LEGATO 2: Toward Multimodal Sheet Music Recognition and Understanding

Guang Yang[♠*] **Brian Siyuan Zheng**[♠*] **Victoria Ebert**[♠] **Noah A. Smith**[♠,♣]
[♠]Paul G. Allen School of Computer Science & Engineering, University of Washington
[♣]Allen Institute for AI
{gyang1, nasmith}@cs.washington.edu

Abstract

We propose a novel pipeline, Legato 2, for extracting symbolic notation and semantic knowledge from images of sheet music. Legato 2 features the first large-scale neural model for optical music recognition (OMR) to operate sequentially on a system-by-system basis, following the horizontal lines of notation as they are read on the page, rather than treating the page as an undifferentiated image, enabling better scaling to arbitrarily long inputs. It is also the first OMR model capable of generating symbolic transcriptions that include embedded textual content, such as titles and annotations. The pipeline combines system-level segmentation with an autoregressive vision-LM to capture both local notation details and score structure. Across multiple datasets, Legato 2 consistently outperforms prior state of the art. We also show that symbolic transcriptions complement visual inputs for frontier language models, improving their interpretation of dense musical documents. Legato 2 establishes new state-of-the-art performance in both OMR and downstream sheet music understanding.

1 Introduction

Written music is central to many musical traditions as a medium of creation, transmission, study, and performance. In traditions where sheet music is an authoritative artifact,[1] musicians cultivate a specifically *visual* form of musical intelligence: they read spatial arrangements of staves, symbols, lyrics, dynamics, articulations, rehearsal marks, annotations, and page layout, transforming musical notation into coordinated musical action. Scholars likewise read sheet music as a document of musical thought, tracing form, harmony, rhythm, text-setting, instrumentation, genre, influence, and performance practice. As Aucoin [2] argues, Western classical music in particular is arguably better understood by its "writtenness" rather than by a stable sound or style. Sheet music is thus both a site of creativity and a shared technology of transmission across otherwise different musical idioms.

We take that visual-musical intelligence as a computational challenge. We introduce Legato 2, a pipeline for sheet music recognition and understanding: extracting symbolic notation and semantic knowledge from visual musical documents.[2] We evaluate the system both as a stand-alone optical music recognition (OMR) model and as an upstream context provider, feeding symbolic output to a frontier language model for musical question answering.

Conventional approaches to OMR typically decompose the task into sequential stages, addressing each with independent modules [4, 22, 6]. Because these pipelined architectures inherently suffer from compounding error propagation, recent research has shifted toward integrated solutions. This

*Equal contribution.

[1]Such traditions include Western classical music, many forms of jazz, liturgical music, theater music, and more.

[2]Code and data are available at `https://github.com/guang-yng/legato` (release v2.0).

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

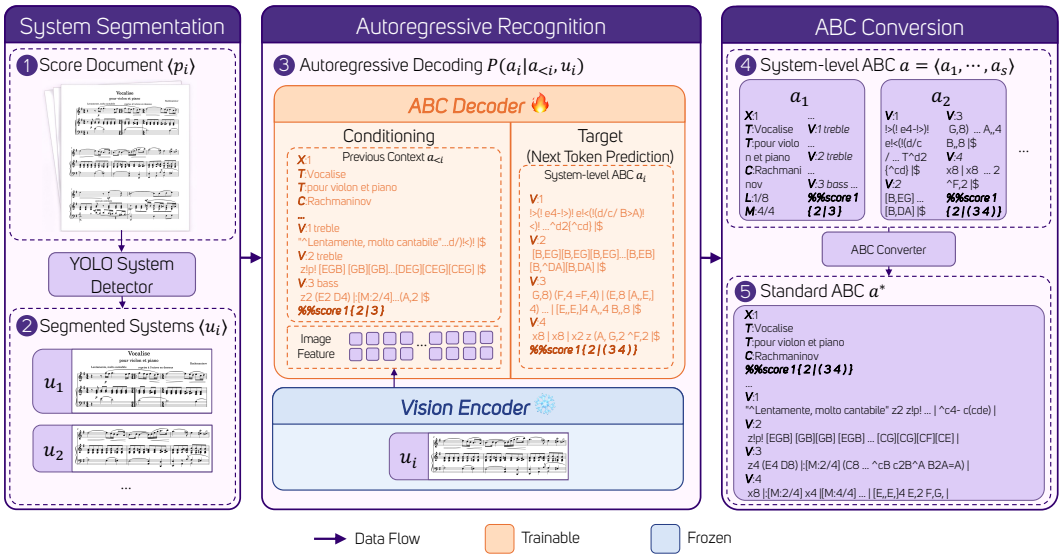


Figure 1: **Overview of Legato 2 pipeline.** The process consists of three stages: (i) System segmentation (§2.1), which uses a YOLO model [14] to extract musical systems from the document; (ii) Autoregressive recognition (§2.2), wherein a vision-language model transcribes the current system based on the preceding context and the current image; and (iii) ABC conversion (§2.3), which employs a rule-based converter to merge the system-level ABC outputs into a standard ABC format.

includes bridging intermediate processing stages [27] or developing end-to-end models either from-scratch [23, 24, 18, 9] or by finetuning [28]. The capabilities of large models have made the end-to-end paradigm highly robust, especially for real printed sheet music. However, previous methods either focus on a single musical system [18, 23] or process a full page at once, ignoring music's sequential structure [24, 28]. They also do not recognize embedded text such as titles, authorship, and annotations. We therefore build a large neural OMR model that reads system by system, recognizes embedded text, and scales naturally to long documents (see §4.2).

The resulting pipeline combines modular and end-to-end components. A vision model segments sheet music into systems, an autoregressive recognition model transcribes each system end-to-end, and a rule-based ABC conversion step merges the outputs. Figure 1 gives the full inference pipeline.

Beyond OMR, sheet music *understanding* has emerged as a challenging task for vision-language models, which are mainly trained on natural images. Prior work shows that frontier VLMs fail on most sheet-music questions [7, 26, 19], especially those involving musical semantics. Rather than modifying these models, we use OMR to provide external symbolic music context. We find that this approach significantly advances the state of the art in sheet music understanding (§4.4).

