents the experiments evaluating how the proposed system is useful for synthesising HMR training data. We employ a state-of-the-art model trained on synthetic data generated by *Mashcima* and evaluate on real music sheets from MUSCIMA++ and a completely independent piece of written music.

### 5.1 Model Architecture

The neural network model employed in our experiments is inspired by the architecture proposed by Scheidl [25] for handwritten text recognition (the architecture used by Baró et al. [2] is not an alternative since it requires pixel-perfect annotation alignment that is not available in our data). The model is based on Convolutional Recurrent Neural Network (CRNN) with the Connectionist Temporal Classification (CTC) loss function [11,21]. In our model, the convolutional block is slightly modified by adding one layer and shifting the pooling parameters to preserve the temporal resolution. Inspired by Calvo-Zaragoza et al. [6], we also add dropout to the Bidirectional Long Short-Term Memory (BLSTM) layer to improve convergence.



```
e6 ( ) e-4 | h-4 ( qr qr ) h8 |
h0 ( ) h0 ( ) h0 | h-6 ( h-4 ( ) h-4 ) h-6
```

**Fig. 10.** Example of an incipit with misplaced slurs. Slurs do not affect positioning of neighbouring symbols and may improperly intersect them. Nested slurs are allowed.

**Table 1.** Model architecture overview.

| Layer | Shape | Note |
|---|---|---|
| Input | `w` × `64` × `1` | |
| Convolution | `w` × `64` × `16` | Kernel `5` × `5` |
| Max pooling | `w/2` × `32` × `16` | Stride `2,2` |
| Convolution | `w/2` × `32` × `32` | Kernel `5` × `5` |
| Max pooling | `w/4` × `16` × `32` | Stride `2,2` |
| Convolution | `w/4` × `16` × `64` | Kernel `5` × `5` |
| Max pooling | `w/4` × `8` × `64` | Stride `1,2` |
| Convolution | `w/4` × `8` × `128` | Kernel `3` × `3` |
| Max pooling | `w/4` × `4` × `128` | Stride `1,2` |
| Convolution | `w/4` × `4` × `128` | Kernel `3` × `3` |
| Max pooling | `w/4` × `2` × `128` | Stride `1,2` |
| Convolution | `w/4` × `2` × `256` | Kernel `3` × `3` |
| Max pooling | `w/4` × `1` × `256` | Stride `1,2` |
| Reshape | `w/4` × `256` | |
| BLSTM | `w/4` × `256` + `w/4` × `256` | Dropout |
| Concatenate | `w/4` × `512` | |
| Fully connected | `w/4` × `num_classes+1` | No activation function |
| CTC | ≤`w/4` | |

Our architecture is described in Table 1. The first convolutional layer accepts images with the height of 64 pixels and a variable width. The convolutional block then squishes the dimensions until the height is equal to 1. The next block is a bidirectional recurrent layer with dropout. Finally, CTC produces the output tokens in the *Mashcima* encoding. In each experiment, the model is trained using the adaptive learning rate optimizer (Adam) [15] with the learning rate equal to 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 10^{-8}$. For evaluation, the beam search decoding algorithm is used with a beam width of 100, while during training the greedy algorithm is applied (the beam width set to 1) [13].

## 5.2 Data

All training and validation data used in experiments are synthetic, generated by *Mashcima*. We experiment with two types of such data, generated from the two types of annotations described in Sect. 4.2: i) *realistic* data synthesized from the 64,127 PrIMuS incipit annotations converted to the *Mashcima* encoding, ii) *random* data of the same size, synthesized from pseudo-randomly generated annotations. In both cases, 1,000 instances are randomly selected and used as validation data and the remaining 63,127 instances comprise the training data.

Two datasets are used for evaluation: The *in-domain* set is an unseen subset of 17 pages from MUSCIMA++ containing monophonic music written by 6 writers. The number of pages per writer ranges from 1 to 5. Some pages contain

