GAN-based Content-Conditioned Generation of Handwritten Musical Symbols

Gerard Asbert[1 [0009−0005−3316−5463]], Pau Torras[1,2 [0000−0003−0327−9046]], Lei Kang[1,2 [0000−0002−1962−3916]], Alicia Fornés[1,2 [0000−0002−9692−5336]], and Josep Lladós[1,2 [0000−0002−4533−4739]]

[1] Computer Vision Center, Barcelona, Spain {gasbert, ptorras, lkang}@cvc.uab.cat {afornes, josep}@cvc.uab.es
[2] Department of Computer Science, Universitat Autònoma de Barcelona, Spain

Abstract. The field of Optical Music Recognition (OMR) is currently hindered by the scarcity of real annotated data, particularly when dealing with handwritten historical musical scores. In similar fields, such as Handwritten Text Recognition, it was proven that synthetic examples produced with image generation techniques could help to train better-performing recognition architectures. This study explores the generation of realistic, handwritten-looking scores by implementing a music symbol-level Generative Adversarial Network (GAN) and assembling its output into a full score using the Smashcima engraving software. We have systematically evaluated the visual fidelity of these generated samples, concluding that the generated symbols exhibit a high degree of realism, marking significant progress in synthetic score generation.

Keywords: Generative Adversarial Networks · Handwritten Musical Symbols · Content Conditioning · Musical Score Generation

1 Introduction

Optical Music Recognition (OMR) is formally defined as the task of computationally reading music notation in documents [4], whether printed or handwritten. From a research point of view, one of the main goals of OMR is constructing faithful computer-processable representations of these music scores, which would enable their study by scholars and their conservation by archivists. Having computers perform this task is interesting because manual transcription of musical documents is a painstaking, costly process that requires many hours, even for relatively simple scores and seasoned musicologists.

The complexity of this task highly increases when addressing historical handwritten scores, particularly those written in CWMN (Common Western Music Notation) from the 18th century onward. Thousands of these musical scores, are scattered throughout the world, only a small percentage of which has been transcribed. This scarcity of data added to the presence of artifacts due to the aging of the paper plummet the performance of OMR models. Multiple authors [4, 25] have stated that the scarcity of annotated handwritten musical data is one of

the biggest bottlenecks to the improvement of the field. Many have resorted to using printed musical data as a workaround, which can provide an improvement in model performance [1]. However, it remains inferior to the value offered by real handwritten data. Therefore, the generation of synthetic handwritten data seems a viable option to explore.

For this reason, the aim of this work is to design an effective and robust generative architecture that can generate synthetic handwritten scores to increase the training set for OMR systems, and thus, improve their overall performance. Concretely, we propose to generate isolated symbols using Generative Adversarial Networks (GAN) [9], and use the Smashcima engraving software [18] to create full music sheets. From the extensive evaluation, we observe that the resulting music scores are realistic, paving the way to the use of generated music data for training OMR systems.

As for the structure followed in this work, after outlining the research context and motivations, we review prior work in OMR, handwriting generation and music symbol synthesis. The methodology section presents our adapted Generative Adversarial Network (GAN) architecture, followed by a description of the dataset composition and preprocessing techniques. The experimental setup is then detailed, including hyperparameter selection and training strategies. Finally, evaluation is conducted through qualitative visual assessments and quantitative metrics such as Fréchet Inception Distance, Kernel Inception Distance, and Handwriting Distance.

2 Previous Work

Early OMR systems relied on classical computer vision techniques to extract simple symbol primitives from images [8, 21, 23]. While these methods gave acceptable results for closed domains, the use of deep learning models significantly improved recognition accuracy and generalization on many sub-tasks [4]. Unfortunately, implementing large-scale learning algorithms is complicated in the field of OMR because of the requirement of large amounts of annotated data, which are not quite there yet [4,25]. Consequently, it has become necessary to explore alternative strategies to obtain samples to train models on, including data augmentation, synthetic dataset generation, and transfer learning from printed data.