Our main contributions are:

- We introduce Legato 2, an OMR model for system-by-system recognition (§2), achieving state-of-the-art performance across test datasets and robust long-document processing (§4.1, §4.2).
- Legato 2 is the first neural OMR architecture to transcribe embedded text alongside musical notation, capturing titles, composers, and inline annotations (§2.4, §4.3).
- We show that using OMR output as external context for frontier VLMs (§2.5) establishes a new state of the art on sheet music understanding (MusiXQA, 7; SSMR-Bench, 26; §4.4).

We will enable reproduction by releasing data and code upon publication.

2 Methodology

Our starting point is Legato 1 [28], the previous state-of-the-art OMR model, and its PDMX-Synth training setup. We build on this paradigm but make several targeted departures motivated by the

structure of sheet music. First, we segment pages into musical systems rather than treating the page as a single undifferentiated recognition unit (§2.1). Second, we train the VLM to recognize one system at a time while conditioning on previous system outputs (§2.2). Third, we introduce system-level ABC, an intermediate representation aligned to musical systems and convertible back to standard ABC (§2.3). Fourth, we make the tokenizer text-aware with byte fallback, allowing the model to preserve embedded textual content rather than replacing it with a placeholder (§2.4). Finally, we use the resulting symbolic transcription as context for downstream sheet-music understanding (§2.5). Figure 1 summarizes the pipeline.

2.1 System Segmentation

Similar to previous research [15], we employ the YOLOv8 medium [14] model ($\sim$26M parameters) to segment systems from each page. The checkpoint released by Jung et al. [15] was trained exclusively on piano sheet music. To generalize its capability, we finetune using 1,024 manually annotated, rendered pages of sheet music, encompassing layouts ranging from single-staff systems to complex orchestral arrangements. To make the YOLO model more robust, half of the dataset (512 pages) is randomly sampled from the PDMX-Synth training split while the other half comes from IMSLP [20], a digital library of realistic typeset sheet music images. We provide the training details of this YOLO model in Appendix A.2 and quantify its performance in Appendix C.1.

Formally, given a sheet of music represented as a set of page images $\langle p_1, \ldots, p_n \rangle$, we apply the YOLO model M_{YOLO} to each p_i to obtain a sequence of s_i system bounding boxes: $\langle b_{i,1}, \ldots, b_{i,s_i} \rangle = M_{\text{YOLO}}(p_i)$. We align the top edge of $b_{i,1}$ with the top of the page, and the bottom edge of b_{i,s_i} with the bottom of the page. Combining results from all pages, we get a list of segmented system images defined by the system bounding boxes; we denote these images by $\langle u_1, \ldots, u_s \rangle$.

2.2 Autoregressive Recognition

The VLM component of Legato 2 is designed to recognize an image of a musical system conditioned on the system-level ABC notation of preceding systems. Specifically, our vision-language model M_{VLM} is trained to model the distribution over system-level ABC expression sequences $\langle a_1, \ldots, a_s \rangle$ (explained further in §2.3) given input images of systems $\langle u_1, \ldots, u_s \rangle$:

$$P(a_1, a_2, \cdots, a_s \mid u_1, u_2, \cdots, u_s) = \prod_i P(a_i \mid a_{<i}, u_1, u_2, \cdots, u_s).$$

Because the system-level representation of the ith system depends only on its corresponding visual representation and preceding context, the right-side factors simplify to $P(a_i \mid a_{<i}, u_i)$. Therefore, M_{VLM} is trained autoregressively to model $P(a_i \mid a_{<i}, u_i)$. The detailed context management strategy for the model is provided in Appendix A.1. In our implementation, $a_{<i}$ is left truncated to 1,024 tokens. Legato 2 has a maximum context length of 2,048 tokens; the other 1,024 are given to the target, a_i.

For the vision-language model M_{VLM}, we adopt the same architecture as Legato 1 [28] due to its demonstrated effectiveness. The total number of trainable parameters is 113.7M. The details of training hyper-parameters are shown in Appendix A.2. Additionally, we conducted an ablation study to investigate the impact of scaling up the architecture (see §4.5). We also quantify robustness of the VLM against upstream YOLO errors in Appendix C.2.

2.3 System-Level ABC

Following Yang et al. [28], we adopt the ABC notation as our target format due to its simplicity and sequential structure. The ABC notation in PDMX-Synth is organized in a voice-by-voice sequence, as illustrated in Figures 1 and 2, which we denote as a^*. This representation does not provide a system-level alignment between the systems in the sheet music image and the underlying symbolic music. Inspired by SMT-ABC Notation [21], we introduce a variant of the ABC notation, "System-Level ABC". Formally, the system-level ABC representation of a piece of sheet music with s systems is defined as $a = \langle a_1, \ldots, a_s \rangle$, where a_i denotes the system-level representation of the ith system. This variant is organized in a system-by-system order. Consequently, the ABC representation a_i for each system depends on its corresponding visual representation and preceding context, but remains otherwise independent of subsequent context; for example, newly introduced voices do not appear in

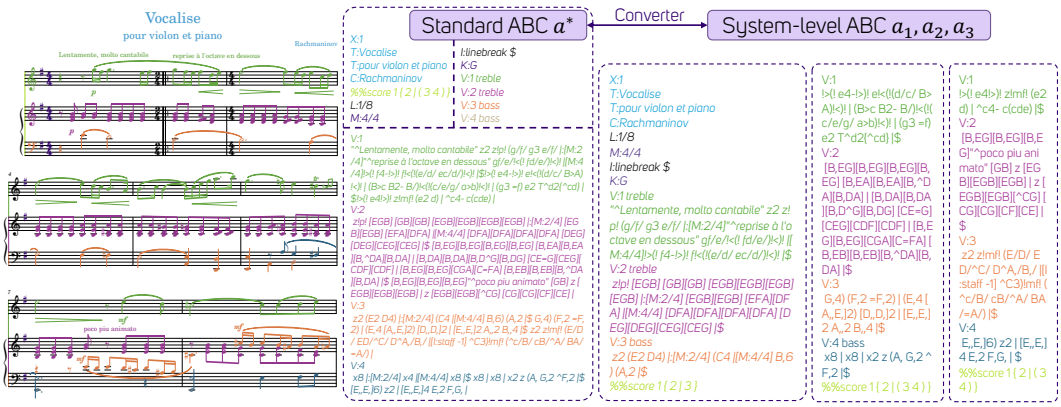


Figure 2: **An example of a piece of sheet music and its corresponding ABC representations.** Unlike the standard ABC format a^* , our system-level ABC $a = \langle a_1, a_2, a_3 \rangle$ is organized system by system, ensuring that the encoding of a given system remains independent of subsequent context.