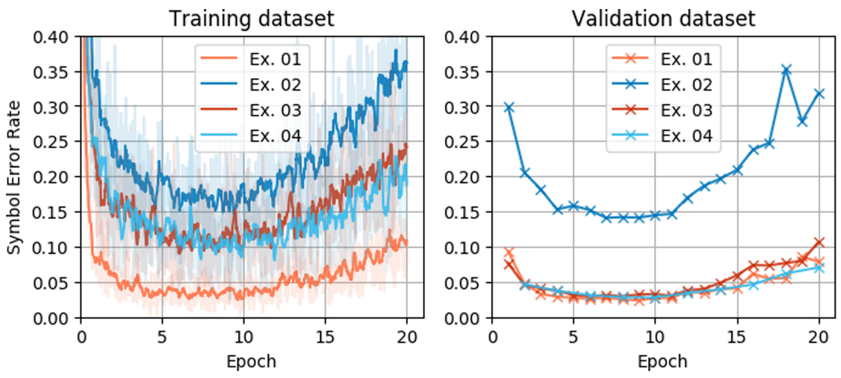


**Fig. 11.** SER on the training and validation sets throughout the training process: the training SER shown for each batch and smoothed over 11 values, the validation SER shown after each epoch (Ex. 4 has half as many epochs, each with twice as much data).

identical music but written by different writers.[6] This dataset contains 115 staves with 5,840 symbols (tokens) in total (50 symbols per staff on average). Images of these pages were split to single staves, each was manually annotated using the *Mashcima* encoding. This evaluation set is completely unseen – not only does the evaluation set contain no symbols from the training data, but there is also no intersection in terms of handwriting style and music content. However, we still (conservatively) consider this set in-domain (similar to the training data).

The *out-of-domain* set is created from a handwritten score of Cavatina by J. Raff for saxophone on three pages, which were scanned, binarized by a fixed threshold, and broken to images of single staves, each manually annotated in the *Mashcima* encoding (no additional preprocessing was performed). This evaluation set contains 35 staves and 1,831 symbols in total. We consider this evaluation set out-of-domain since there is no relation to the training data at all.

## 5.3 Experiment Design

We design four experiments to assess the effect of the two types of synthetic training data and their combination. In Experiment 1, the model is trained on the complete *realistic* data (63,127 instances). In Experiment 2, training is done on the *random* data of the same size. In Experiment 3, the model is trained on the same amount of data, while one half is a random sample from the *realistic* data and one half is sampled from the *random* data. Experiment 4 exploits both sets together doubling the training data size w.r.t. the previous experiments.

For quantitative evaluation, we employ Symbol Error Rate (SER) averaged over all evaluation instances. SER is the Levenshtein edit distance [17] normalized by the length of the ground-truth annotation. It was proposed for the evaluation of CRNN models for OMR in [6]. SER processes annotations at the token level with insertion, deletion, and substitution as elementary operations.

[6] Writers(pages): 13(2, 3, 16); 17(1); 20(2, 3, 16); 34(2, 3, 16); 41(2, 3, 16); 49(3, 5, 9, 11).

**Table 2.** Experiment results (%) averaged over three runs with standard deviation.

| Ex. | Training | Validation | Evaluation (SER) | |
|---|---|---|---|---|
| | | | *In-domain* | *Out-of-domain* |
| 1 | Full *realistic* | *Realistic* | 34.2 (±0.4) | 59.2 (±1.3) |
| 2 | Full *random* | *Random* | 28.0 (±1.4) | 58.7 (±1.6) |
| 3 | 1/2 *realistic* + 1/2 *random* | *Realistic* | **25.1** (**±1.2**) | **49.2** (**±2.0**) |
| 4 | Full *realistic* + full *random* | *Realistic* | 25.6 (±0.9) | 51.9 (±2.6) |

All models are trained for a maximum of 20 epochs, keeping the model with the lowest validation SER (usually from the 6th to 10th epoch, see Fig. 11).

### 5.4 Results and Analysis

Results of our experiments are displayed in Table 2. Overall, the results are very optimistic, especially given the fact that training is done on synthetic data only. The best SER on the *in-domain* evaluation set is achieved by Experiment 3 and is as low as 25.1%. Vaguely interpreted, this indicates that only about 25% of all symbols in the *in-domain* evaluation set are misrecognized.

Another interesting finding comes from the comparison of Experiments 1 and 2 on the *in-domain* evaluation set. The results surprisingly indicate that training on the *random* data is more effective than training on the *realistic* data (SER drops from 34.2% to 28.0%). The extensive variability of music symbols occurring chaotically in the *random* data (unconstrained by any musical/notation rules) seems more beneficial than learning from real melodies and valid notations.