Despite the fact that data augmentation [23] can mitigate overfitting, it is inherently limited to producing variations within the original feature space, being unable to generate entirely new samples. The creation and popularization of generative deep learning models surpassed this limitation, as it enables to generate realistic samples and not as resembling of the input image. Among the predominant generative models developed during the past decade, including Generative Adversarial Networks (GANs) [9], Diffusion Models (DFs) [13], Variational Autoencoders (VAEs) [16], and Transformers [3, 19], GAN architectures have been among the most popular in the field of Handwritten Text Recognition (HTR). Indeed, HTR is a much more developed field than OMR, especially in the

generation of synthetic data. Numerous studies [7, 14, 15, 26, 27] have leveraged Generative Adversarial Networks (GANs) for synthesizing handwritten words, benefiting from the extensive availability of training data—far exceeding the datasets available for handwritten OMR. In this context, the work of Kang *et al.* [15] was among the first to apply a GAN architecture in the field of HTR, yet it managed to generate highly realistic handwritten text, and thus showing a promising research direction for related domains, such as handwritten music symbols.

However, very few of the above mentioned techniques have been explored in the context of Optical Music Recognition. VAEs [16] and Adversarial Autoencoders (AAEs) [17] have been tested in [11] showing acceptable results. Also in the field of OMR, Shatri et al. [22] and Tirupati et al. [24] propose the use of GANs for line-level music scores. In both works, the generated images display notable realism, however, there is still room for improvement, especially in the generation of small and detailed symbols, such as accidentals and rests. Finally, it is worth to mention that there are other types of contributions to the field of data generation that do not necessarily involve deep learning models. The Smashcima software [18], created by Mayer and Pecina, consists of an architecture that takes existing individual music symbols and engraves them into realistic music sheets. However, it does not generate new music symbols, which means that, in terms of handwriting style variability, it is limited to the styles of the existing set of music symbols.

In this work, and inspired by the above findings, we propose to combine the benefits of GANs and the Smashcima software to generate more realistic and varied music scores. Concretely, we propose a content-conditioned GAN architecture to generate music symbols to be fed into the Smashcima software. Thus, we can generate music scores without limitations regarding contents and handwriting styles while avoiding the need of well-aligned datasets for training – we can rely on symbol-level annotations from widespread datasets instead, which are easier to obtain, and generate smaller and more controlled classes, which is computationally more efficient.

3 Methodology

3.1 Preliminaries

The architecture of a basic GAN [9] comprises two main networks: The Generator $G(\boldsymbol{z}; \boldsymbol{\theta_g})$ and the Discriminator $D(\boldsymbol{x}; \boldsymbol{\theta_d})$, which are differentiable functions parameterized by some learnable weights $\boldsymbol{\theta}_{[.]}$. The Generator produces an image conditioned by some noise vector $\boldsymbol{z}$, while the Discriminator computes the probability that an image $\boldsymbol{x}$ is a real sample or is produced by the Generator. These two networks' parameters are optimized jointly by minimizing the discriminator loss

$$\mathcal{L}_d = \log(1 - D(G(\boldsymbol{z}))). \tag{1}$$

Essentially, training follows an adversarial process: the Discriminator minimizes its classification error, while the Generator maximizes it, aiming to generate images indistinguishable from real ones.

This min-max optimization has its benefits, as it fosters continuous improvement in both networks while not relying on a pixel-level loss function. Nevertheless, it also has its drawbacks, the most prominent being, that balancing the simultaneous learning of both architectures can be difficult – if one network significantly outperforms the other, the adversarial dynamic collapses. This scenario may occur if the Discriminator quickly reaches near-perfect accuracy, which can lead to vanishing gradients for the Generator. Alternatively, if the Generator produces non-realistic samples while the Discriminator has not learned to distinguish them correctly, the Discriminator's loss quickly saturates and stops providing useful feedback.

3.2 Baseline Model

We base our work on the GANWriting model by Kang *et al.* [15]. This model expands on the basic premise of a GAN, adapting it to the idiosyncrasy and requirements of handwriting generation. The primary distinction from a conventional GAN is the ability to condition the output to adapt to a specific writing style. Rather than using a random vector as input, the Generator receives an encoded representation of a set of images from the same author, forcing the GAN to learn their specific calligraphic style. The secondary main distinction is that the model can be conditioned to generate images with a specific textual content. Additionally, additive noise is applied to the feature space representing the calligraphic style before concatenating the two content encoding vectors, forcing the appearance of alterations on the output.