prior systems. To bridge these formats, we implement a rule-based converter to map between a^* and a (see details in Appendix A.3). This enables the ground-truth data in PDMX-Synth to be split into individual systems for training (converting a^* to a), while also allowing the VLM's predictions for individual systems to be reconstructed back into the standard ABC format a^*.

2.4 Text-Aware ABC Tokenizer

Unlike Legato 1 which replaces all text spans with a single `<|text|>` token [28], a central goal of our system is to transcribe both musical notation and embedded text, including titles, composer names, staff labels, and inline annotations. Although the BPE over ABC tokenizer introduced in Legato 1 preserves frequent musical patterns, the lack of textual elements in its training data prevents the recovery of the original text. We therefore train a text-aware ABC tokenizer that preserves textual content, while still retaining the ability to represent complex musical concepts efficiently.

Following PDMX-Synth, we construct the tokenizer-training corpus from PDMX [16], but, unlike Legato 1, we retain all text.[3] This text-inclusive PDMX-Synth corpus contains relatively sparse text that is dominated by a few common languages, and we find that naive BPE over this corpus allocates vocabulary to many rare characters that are not learned reliably, degrading performance (Appendix F.3). To retain the benefits of BPE for musical notation while still supporting arbitrary text, we instead use BPE with byte fallback. We initialize the tokenizer with the 243 valid byte values used in UTF-8 and learn merge rules from the text-inclusive PDMX-Synth corpus. This allows common musical and textual patterns to be represented compactly, while unseen or rare characters can still be represented through their byte sequences. §4.5 evaluates the performance of our byte-fallback tokenizer at different vocabulary sizes; we find best performance at a vocabulary size of 4096.

2.5 OMR for Sheet Music Understanding

As shown by [28], frontier vision-language models still make many errors when transcribing sheet music; we therefore test whether OMR output can serve as complementary context for them. Given an image of sheet music, we first transcribe it with Legato 2, and then provide the resulting transcription to the frontier VLM alongside the original image when prompting it on a downstream task. In this setup, the frontier VLM can optionally use the transcription in addition to the visual input, allowing it to condition on a symbolic representation of the sheet music. We hypothesize that access to this transcription improves the model's music understanding abilities. The exact prompts used in this investigation are given in the Appendix D. We test GPT-5 [25] and Gemini 3.1 Pro [10], two of the top-performing vision language models, as part of our experiments.

[3]We continue to exclude lyrics from our corpus as they are both uncommon in PDMX and less tightly coupled to the musical structure than other embedded textual elements; we expect that existing OCR tools can be adapted for lyric reconstruction in future work.

3 Implementation and Evaluation Details

3.1 Baselines

We evaluate our approach against three distinct baselines. First, **Legato 1** [28] represents the state of the art in end-to-end neural OMR, utilizing a vision-language architecture to autoregressively transcribe full-page or concatenated multi-page images directly into ABC notation. Second, we compare against **Audiveris** [5], a comprehensive, open-source rule-based system that outputs MusicXML, noting it as the sole prior system equipped to transcribe embedded textual metadata. Finally, we benchmark general-purpose **Frontier VLMs**—specifically GPT-5 and Gemini 3.1 Pro [10, 25]—on both sheet music recognition and understanding tasks, using the prompts detailed in Appendix D. Note that Audiveris requires input preprocessing, which we detail in Appendix B.

3.2 Evaluation Metrics

For our main results, we report the Optical Music Recognition - Normalized Edit Distance (OMR-NED) [17]. This is consistent with prior work such as Legato 1, and offers several benefits over traditional metrics such as Symbol Error Rate [28]. More detail is provided in Appendix E.1.

To evaluate embedded text recognition, we use character error rate (CER) for OMR-style symbolic outputs and unordered character error rate (Unordered-CER) for OCR-style bounding-box outputs. For OMR models, we convert outputs to ABC, extract title, composer, and inline quoted text, canonicalize the extracted text, and compute character-level Levenshtein distance. For OCR models, whose bounding boxes have no reliable canonical order, we instead compute a set edit distance over text boxes using Levenshtein distance as the box-level edit cost. Full details are provided in Appendix E.3.

3.3 Evaluation Datasets

3.3.1 Sheet Music Recognition

We employ a series of realistic datasets—including the test split of PDMX-Synth [28], OpenScore String Quartets, OpenScore Lieder, and IMSLP Piano Scores—to assess model performance in sheet music recognition. Note that the OpenScore String Quartets and Lieder datasets used in this suite are small evaluation subsets derived from their respective full collections [13, 12], providing both software-rendered images and scans of published physical copies retrieved from IMSLP. IMSLP Piano Scores is an additional dataset sourced from IMSLP and annotated by the authors of Legato 1 [28].

3.3.2 Sheet Music Understanding

MusiXQA We evaluate sheet music understanding on MusiXQA [7], a visual question-answering benchmark that tests multimodal reasoning across core musical elements (e.g., pitch, duration, chords, clefs, signatures, and textual markings). The benchmark provides two test splits; we evaluate exclusively on the harder "OMR split"—which necessitates joint recognition and semantic comprehension—because the "simple split" is already saturated by frontier VLMs (scoring $> 90\%$). Following Chen et al. [7], we measure open-ended accuracy using the G-Acc metric, wherein a language model judges the semantic correctness of predictions against a ground-truth reference. Because the originally specified GPT-4o is no longer publicly accessible, we substitute GPT-5 as our evaluator.