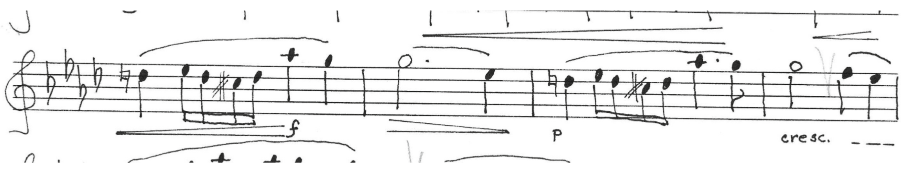
However, further comparison of Experiments 2 and 3 shows that the effect of training on the *realistic* data should not be neglected – if half of the *random* data in Experiment 2 is replaced by the same amount of the *realistic* data (Experiment 3), the SER on the *in-domain* evaluation set improves substantially (decreases from 28.0% to 25.1%). This suggests that seeing music symbols in their natural/meaningful configurations (think about clefs placed only at the beginning of scores, etc.) might be positive for learning.

Experiment 4 allows a direct comparison with all previous experiments and reveals additional findings. In comparison with Experiment 1, it quantifies the contribution of the *random* data added on top of the *realistic* data (-8.6% SER), which is much larger than the benefit of the *realistic* data added on top of the *random* data (−2.4% SER, cf. Experiment 2). In comparison with Experiment 3, it illustrates the effect of doubling the size of the training data, which is negligible (+0.5% SER). This suggests that enlarging the amount of training data of the same nature would not probably bring any major performance gains.

By comparing the *in-domain* and *out-of-domain* results, the model performs significantly worse on the *out-of-domain* data. This may be due to the fact that the *out-of-domain* data was preprocessed differently from the MUSCIMA++ data used for generating the training data. Cavatina contains faded text written onto

**Table 3.** Results achieved by Baró et al. [2] and our model from Experiment 4 on two sheets from MUSCIMA++. All scores are SER (%), however, not directly comparable.

| Page | Writer | Baró et al. [2] | | | Our method | |
|---|---|---|---|---|---|---|
| | | Rhythm | Pitch | Combined | #symbols | SER |
| 1 | 17 | 52.8 | 34.9 | 59.2 | 304 | 28.1 |
| 3 | 13 | 22.6 | 17.5 | 27.0 | 334 | 11.6 |



```
clef.G-2 b3 #3 N0 #2 N2 q2 . ( s=3 =s=2 #2 =t=1 =s2 q6 q5 | h5 * q3 |
N2 q2 s=1 =s=0 #1 =s=1 . =s2 q4 ( N5 e5 | h5 w-6 q4 . q3
```

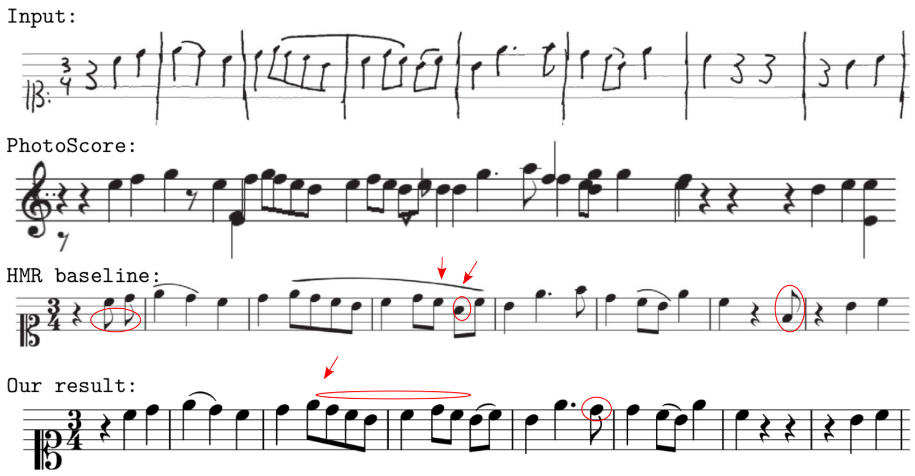
**Fig. 12.** Example from the *out-of-domain* evaluation data (SER on this staff is 49%).

the piece by a permanent marker, that cannot be fully removed by thresholding. Staff lines are also far thicker than in CVC-MUSCIMA and the model oftentimes confuses them with beams (also rendered as straight lines of the same width). Lastly, after binarization, the image is overall noisier (see Fig. 12).

### 5.5 Comparison with Other Works

Baró et al. [2] is the only work we can compare with. However, their method was not limited to monophonic music – the model was based on CRNN and did not use the CTC loss function, therefore the temporal resolution of their output is an order of magnitude higher than ours (since CTC collapses repeated tokens). Although their evaluation data overlaps in two pages (1 and 3) with our *in-domain* set and their evaluation was also done by SER, our scores (although appearing much higher, see Table 3) cannot be fairly compared to theirs.