To support this finer-grained generation process, the model employs three distinct loss functions to guide the training process. The first one is the Discriminator Loss from standard GANs. The second one is the Word Recognizer Loss, which is computed by passing the generated image through a word recognition model, encouraging the Generator to produce images that accurately contain the word specified in the input string. Lastly, the Writer Classifier Loss incentivizes the Generator to generate images that exhibit a stylistic resemblance to the handwriting of the specified author from the input set of images.

3.3 Adaptation to Music Symbol Generation

Our architecture, as can be observed in Fig. 1, replicates the content conditioning in [15] with some necessary changes. Given that writer identification data is not available for all of the datasets in use, we restrict the style input to a single image. Furthermore, the content conditioning sequence is replaced by a single one-hot encoded vector, given that individual symbols are generated on every run instead of full words. These changes imply the modification of the loss function of the model w.r.t. the baseline implementation and thus the modification of specific components of the model.

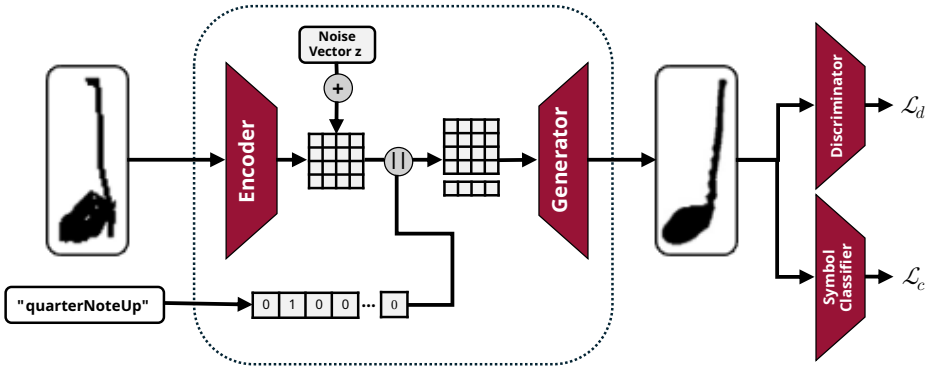

Fig. 1. Architecture of the proposed handwritten musical symbol generation model. The input to the generator is a style example and the symbol class encoded as a one-hot encoded vector. The architecture is trained combining a standard GAN discriminator loss and a symbol classification loss, each produced by two sub-networks.

We employ the same Discriminator network proposed in [15]. This architecture consists of an initial convolutional layer, followed by six residual blocks with Leaky ReLU activations and Average Pooling. A final binary classification layer is used to distinguish between real and generated images. The standard GAN Discriminator loss $\mathcal{L}_d$ is computed from this module.

For the content loss, the text recognition network is replaced by a CNN-based symbol classifier $C(\boldsymbol{x}; \boldsymbol{\theta_c})$, which consists of three convolutional layers with batch normalization. Each of these blocks is in turn followed by ReLU activation and max pooling to progressively downsample the spatial dimensions while increasing feature depth. The extracted feature maps are flattened and fed into a final classification layer that outputs predictions over our vocabulary of musical symbols. Denoting the final predicted distribution over the output class vocabulary V as $C(\boldsymbol{x}) = \hat{y}$ and the ground truth distribution as y, the final Classification loss $\mathcal{L}_c$ is formally defined as

$$\mathcal{L}_c = \mathrm{KL}(\hat{y}, y) = \sum_{\forall v \in V} y(v) \cdot \log \frac{y(v)}{\hat{y}(v)}. \tag{2}$$

A sweep was performed to test if the above specified Kullback–Leibler divergence loss was the right choice compared to the usual cross-entropy loss, for which the former gave slightly better results.

Given that there is no writer identification ground truth in all of the training datasets, the writer identification loss is not used for this model. The final loss function for the full model is thus

$$\mathcal{L}_{\text{Model}} = \alpha \mathcal{L}_d + \beta \mathcal{L}_c, \tag{3}$$

where both α and β are hyperparameters that allow weighting the importance of each of the sub-modules.

4 Datasets

The data used for training the GAN consists of images of real handwritten musical symbols, which have been extracted from multiple sources:

MUSCIMA++ [10]:. A collection of 140 page-level modern handwritten score samples with pixel-level symbol annotations. It contains examples of most classes.

Fornes Dataset [8]:. A symbol-level dataset containing 2128 examples of clefs and 1970 examples of accidentals. Originally intended for classification.