SSMR-Bench We additionally evaluate Legato 2 on the SSMR-Bench dataset [26]. Unlike MusiXQA, SSMR-Bench is structured as a multiple-choice visual question-answering task, where sheet music images may appear in both the prompts and the candidate choices. The questions focus on sheet music reasoning and are programmatically generated around core musical concepts, including rhythm, chords, intervals, and scales. SSMR-Bench provides both training and evaluation splits, but we only utilize the evaluation set. Following the original methodology, we employ standard accuracy as our evaluation metric.

Table 1: **Evaluation of page-level sheet music recognition.** Legato 2 consistently outperforms other models across all datasets.

Dataset (# of Pages)	OMR-NED ↓			
	Audiveris	Gemini 3.1 Pro	Legato 1	Legato 2
PDMX-Synth Test Set (411)	56.3	90.3	28.6	**23.5**
Rendered OpenScore String Quartets (252)	64.6	93.5	32.9	**17.1**
Camera OpenScore String Quartets (252)	75.4	94.1	58.2	**31.6**
Rendered OpenScore Lieder (64)	76.9	90.9	39.5	**27.6**
Camera OpenScore Lieder (64)	85.8	91.7	44.9	**43.6**
IMSLP Piano Scores (32)	71.5	89.1	44.3	**34.2**

3.4 Implementation Details

For VLM training, we adopt the protocol established in [28] (see Appendix A.2). Because our VLM is trained exclusively on synthetic sheet music but evaluated on real-world data, optimizing hyperparameters on the PDMX-Synth validation set may cause the model to overfit to synthetic artifacts. To address this, we compile a new validation set comprising 130 pages from the OpenScore String Quartets and OpenScore Lieder datasets, ensuring these pages are strictly excluded from all test sets. For each model checkpoint, we perform a grid search on this validation set over a repetition penalty of $\{1.0, 1.1, 1.2\}$ and a beam size of $\{1, 2, 5, 10\}$. Finally, we select the checkpoint and its corresponding inference parameters that achieve the lowest OMR-NED on this validation set. See Appendix G for detailed validation results of all model variants.

4 Experimental Results

4.1 Page-level Recognition

We evaluate Legato 2 under the exact same setting as Legato 1; specifically, we provide the model with a single page of sheet music and evaluate its predictions against the ground truth of the corresponding page. As shown in Table 1, Legato 2 outperforms Legato 1, the previous state of the art, across all datasets. As noted by Yang et al. [28], Legato 1 exhibits degraded performance on the OpenScore String Quartets dataset due to the dense visual nature of string quartets, where a single page contains numerous staves of individual voices. In contrast, Legato 2 demonstrates significant improvements on these datasets primarily because it operates at the system level. By first segmenting the sheet music into systems, the VLM is able to process each system at a higher effective resolution.

4.2 Multi-page Recognition

While page-level evaluation establishes a baseline, real-world applications require multi-page processing. Previous methods either concatenate images (degrading performance) or process pages independently (losing cross-page context and requiring complex merging logic). In contrast, our autoregressive, system-level processing naturally accommodates arbitrary-length documents. We demonstrate this advantage through multi-page sheet music recognition evaluations. To this end, we utilize the full OpenScore Lieder dataset—distinct from the 64-page evaluation subset in Table 1—which comprises sheet music ranging from 1 to 37 pages per piece. We render the document images directly from source MusicXML files and partition them into five bins based on cumulative aspect ratio: one bin for ratios ≥ 12, and four equally spaced bins for the remainder. To counteract naturally uneven length distributions, we uniformly sample 100 instances per bin, yielding a balanced evaluation set of 500 documents.

Due to the high computational cost associated with processing large, multi-page images, this analysis is restricted to a direct comparison between Legato 1 and 2. As illustrated in Figure 3, Legato 2 not only outperforms Legato 1 across all aspect ratio bins, but its performance also degrades at a substantially slower rate, demonstrating robust capabilities across varying document lengths. We attribute this stability to our context management strategy: by applying left-truncation to the context window during training, the model naturally adapts to the identical truncation applied during inference.

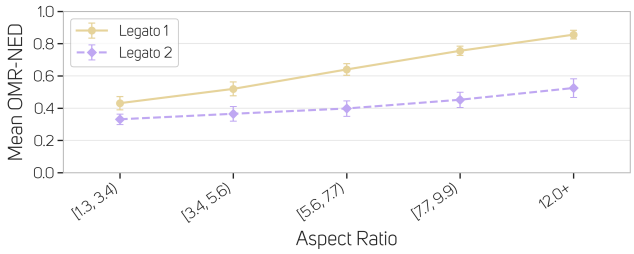


Figure 3: **Evaluation on multi-page sheet music recognition.** Legato 2 consistently outperforms Legato 1 across all aspect ratio bins, and also degrades at a substantially slower rate. The error bars represent 95% normal approximated confidence intervals.

Table 2: **Evaluation on text recognition.** Lower is better for all metrics. Legato 2 is compared against OMR models (Table 2a) and OCR models (Table 2b) on Rendered OpenScore String Quartets.

Model	Character Error Rates (CER) % ↓			
	title	composer	others	Total
Audiveris	29.8	157.9	101.0	78.8
Gemini	18.7	130.8	76.1	58.1
Legato 2	**10.6**	**97.2**	**31.0**	**24.8**

(a) **Character error rates for textual elements.** The categories "`title`", "`composer`", and "`others`" denote the error rates for the title, composer name and other embedded annotations. The error for "`composer`" is high because models usually detect page numbers as composer names.

Model	Unordered CER % ↓
PaddleOCR	85.5
Legato 2	**25.1**

(b) **Unordered character error rates for textual elements.** The baseline OCR model likely struggles to distinguish text from the surrounding musical notation.

This consistency between the training and inference phases effectively mitigates the performance degradation typically associated with exceptionally long documents.