The only possible direct comparison of their and our method is qualitative. Baró et al. [2] illustrated the effectiveness of their model on a certain image from MUSCIMA++ (page 3, writer 13). We recognized the same image by our model from Experiment 4 and display the results in Fig. 13 for direct comparison. The output of Baró et al. [2] contains four misrecognized notes and two missing slur endings (SER = 12.5%). Our model makes one error in note recognition and misses one slur beginning (causing the whole slur to disappear, SER = 4.2%).



```
clef.C-4 time.3 time.4 qr q3 q4 | q5 ( ) q4 q3 |
q4 e=5 =e=4 =e=3 =e2 | q3 e=4 ) =e3 e=2 ( ) =e3 |
q2 q5 * e4 | q4 e=3 ( ) =e2 q5 | q3 qr qr | qr q2 q3
```

**Fig. 13.** Qualitative comparison of our results to Baró et al. [2]. The first three staves are taken from Baró et al. [2]. The first staff is the input image from CVC-MUSCIMA (page 03, writer 13). The second staff is produced by PhotoScore. The third staff is the output of Baró et al. [2]. The last staff is the results of our model (Ex. 4) manually rendered by MuseScore (https://musescore.org/) followed by the output annotation in the *Mashcima* encoding.

## 6 Conclusion

We introduced the *Mashcima* system for creating synthetic handwritten music scores. It allows generating vast amounts of training data for HMR. Currently, HMR is mainly held back by the lack of training data. We were able to achieve state-of-the-art results by training on synthetic data generated by a relatively simple system. The MUSCIMA++ dataset is a step in the right direction. It provides rich annotations of music on various levels, from which any (simpler) encoding can be derived. However, its size is not sufficient for large experiments, which is understandable, given how much work such annotations take to produce. In *Mashcima*, MUSCIMA++ serves extremely well as a source of symbols for synthetic data generation. It is yet to be discovered what is the optimal mixture of random and realistic data to achieve even better results.

The *Mashcima* system still has limitations. Although it produces very real-world-looking data, the images are similar in style to those in the MUSCIMA++ dataset. The synthesized images are born binarized as they would have gone through the same binarization process as the original data from MUSCIMA++. Furthermore, the symbols and handwriting styles in the synthesized images are limited to the ones present in MUSCIMA++. In fact, *Mashcima* only enlarges existing datasets by reusing the existing symbols in new configurations and does not produce entirely new data. Altogether, this reduces the effectiveness of a model trained on such data and applied to data from a different source. This problem could be tackled by increasing the data variability, e.g., by additional distortion of symbol masks and/or collecting additional masks from other sources.

Our future work on *Mashcima* includes a more realistic rendering of slurs and beams, rendering of unsupported symbols, adding noise, and additional distortions of symbols and entire images. Ultimately, we aim at handling polyphonic music which would require changes in the HMR model architecture. This could be realized by producing more complex data with segmentation masks and relationship graphs as in MUSCIMA++. This would provide extensive amounts of data for many different HMR approaches to be trained and compared.

Despite the limitations of this first version, we believe the *Mashcima* system for handwritten musical score synthesis presents a valuable first-of-its-kind contribution to the field of OMR.

**Acknowledgment.** This work described in this paper has been supported by the Czech Science Foundation (grant no. 19-26934X), CELSA (project no. 19/018), and has been using data provided by the LINDAT/CLARIAH-CZ Research Infrastructure (https://lindat.cz), supported by the Ministry of Education, Youth and Sports of the Czech Republic (project no. LM2018101). The authors would like to thank Jan Hajič jr. for his valuable comments.