Homus [5]:. A broad "online" (as opposed to rasterized; containing stroke information) symbol database containing 15400 samples authored by 100 different musicians. It contains examples of most classes.

Capitan Collection [6]:. A collection of symbols extracted by tracing real mensural scores. Some of the classes are compatible with CWMN.

The combination of these datasets is not straightforward, given that their symbol definitions are not directly compatible. A pre-processing step is performed to address inconsistencies in naming conventions for objects pertaining to the same class (e.g. `Eight-Rest` from Homus or `Rest8th` from MUSCIMA++ to a generic `eightrest` class) and to address granularity differences between the classes of the datasets (e.g. converting objects in the `halfNote` class into `halfNoteUp` and `halfNoteDown`).

Due to the scarcity of samples and high imbalance found in the initial class distribution, data augmentation techniques are applied on the training symbol dataset. New images are created by rotating dataset samples by ± 10 or mirror flipping them in those cases where semantic integrity is preserved, thus re-balancing some of the underrepresented classes. A threshold of 3,000 images per class is set to ensure an adequate amount of training data for reliable model performance, resulting in a final collection of 49 classes.

Additional line-level samples are required in order to evaluate the full generation pipeline – generation of symbols and engraving. The MUSCIMA++ dataset is used as gold standard, since it is the only one of the sourced datasets that contains full-line images as well as symbol annotations. This ensures that comparison is performed against samples that are reasonably in-distribution.

5 Experimental Setup

5.1 Hyperparameter Tuning

The learning rates for the discriminator, generator, and recognizer are set to 1e-5, 1e-4, and 1e-5, respectively, based on optimal performance observed during a hyperparameter sweep. We employ a batch size of 16, chosen to balance computational efficiency with stable gradient updates.

To regulate training dynamics, we apply different weighting factors to the loss components. The discriminator loss is weighted at $\alpha = 1.0$, while the recognizer loss is assigned a weight of $\beta = 2.5$ to emphasize accurate classification of generated symbols. Additionally, a noise penalty term with a weight of 3.0

is introduced to encourage diversity in the generated samples, mitigating mode collapse.

5.2 Additional Modifications

Applying the described GAN architecture to handwritten data produces already acceptable synthetic symbols, as shown in Fig. 2.a. Nevertheless, certain generated examples, as illustrated in Fig. 2.b, display some issues that require further refinement.

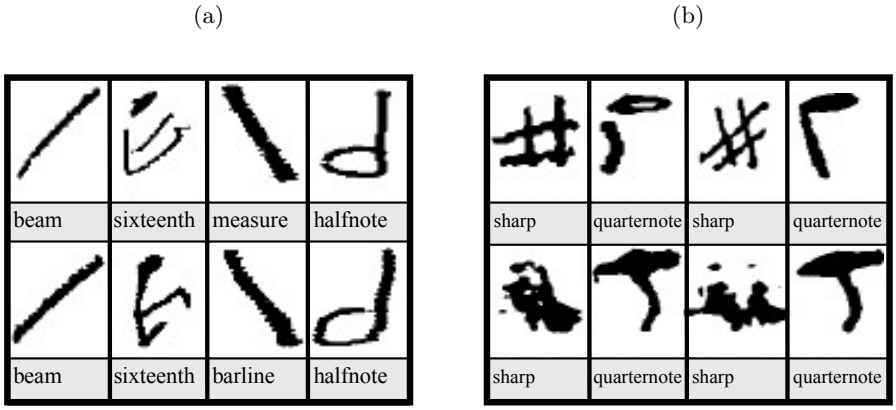

Fig. 2. (a) An example of an average generation. (b) An example of a faulty generation. Top row: Input symbols | Bottom row: Generated symbols.

The primary challenge is the GAN's difficulty to generate handwritten-looking images for the more complex symbols, concretely the `accidentalSharp` and `gClef` classes. As it can be observed in Fig. 2.b, results often do not resemble the input symbol. Our current hypothesis suggests that this issue is caused by the extreme variability in the handwritten representations of these symbols, which may hinder the model's ability to learn a consistent mapping.

To address this issue, two potential solutions are explored. The first solution involves implementing symbol-specific training steps. After 150 standard steps – that is, where each batch contains randomly selected symbols – the GAN undergoes 50 steps where the batch consists exclusively of the symbols that prove more challenging to generate, allowing for a more focused training on them.