4.3 Text Recognition

The text-aware tokenizer in Legato 2 enables neural OMR to recognize embedded textual elements for the first time. We evaluate this capability on the Rendered OpenScore String Quartets dataset. We compare against prior text-capable OMR systems and a general-purpose OCR system applied directly to full-page sheet music. Because Legato 1 and earlier neural OMR models do not process text, we use Audiveris and Gemini 3.1 Pro as representative rule-based and frontier VLM baselines. For the OCR baseline, we employ PaddleOCR [8], a widely used general-purpose OCR system. Tables 2a and 2b report results comparing to the OMR and OCR baselines, respectively. Legato 2 outperforms all baselines on textual components, even when text is evaluated independently of the musical notation.

4.4 Sheet Music Understanding

We evaluate frontier VLMs for sheet music understanding under three contexts: no transcription, Legato 1's transcription, and Legato 2's transcription. As shown in Table 3, OMR context improves performance on both benchmarks. Legato 2's transcriptions yield further gains over Legato's, likely because they are higher quality. Although current OMR models remain imperfect, these results suggest that frontier VLMs can extract information more effectively from symbolic OMR outputs than from raw sheet music images.

4.5 Ablation Studies

System Segmentation and Byte Fallback We first investigate the effectiveness of two key design choices in Legato 2: system segmentation and byte fallback. Starting from the baseline Legato 1 architecture, we first incorporate the system segmentation mechanism; specifically, we train an identical Legato 1 model from scratch under the system-by-system conditional generation setting.

Table 3: **Evaluation on sheet music understanding task.** Higher is better for all metrics. The "Context" column specifies which OMR model's output is used as context.

Context	Gemini 3 Flash	GPT-5 mini
None	8.4	8.0
Legato 1	20.1	16.6
Legato 2	**25.3**	**20.2**

(a) **G-Acc ↑ on MusiXQA.**

Context	Gemini 3.1 Pro	GPT-5
None	71.4	51.8
Legato 1	84.8	65.3
Legato 2	**92.7**	**71.7**

(b) **Accuracy ↑ on SSMR-Bench.**

Table 4: **Ablation study on system segmentation (`SS`), byte fallback (`BF`), and vocabulary size.** Starting from the Legato 1 baseline, we incorporate system segmentation and byte fallback into the architecture, and subsequently increase the vocabulary size from 2,048 to 4,096 (`+V4096`) and 8,192 (`+V8192`). The modifications from left to right are cumulative. †: Indicates our final model, Legato 2, selected based on validation error.

Dataset (# of Pages)	OMR-NED ↓				
	Legato 1	`+SS`	`+BF`	`+V4096`†	`+V8192`
Validation Set (130)	70.2	48.5	48.5	**46.8**	52.8
PDMX-Synth Test Set (411)	28.6	25.8	26.4	**23.5**	**23.5**
Rendered OpenScore String Quartets (252)	32.9	17.9	**16.4**	17.1	16.6
Camera OpenScore String Quartets (252)	58.2	35.2	35.6	**31.6**	33.5
Rendered OpenScore Lieder (64)	39.5	29.1	28.7	**27.6**	31.3
Camera OpenScore Lieder (64)	44.9	45.8	44.8	43.6	**42.7**
IMSLP Piano Scores (32)	44.3	34.3	35.1	**34.2**	37.8

Note that this intermediate model retains the original Legato 1 tokenizer. Next, we replace this tokenizer with the byte-fallback variant described in §2.4, and retrain both the tokenizer and the model. To ensure a fair comparison, we maintain a vocabulary size of 2,048, identical to that of the original Legato. The performance of these two variants, alongside the base Legato 1 model, is reported in the left three columns of Table 4. The results demonstrate that system segmentation improves performance by a large margin. Furthermore, the incorporation of byte fallback, while enabling the model to process embedded textual content, yields comparable performance.

Vocabulary Size Based on the previous model (Legato 1 `+SS+BF`), we then investigate the effect of different vocabulary sizes on Legato 2's performance. Maintaining the tokenizer construction methodology detailed in §2.4, we vary only the final vocabulary size. While larger vocabulary sizes enable the tokenizer to capture more common patterns from the training data, they also increase the risk of overfitting. Right three columns of Table 4 present the validation and test errors across the evaluated vocabulary sizes $\{2048, 4096, 8192\}$. Instead of Legato 1's vocabulary size of 2048, we selected a vocabulary size of 4,096 strictly based on achieving the lowest validation error, rather than test error.

Additional Ablation Studies We explore further architectural modifications, which ultimately yield minimal improvements (Appendix F.1). As detailed in Appendix F.2, we hypothesize these limited scaling benefits stem from the distribution gap between the training and evaluation datasets. Finally, we investigate a variant employing a BPE tokenizer naively trained on the text-inclusive PDMX-Synth dataset (Appendix F.3).

4.6 Qualitative Results

Sheet Music Recognition Figure 4 illustrates an example from the IMSLP Piano Scores dataset. This example was selected because the OMR-NED scores for both models closely approximate their respective averages across the entire dataset. For this specific example, Legato 1 and Legato 2 achieve OMR-NED scores of 46.1 and 33.5, respectively. As observed, both models fail to transcribe the

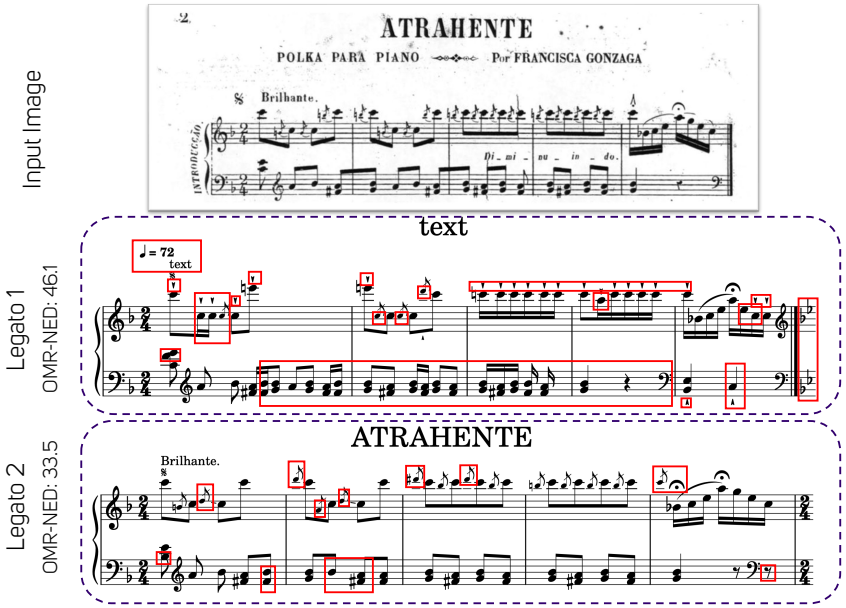


Figure 4: **Qualitative example from the IMSLP Piano Scores dataset.** The red boxes represent errors from the ground truth. This example was selected because the OMR-NED scores for both models closely approximate their respective dataset averages. Although only the first system is shown here, the models process the full-page image as input. Note that the input's lower staff exhibits a rare formatting irregularity where beams cross the barlines.