## References

1. Baró, A., Badal, C., Fornés, A.: Handwritten historical music recognition by sequence-to-sequence with attention mechanism. In: 17th International Conference on Frontiers in Handwriting Recognition (ICFHR), Dortmund, Germany, pp. 205–210 (2020)
2. Baró, A., Riba, P., Calvo-Zaragoza, J., Fornés, A.: From optical music recognition to handwritten music recognition: a baseline. Pattern Recogn. Lett. **123**, 1–8 (2019)
3. Baró, A., Riba, P., Fornés, A.: Towards the recognition of compound music notes in handwritten music scores. In: 15th International Conference on Frontiers in Handwriting Recognition (ICFHR), Shenzhen, China, pp. 465–470 (2016)
4. Calvo-Zaragoza, J., Castellanos, F., Vigliensoni, G., Fujinaga, I.: Deep neural networks for document processing of music score images. Appl. Sci. **8**(5), 654 (2018)
5. Calvo-Zaragoza, J., Hajič, J., Jr., Pacha, A.: Understanding optical music recognition. ACM Comput. Surv. **53**(4), 77 (2020)
6. Calvo-Zaragoza, J., Rizo, D.: End-to-end neural optical music recognition of monophonic scores. Appl. Sci. **8**(4), 606 (2018)
7. Calvo-Zaragoza, J., Toselli, A., Vidal, E.: Handwritten music recognition for mensural notation with convolutional recurrent neural networks. Pattern Recogn. Lett. **128**, 115–121 (2019)
8. Fornés, A., Dutta, A., Gordo, A., Lladós, J.: CVC-MUSCIMA: a ground truth of handwritten music score images for writer identification and staff removal. Int. J. Doc. Anal. Recogn. **15**, 243–251 (2011)
9. Fornés, A., Sánchez, G.: Analysis and recognition of music scores. In: Doermann, D., Tombre, K. (eds.) Handbook of Document Image Processing and Recognition, pp. 749–774. Springer, London (2014). https://doi.org/10.1007/978-0-85729-859-1_24

10. Good, M.: MusicXML: An internet-friendly format for sheet music. In: Proceedings of the XML Conference, Orlando, FL, USA, pp. 3–4 (2001)
11. Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification. In: Proceedings of the 23rd International Conference on Machine Learning (ICML), Pittsburgh, PA, USA, pp. 369–376 (2006)
12. Hajič, J., Jr., Pecina, P.: The MUSCIMA++ dataset for handwritten optical music recognition. In: 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, pp. 39–46 (2017)
13. Hwang, K., Sung, W.: Character-level incremental speech recognition with recurrent neural networks. In: IEEE International Conference on Acoustics. Speech and Signal Processing (ICASSP), Lujiazui, Shanghai, China, pp. 5335–5339 (2016)
14. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Synthetic data and artificial neural networks for natural scene text recognition (2014)
15. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations (ICLR), San Diego, USA (2014)
16. Krishnan, P., Jawahar, C.: Generating synthetic data for text recognition (2016)
17. Levenshtein, V.: Binary codes capable of correcting spurious insertions and deletions of ones. Probl. Inf. Transm. **1**, 8–17 (1965)
18. Pacha, A., Calvo-Zaragoza, J., Hajič, J., Jr.: Learning notation graph construction for full-pipeline optical music recognition. In: 20th International Society for Music Information Retrieval Conference (ISMIR), Delft, Netherlands, pp. 75–82 (2019)
19. Pacha, A., Choi, K.Y., Eidenberger, H., Ricquebourg, Y., Coüasnon, B., Zanibbi, R.: Handwritten music object detection: open issues and baseline results. In: 13th IAPR Interantional Workshop on Document Analysis Systems (DAS), Vienna, Austria, pp. 163–168 (2018)
20. Peng, X., Sun, B., Ali, K., Saenko, K.: Learning deep object detectors from 3D models. In: IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, pp. 1278–1286 (2015)
21. Puigcerver, J.: Are multidimensional recurrent layers really necessary for handwritten text recognition? In: 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, pp. 67–72 (2017)
22. Rebelo, A., Fujinaga, I., Paszkiewicz, F., Marçal, A., Guedes, C., Cardoso, J.: Optical music recognition: State-of-the-art and open issues. Int. J. Multimed. Inf. Retr. **1**, 173–190 (2012)
23. Roland, P.: The music encoding initiative (MEI). In: First International Conference on Musical Application Using XML, Milan, Italy, pp. 55–59 (2002)
24. Rothstein, J.: MIDI: A Comprehensive Introduction, vol. 7. AR Editions, Inc. (1992)
25. Scheidl, H.: Handwritten text recognition in historical documents. Master's thesis, Vienna University of Technology (2018)
26. Tuggener, L., Elezi, I., Schmidhuber, J., Pelillo, M., Stadelmann, T.: DeepScores - A dataset for segmentation, detection and classification of tiny objects. In: 24th International Conference on Pattern Recognition (ICPR), Beijing, China, pp. 3704–3709 (2018)
27. Tuggener, L., Elezi, I., Schmidhuber, J., Stadelmann, T.: Deep watershed detector for music object recognition. In: Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR), Paris, France, pp. 271–278 (2018)
28. van der Wel, E., Ullrich, K.: Optical music recognition with convolutional sequence-to-sequence models. In: Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR), Suzhou, China, pp. 731–737 (2017)