The second solution involves the symbol classifier and its corresponding loss. The issue arises from the fact that the symbol classifier correctly labels disorted symbols because they resemble the real symbol class more closely than the other classes. Consequently, the loss does not reflect the presence of distortions, preventing the GAN from learning to correct them. Our hypothesis suggests that the

classifier needs to be more stringent, as a result, several pictures of the distorted symbols were assigned to new "bad" classes (in this case, `accidentalSharpBad` and `gClefBad`). This adjustment causes the classifier loss to be affected when a deformed symbol is generated, as these images are now classified under the "bad" classes rather than their correct counterparts.

A second issue is evident in Fig. 2.b, where, despite significant variation in the input images of the quarter notes, the generated synthetic images exhibit a high degree of similarity. This behavior suggests that the GAN is excessively dependent on the input text label, leading to nearly equivalent outputs for each instance of a given class while neglecting the style reference image. To address this, we introduce a random swap of a subset of the input labels within a batch. This strategy reduces the GAN's reliance on the input label, as it may occasionally be inaccurate, and encourages the model to focus more on the input image. As a result, this modification enhances the variability of generated instances within the same symbol class.

As the GAN continues its training process, determining the best time to freeze its weights and save the synthetic symbols to put them into musical staves is required. A first approach to automate this process consists on comparing the encoded feature vectors of the input and generated images using Euclidean Distance and Cosine Similarity. However, using this decision criterion alone, the generated images do not conform to desired quality standards. To solve this, the Structural Similarity Index Metric (SSIM) is additionally incorporated, which considers structural information, luminance, and contrast to better align with human visual perception. The combination of these metrics allows saving models that generate higher-quality images.

5.3 Line-level Generation

Once the synthetic music symbol images are generated, they need to be placed onto music sheets for testing purposes. This task is accomplished using the Smashcima software [18], which operates as follows: first, the user provides a MusicXML file, which describes the semantics of the music sheet to engrave. Smashcima then collects musical symbol samples from the MUSCIMA++ dataset, as well as their masks, bounding boxes and other positional parameters. The software then places the symbols on the score according to the MusicXML file, resizing and merging them accordingly. For this work, the Smashcima is modified to take the symbol-level images generated by our system rather than the data from MUSCIMA++.

5.4 Considered Metrics

For the evaluation of the synthetic musical lines containing our symbols, we utilize the methodology proposed by Pippi et al. [20], which provides a tool for comparing two image datasets—music sheets at the line level in our case—by computing multiple metrics in their latent spaces. The following metrics are

selected to quantitatively assess the realism of the generated data. Note that in all of these metrics, the lower the value the better.

Fréchet Inception Distance (FID) [12]. It measures the similarity between two datasets by extracting feature embeddings from a pretrained InceptionV3 model and computing the Fréchet distance between their distributions, assuming a Gaussian form.

Kernel Inception Distance (KID) [2]. Extracts feature embeddings using InceptionV3, like FID, but measures similarity with Maximum Mean Discrepancy (MMD) using a polynomial kernel. Unlike FID, KID does not assume a Gaussian distribution, making it more robust to different data distributions within the data.

Handwriting Distance (HWD) [20]. A metric for Handwritten Text Generation (HTG) that uses a deep network to extract handwriting style features and compute perceptual distance between styles.

To evaluate these metrics, we conducted a comparative analysis of our synthetic lines against multiple reference datasets. Specifically, we compared them to lines generated using the base Smashcima software with Muscima++ symbols, real historical lines, and printed lines, all benchmarked against the Muscima++ dataset.

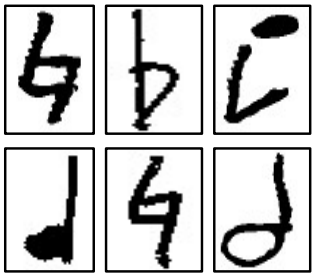

Fig. 3. Mix of real and synthetic symbols. Check footnote [3] to know which ones are generated.

6 Evaluation

In this section, we evaluate our GAN-generated symbols and staves, both qualitatively and quantitatively.

6.1 Qualitative Results

For the qualitative analysis, we first evaluate the individual handwritten symbols, followed by an assessment of the lines engraved using these synthetic symbols.