subtitle and the "`Diminuindo.`" annotation, an omission likely attributable to the unconventional placement of these textual elements. Furthermore, while Legato 1 only detects the spatial position of the title, Legato 2 accurately transcribes the exact text. At the beginning of the sheet music, Legato 1 generates a redundant tempo mark; this is likely because the model is confused by the "`Brilhante.`" annotation. In contrast, Legato 2 processes this content correctly. Notably, Legato 1 also misses most of the grace notes, misclassifying them as staccatissimo marks. Conversely, because Legato 2 operates on a system-by-system basis, it captures these fine-grained visual details much more effectively. Finally, it is worth noting that the input sheet music contains formatting irregularities; specifically, on the lower staff, several beams cross the barlines. These visual artifacts disrupt Legato 1's predictions, causing it to generate an output that improperly mixes these measures. Although Legato 2 generates redundant chords in the first two measures, it quickly recovers to produce correct predictions in measures 3 and 4, demonstrating greater robustness to such layout anomalies. Appendix E.2 breaks down the OMR-NED of Legato 2 on OpenScore String Quartets by error category.

Sheet Music Understanding We also provide a sheet music understanding example in Appendix H.1, where we show that Gemini 3 Flash [11] successfully uses and cites Legato 2's transcription when responding to a question from SSMR-Bench.

5 Conclusions and Future Work

In this paper, we introduced Legato 2, an optical music recognition pipeline that segments sheet music into individual systems and trains a vision-language model to recognize each system autoregressively. We demonstrated that this approach achieves state-of-the-art performance in both single-page and multi-page sheet music recognition. Furthermore, we integrated the capability to recognize embedded textual elements into Legato 2, surpassing previous OMR baselines. Finally, we investigated the potential of using OMR as an upstream context provider for broader sheet music understanding, demonstrating that specialized OMR models can effectively complement general-purpose large vision-language models on domain-specific tasks. Looking forward, we envision a framework where diverse, specialized musical tools—including OMR, music transcription, and analysis software—can be used

by general VLMs to solve complex, open-ended musical problems. Such a modular architecture promises greater flexibility, efficiency, and accuracy compared to monolithic general-purpose models.

Acknowledgments and Disclosure of Funding

We thank Nazif Can Tamer for strongly advocating for the system-level OMR direction and repeatedly encouraging us to pursue it, drawing our attention to prior work and resources based on YOLO-based music-system detection, and connecting us with Dasaem Jeong to access these resources. We also acknowledge his input during the early development of the idea of using OMR outputs as external symbolic context. We thank Dasaem Jeong for providing early access to the music-system detection resources that we subsequently fine-tuned for this work.

Funding. This work received no specific funding.

Competing interests. The authors declare no competing interests.

References

[1] AI@Meta. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. Technical report, Meta Platforms, Inc., September 2024. URL https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/.

[2] Matthew Aucoin. Do you actually know what classical music is? Does anyone? *The Atlantic*, April 2025. URL https://www.theatlantic.com/magazine/archive/2025/05/aucoin-what-is-classical-music/682119/.

[3] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL technical report, 2025. URL https://arxiv.org/abs/2511.21631.

[4] David Bainbridge and Tim Bell. The challenge of optical music recognition. *Computers and the Humanities*, 35(2):95–121, May 2001. ISSN 1572-8412. doi: 10.1023/A:1002485918032. URL https://doi.org/10.1023/A:1002485918032.

[5] Hervé Bitteur and Audiveris Contributors. Audiveris, 2025. URL https://github.com/Audiveris/audiveris.

[6] Jorge Calvo-Zaragoza, Jan Hajič Jr., and Alexander Pacha. Understanding optical music recognition. *ACM Comput. Surv.*, 53(4), July 2020. ISSN 0360-0300. doi: 10.1145/3397499. URL https://doi.org/10.1145/3397499.

[7] Jian Chen, Wenye Ma, Penghang Liu, Wei Wang, Tengwei Song, Ming Li, Chenguang Wang, Jiayu Qin, Ruiyi Zhang, and Changyou Chen. MusiXQA: Advancing visual music understanding in multimodal large language models, 2025. URL https://arxiv.org/abs/2506.23009.

[8] Cheng Cui, Ting Sun, Manhui Lin, Tingquan Gao, Yubo Zhang, Jiaxuan Liu, Xueqing Wang, Zelun Zhang, Changda Zhou, Hongen Liu, Yue Zhang, Wenyu Lv, Kui Huang, Yichao Zhang, Jing Zhang, Jun Zhang, Yi Liu, Dianhai Yu, and Yanjun Ma. PaddleOCR 3.0 technical report, 2025. URL https://arxiv.org/abs/2507.05595.

[9] Daniel Dratschuk and Paul Swoboda. Transcoda: End-to-end zero-shot optical music recognition via data-centric synthetic training, 2026. URL https://arxiv.org/abs/2605.10835.

[10] Google DeepMind. Gemini 3.1 Pro model card. Model card, Google DeepMind, February 2026. URL https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-1-Pro-Model-Card.pdf. Published February 2026.

[11] Google DeepMind. Gemini 3 flash model card. https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Flash-Model-Card.pdf, 2026. Model card for the Gemini 3 Flash multimodal model.

[12] Mark Gotham, Maureen Redbond, Bruno Bower, and Peter Jonas. The "OpenScore String Quartet" corpus. In *Proceedings of the 10th International Conference on Digital Libraries for Musicology*, pages 49–57, New York, NY, USA, 2023. Association for Computing Machinery. doi: 10.1145/3625135.3625155.

[13] Mark Robert Haigh Gotham and Peter Jonas. The OpenScore Lieder corpus. In *Music Encoding Conference Proceedings 2021*, pages 131–136. Humanities Commons, 2022. doi: 10.17613/1my2-dm23.

[14] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLO, 2023. URL https://github.com/ultralytics/ultralytics.

[15] Jongmin Jung, Dongmin Kim, Sihun Lee, Seola Cho, Hyungjoon So, Irmak Bukey, Chris Donahue, and Dasaem Jeong. U-MusT: A unified framework for cross-modal translation of score images, symbolic music, and performance audio. *IEEE Transactions on Audio, Speech and Language Processing*, pages 1–16, 2025. doi: 10.1109/TASLPRO.2025.3648794.

[16] Phillip Long, Zachary Novack, Taylor Berg-Kirkpatrick, and Julian McAuley. PDMX: A large-scale public domain musicxml dataset for symbolic music processing. In *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2025. doi: 10.1109/ICASSP49660.2025.10890217.

[17] Juan C. Martinez-Sevilla, Joan Cerveto-Serrano, Noelia Luna, Greg Chapman, Craig Sapp, David Rizo, and Jorge Calvo-Zaragoza. Sheet music benchmark: Standardized optical music recognition evaluation, 2025. URL https://arxiv.org/abs/2506.10488.

[18] Jiří Mayer, Milan Straka, Jan Hajič, and Pavel Pecina. Practical end-to-end optical music recognition for pianoform music. In Elisa H. Barney Smith, Marcus Liwicki, and Liangrui Peng, editors, *Document Analysis and Recognition - ICDAR 2024*, pages 55–73, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-70552-6.

[19] Gagan Mundada, Yash Vishe, Amit Namburi, Xin Xu, Zachary Novack, Julian McAuley, and Junda Wu. WildScore: Benchmarking MLLMs in-the-wild symbolic music reasoning. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 16847–16863, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.853. URL https://aclanthology.org/2025.emnlp-main.853/.

[20] Project Petrucci LLC. IMSLP petrucci music library. https://imslp.org, 2026.

[21] Xingwei Qu, yuelin bai, Yinghao MA, Ziya Zhou, Ka Man Lo, JIAHENG LIU, Ruibin Yuan, Lejun Min, Xueling Liu, Tianyu Zhang, Xeron Du, Shuyue Guo, Yiming Liang, Yizhi Li, Shangda Wu, Junting Zhou, Tianyu Zheng, Ziyang Ma, Fengze Han, Wei Xue, Gus Xia, Emmanouil Benetos, Xiang Yue, Chenghua Lin, Xu Tan, Wenhao Huang, Jie Fu, and Ge Zhang. MuPT: A generative symbolic music pretrained transformer. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu, editors, *International Conference on Learning Representations*, volume 2025, pages 46753–46779, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/73f6f8897896f7bda86ea7d1ebc1dc4f-Paper-Conference.pdf.

[22] Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andre R. S. Marcal, Carlos Guedes, and Jaime S. Cardoso. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1(3):173–190, Oct 2012. ISSN 2192-662X. doi: 10.1007/s13735-012-0004-6. URL https://doi.org/10.1007/s13735-012-0004-6.

[23] Antonio Ríos-Vila, Jorge Calvo-Zaragoza, and Thierry Paquet. Sheet music transformer: End-to-end optical music recognition beyond monophonic transcription. In Elisa H. Barney Smith, Marcus Liwicki, and Liangrui Peng, editors, *Document Analysis and Recognition - ICDAR 2024*, pages 20–37, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-70552-6.

[24] Antonio Ríos-Vila, Jorge Calvo-Zaragoza, David Rizo, and Thierry Paquet. End-to-end full-page optical music recognition for pianoform sheet music, 2025. URL `https://arxiv.org/abs/2405.12105`.