As illustrated in Fig. 2.a, the majority of the generated symbols are realistic, recognizable, and visually appealing. To further demonstrate this, a mix of real and synthetic symbols is shown in Fig. 3. The generated symbols are revealed in the footnote [3].

However, certain symbols remain challenging for the GAN to reproduce accurately, particularly the `accidentalSharp` and `gClef` classes. Nevertheless, as shown in Fig. 4, while not of perfect quality, the generated instances start to ressemble more the real symbols.

Regarding the staves generated using our synthetic symbols, some examples are shown in Fig. 5. Overall,the backgrounds utilized by Smashcima effectively replicate various types of aged paper, while the connection between primitives is well-executed.

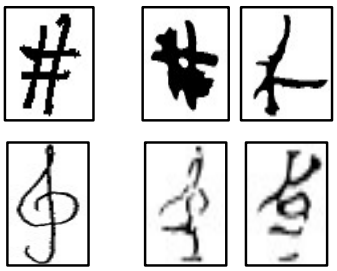

Fig. 4. On the left, reference images of an `accidentalSharp` and a `gClef` from the input dataset. On the right, faulty images generated by the GAN of each respective class.

Some comparison can be drawn between our approach and that of Tirupati *et al.* [24], since both methods employ GANs in different ways for music score generation. One of the key strengths of their work is its ability to closely replicate the musical structure from the input images. However, this also introduces a limitation, as the generated scores closely adhere to the style of the original input music sheet, restricting stylistic variability and the model's freedom to generate interesting handwriting artifacts. We believe that an ideal synthetic music score generation system should allow for the creation of any music sheet, in the style of any specific composer while maintaining a high degree of realism. Our proposed method, which first generates the primitives and then arranges them on the page, has the advantage of letting us control the musical structure without being limited by the input data.

[3] Images generated by the GAN: top-left, top-right, bottom-right.

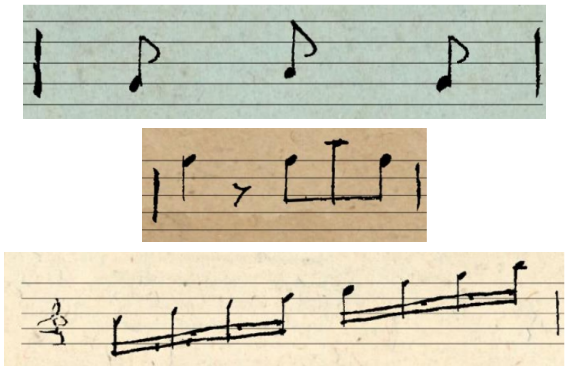

Fig. 5. Musical sheets made with the symbols generated by the GAN, then cut into a measure and cropped vertically.

6.2 Quantitative Results

Following a visual assessment of the generated musical lines, their accuracy was quantitatively evaluated by comparing them to real lines extracted from the MUSCIMA++ dataset. In addition to our proposed dataset – referred to as `GAN Smashcima` – several other line-level datasets were included for comparison. These comprise `Base Smashcima`, which contains lines generated using the original Smashcima software based on MUSCIMA++ symbol masks; `Real`, which includes authentic historical manuscript lines; and `Printed`, consisting of typeset musical lines derived from printed scores. All datasets were binarized prior to comparison to ensure that evaluations focused exclusively on the stroke and structural form of the symbols, rather than on background similarity.

	FID ↓	**KID ↓**	**HWD ↓**
Printed	118.96	0.114	2.194
Real	136.82	0.117	2.32
Base Smashcima	132.38	0.128	3.052
GAN Smashcima	142.48	0.150	2.85
MUSCIMA++ (Self)	**0.033**	**-1.06**	**0.0**

Table 1. Fréchet Inception Distance (FID), Kernel Inception Distance (KID), and Handwriting Distance (HWD) between various datasets and the MUSCIMA++ dataset as reference, which consists of real handwritten lines.

Table 1 reveals that the dataset most similar to MUSCIMA++ across all metrics is the printed dataset. According to the FID metric, Base Smashcima follows, which is consistent given that its symbols originate from MUSCIMA++. The real dataset ranks next, as expected, since it consists of human-written symbols. Finally, the dataset generated in this work achieves an FID score close to the real dataset, which indicates that the global structure of our images closely resembles MUSCIMA++.

The KID metric, also in Table 1, exhibits a similar ranking, with the only difference being that real lines outperform those from Base Smashcima. This may be attributed to the greater stability of the KID metric with respect to the number of samples, given that the number of lines generated for this test is approximately in the thousands.

For the HWD metric, illustrated in the last chart in Table 1, the printed and real datasets again yield the best results. Notably, our generated symbols score higher in handwriting similarity than Base Smashcima, despite the latter being derived from MUSCIMA++. This suggests that, in the absence of real data, our synthetic symbols may serve as a viable alternative for augmenting datasets.

To better contextualize the significance of the differences observed in the evaluation metrics, we established a baseline by comparing the Muscima++ dataset against itself. As expected, the HWD metric yielded a score of 0.0, indicating perfect similarity. The FID metric produced a low score of 0.033, while the KID metric resulted in a slightly negative value of -1.06. Although the KID metric is theoretically non-negative—as it estimates a squared distance—it is an unbiased estimator and can exhibit slight negative values in practice due to variance introduced by finite sample sizes.

7 Conclusions

This work has proposed a GAN architecture for generating synthetic handwritten musical symbols. We have demonstrated that, even in a field characterized by limited and complex structured data, a GAN-based model can produce promising results, offering a new avenue for providing training data for OMR. Furthermore, we have explored the integration of the Smashcima software for engraving synthetic symbols, which proved effective in generating realistic music scores while allowing precise control over the desired musical content.

Our pipeline opens several possibilities for future research. First, by leveraging our GAN-generated symbols, researchers can explore other techniques for arranging them into realistic sheets. Second, different generative architectures could be tested, such as Diffusion Models or Transformers, which have not been explored in the field of OMR yet, and then integrated into our existing pipeline, and using the final processing stage to structure the generated symbols into coherent musical notation.

Despite the promising results, opportunities for further improvement remain. The generation of some symbols (i.e. clef and accidental sharp) is challenging due to their inherent complexity and the high handwriting style variability. This

could be addressed by making the GAN learn the handwritten style of a specific author, whenever data of distinct writers may be available. Another limitation is the lack of symbol classes that the Smashcima software accepts and includes in the generated scores. If the number of classes accepted were to be increased, more complex music sheets could be generated, and therefore, be able to get closer to the feature representation of the real data.

Addressing these challenges could bring us closer to high-quality synthetic handwritten notation. As generative models and domain adaptation advance, synthetic data is likely to play a crucial role in bridging the gap between handwritten and machine-readable notation.

Acknowledgments. This work has been partially supported by the Spanish projects CNS2022-135947 (DOLORES) and PID2021-126808OB-I00 (GRAIL) from the Ministerio de Ciencia e Innovación, and the grant Càtedra ENIA UAB-Cruïlla (TSI-100929-2023- 2) from the Ministry of Economic Affairs and Digital Transition of Spain. Pau Torras is funded by the Spanish FPU Grant FPU22/00207.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

Code Availability
Link to the GAN code: `https://github.com/GerardAsbert/CVC_OMR-GAN`.
Link to the custom Smashcima code: `https://github.com/GerardAsbert/Smashcima`

References

1. Baró, A., Badal, C., Fornês, A.: Handwritten Historical Music Recognition by Sequence-to-Sequence with Attention Mechanism. In: 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 205–210 (Sep 2020). https://doi.org/10.1109/ICFHR2020.2020.00046
2. Bińkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying mmd gans. arXiv e-prints pp. arXiv–1801 (2018)
3. Bond-Taylor, S., Leach, A., Long, Y., Willcocks, C.G.: Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. CoRR **abs/2103.04922** (2021), `https://arxiv.org/abs/2103.04922`
4. Calvo-Zaragoza, J., Jr, J.H., Pacha, A.: Understanding optical music recognition. ACM Computing Surveys (CSUR) **53**(4), 1–35 (2020)
5. Calvo-Zaragoza, J., Oncina, J.: Recognition of pen-based music notation: the homus dataset. In: 2014 22nd International Conference on Pattern Recognition. pp. 3038–3043. IEEE (2014)
6. Calvo-Zaragoza, J., Rizo, D., Quereda, J.M.I.: Two (note) heads are better than one: Pen-based multimodal interaction with music scores. In: ISMIR. pp. 509–514 (2016)
7. Elanwar, R., Betke, M.: Generative adversarial networks for handwriting image generation: a review. The Visual Computer pp. 1–24 (2024)
8. Fornés, A., Lladós, J., Sánchez, G.: Old handwritten musical symbol classification by a dynamic time warping based method. In: International workshop on graphics recognition. pp. 51–60. Springer (2007)

9. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. Commun. ACM **63**(11), 139–144 (Oct 2020). https://doi.org/10.1145/3422622
10. Hajič, J., Pecina, P.: The muscima++ dataset for handwritten optical music recognition. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). vol. 1, pp. 39–46. IEEE (2017)
11. Havelka, J., Mayer, J., Pecina, P.: Symbol generation via autoencoders for handwritten music synthesis. In: Proc. 5th Int. Workshop on Reading Music Systems. p. 20 (2023)
12. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017)
13. Ho, J., Jain, A., Abbeel, P.: Denoising Diffusion Probabilistic Models. In: Advances in Neural Information Processing Systems. vol. 33, pp. 6840–6851. Curran Associates, Inc. (2020)
14. Kang, L., Riba, P., Rusiñol, M., Fornés, A., Villegas, M.: Content and Style Aware Generation of Text-Line Images for Handwriting Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(12), 8846–8860 (Dec 2022). https://doi.org/10.1109/TPAMI.2021.3122572
15. Kang, L., Riba, P., Wang, Y., Rusinol, M., Fornés, A., Villegas, M.: Ganwriting: content-conditioned generation of styled handwritten word images. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16. pp. 273–289. Springer (2020)
16. Kingma, D.P., Welling, M.: Auto-Encoding Variational Bayes (Dec 2022). https://doi.org/10.48550/arXiv.1312.6114
17. Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., Frey, B.: Adversarial Autoencoders. arXiv preprint arXiv:1511.05644 (arXiv:1511.05644) (May 2016). https://doi.org/10.48550/arXiv.1511.05644
18. Mayer, J., Pecina, P.: Synthesizing training data for handwritten music recognition. In: International Conference on Document Analysis and Recognition. pp. 626–641. Springer (2021)
19. Mehmood, R., Bashir, R., Giri, K.: Deep generative models: A review. Indian Journal Of Science And Technology **16**, 460–467 (02 2023). https://doi.org/10.17485/IJST/v16i7.2296
20. Pippi, V., Quattrini, F., Cascianelli, S., Cucchiara, R.: Hwd: A novel evaluation score for styled handwritten text generation. arXiv e-prints pp. arXiv–2310 (2023)
21. Rebelo, A., Fujinaga, I., Paszkiewicz, F., Marcal, A.R., Guedes, C., Cardoso, J.S.: Optical music recognition: state-of-the-art and open issues. International Journal of Multimedia Information Retrieval **1**, 173–190 (2012)
22. Shatri, E., Palavala, K.R., Fazekas, G.: Synthesising handwritten music with gans: A comprehensive evaluation of cyclewgan, progan, and dcgan. In: 2024 IEEE International Conference on Big Data (BigData). pp. 3208–3217. IEEE (2024)
23. Tardón, L.J., Sammartino, S., Barbancho, I., Gómez, V., Oliver, A.: Optical music recognition for scores written in white mensural notation. EURASIP Journal on Image and Video Processing **2009**, 1–23 (2009)
24. Tirupati, N., Shatri, E., Fazekas, G., et al.: Crafting handwritten notations: Towards sheet music generation. In: Proc. 5th Int. Workshop on Reading Music Systems (2024)
25. Torras, P., Biswas, S., Fornés, A.: A unified representation framework for the evaluation of optical music recognition systems. International Journal on Document Analysis and Recognition (IJDAR) **27**(3), 379–393 (2024)

26. Zdenek, J., Nakayama, H.: JokerGAN: Memory-Efficient Model for Handwritten Text Generation with Text Line Awareness. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 5655–5663. MM '21, Association for Computing Machinery, New York, NY, USA (Oct 2021). https://doi.org/10.1145/3474085.3475713
27. Zdenek, Jan, Nakayama, Hideki: Handwritten text generation with character-specific encoding for style imitation. In: Fink, G.A., Jain, R., Kise, K., Zanibbi, R. (eds.) Document Analysis and Recognition - ICDAR 2023. pp. 313–329. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-41679-8_18