[25] Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, Akshay Nathan, Alan Luo, Alec Helyar, Aleksander Madry, Aleksandr Efremov, Aleksandra Spyra, Alex Baker-Whitcomb, Alex Beutel, Alex Karpenko, Alex Makelov, Alex Neitz, Alex Wei, Alexandra Barr, Alexandre Kirchmeyer, Alexey Ivanov, Alexi Christakis, Alistair Gillespie, Allison Tam, Ally Bennett, Alvin Wan, Alyssa Huang, Amy McDonald Sandjideh, Amy Yang, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrei Gheorghe, Andres Garcia Garcia, Andrew Braunstein, Andrew Liu, Andrew Schmidt, Andrey Mereskin, Andrey Mishchenko, Andy Applebaum, Andy Rogerson, Ann Rajan, Annie Wei, Anoop Kotha, Anubha Srivastava, Anushree Agrawal, Arun Vijayvergiya, Ashley Tyra, Ashvin Nair, Avi Nayak, Ben Eggers, Bessie Ji, Beth Hoover, Bill Chen, Blair Chen, Boaz Barak, Borys Minaiev, Botao Hao, Bowen Baker, Brad Lightcap, Brandon McKinzie, Brandon Wang, Brendan Quinn, Brian Fioca, Brian Hsu, Brian Yang, Brian Yu, Brian Zhang, Brittany Brenner, Callie Riggins Zetino, Cameron Raymond, Camillo Lugaresi, Carolina Paz, Cary Hudson, Cedric Whitney, Chak Li, Charles Chen, Charlotte Cole, Chelsea Voss, Chen Ding, Chen Shen, Chengdu Huang, Chris Colby, Chris Hallacy, Chris Koch, Chris Lu, Christina Kaplan, Christina Kim, CJ Minott-Henriques, Cliff Frey, Cody Yu, Coley Czarnecki, Colin Reid, Colin Wei, Cory Decareaux, Cristina Scheau, Cyril Zhang, Cyrus Forbes, Da Tang, Dakota Goldberg, Dan Roberts, Dana Palmie, Daniel Kappler, Daniel Levine, Daniel Wright, Dave Leo, David Lin, David Robinson, Declan Grabb, Derek Chen, Derek Lim, Derek Salama, Dibya Bhattacharjee, Dimitris Tsipras, Dinghua Li, Dingli Yu, DJ Strouse, Drew Williams, Dylan Hunn, Ed Bayes, Edwin Arbus, Ekin Akyurek, Elaine Ya Le, Elana Widmann, Eli Yani, Elizabeth Proehl, Enis Sert, Enoch Cheung, Eri Schwartz, Eric Han, Eric Jiang, Eric Mitchell, Eric Sigler, Eric Wallace, Erik Ritter, Erin Kavanaugh, Evan Mays, Evgenii Nikishin, Fangyuan Li, Felipe Petroski Such, Filipe de Avila Belbute Peres, Filippo Raso, Florent Bekerman, Foivos Tsimpourlas, Fotis Chantzis, Francis Song, Francis Zhang, Gaby Raila, Garrett McGrath, Gary Briggs, Gary Yang, Giambattista Parascandolo, Gildas Chabot, Grace Kim, Grace Zhao, Gregory Valiant, Guillaume Leclerc, Hadi Salman, Hanson Wang, Hao Sheng, Haoming Jiang, Haoyu Wang, Haozhun Jin, Harshit Sikchi, Heather Schmidt, Henry Aspegren, Honglin Chen, Huida Qiu, Hunter Lightman, Ian Covert, Ian Kivlichan, Ian Silber, Ian Sohl, Ibrahim Hammoud, Ignasi Clavera, Ikai Lan, Ilge Akkaya, Ilya Kostrikov, Irina Kofman, Isak Etinger, Ishaan Singal, Jackie Hehir, Jacob Huh, Jacqueline Pan, Jake Wilczynski, Jakub Pachocki, James Lee, James Quinn, Jamie Kiros, Janvi Kalra, Jasmyn Samaroo, Jason Wang, Jason Wolfe, Jay Chen, Jay Wang, Jean Harb, Jeffrey Han, Jeffrey Wang, Jennifer Zhao, Jeremy Chen, Jerene Yang, Jerry Tworek, Jesse Chand, Jessica Landon, Jessica Liang, Ji Lin, Jiancheng Liu, Jianfeng Wang, Jie Tang, Jihan Yin, Joanne Jang, Joel Morris, Joey Flynn, Johannes Ferstad, Johannes Heidecke, John Fishbein, John Hallman, Jonah Grant, Jonathan Chien, Jonathan Gordon, Jongsoo Park, Jordan Liss, Jos Kraaijeveld, Joseph Guay, Joseph Mo, Josh Lawson, Josh McGrath, Joshua Vendrow, Joy Jiao, Julian Lee, Julie Steele, Julie Wang, Junhua Mao, Kai Chen, Kai Hayashi, Kai Xiao, Kamyar Salahi, Kan Wu, Karan Sekhri, Karan Sharma, Karan Singhal, Karen Li, Kenny Nguyen, Keren Gu-Lemberg, Kevin King, Kevin Liu, Kevin Stone, Kevin Yu, Kristen Ying, Kristian Georgiev, Kristie Lim, Kushal Tirumala, Kyle Miller, Lama Ahmad, Larry Lv, Laura Clare, Laurance Fauconnet, Lauren Itow, Lauren Yang, Laurentia Romaniuk, Leah Anise, Lee Byron, Leher Pathak, Leon Maksin, Leyan Lo, Leyton Ho, Li Jing, Liang Wu, Liang Xiong, Lien Mamitsuka, Lin Yang, Lindsay McCallum, Lindsey Held, Liz Bourgeois, Logan Engstrom, Lorenz Kuhn, Louis Feuvrier, Lu Zhang, Lucas Switzer, Lukas Kondraciuk, Lukasz Kaiser, Manas Joglekar, Mandeep Singh, Mandip Shah, Manuka Stratta, Marcus Williams, Mark Chen, Mark Sun, Marselus Cayton, Martin Li, Marvin Zhang, Marwan Aljubeh, Matt Nichols, Matthew Haines, Max Schwarzer, Mayank Gupta, Meghan Shah, Melody Huang, Meng Dong, Mengqing Wang, Mia Glaese, Micah Carroll, Michael Lampe, Michael Malek, Michael Sharman, Michael Zhang, Michele Wang, Michelle Pokrass, Mihai Florian, Mikhail Pavlov, Miles Wang, Ming Chen, Mingxuan

Wang, Minnia Feng, Mo Bavarian, Molly Lin, Moose Abdool, Mostafa Rohaninejad, Nacho Soto, Natalie Staudacher, Natan LaFontaine, Nathan Marwell, Nelson Liu, Nick Preston, Nick Turley, Nicklas Ansman, Nicole Blades, Nikil Pancha, Nikita Mikhaylin, Niko Felix, Nikunj Handa, Nishant Rai, Nitish Keskar, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Oona Gleeson, Pamela Mishkin, Patryk Lesiewicz, Paul Baltescu, Pavel Belov, Peter Zhokhov, Philip Pronin, Phillip Guo, Phoebe Thacker, Qi Liu, Qiming Yuan, Qinghua Liu, Rachel Dias, Rachel Puckett, Rahul Arora, Ravi Teja Mullapudi, Raz Gaon, Reah Miyara, Rennie Song, Rishabh Aggarwal, RJ Marsan, Robel Yemiru, Robert Xiong, Rohan Kshirsagar, Rohan Nuttall, Roman Tsiupa, Ronen Eldan, Rose Wang, Roshan James, Roy Ziv, Rui Shu, Ruslan Nigmatullin, Saachi Jain, Saam Talaie, Sam Altman, Sam Arnesen, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Sarah Yoo, Savannah Heon, Scott Ethersmith, Sean Grove, Sean Taylor, Sebastien Bubeck, Sever Banesiu, Shaokyi Amdo, Shengjia Zhao, Sherwin Wu, Shibani Santurkar, Shiyu Zhao, Shraman Ray Chaudhuri, Shreyas Krishnaswamy, Shuaiqi, Xia, Shuyang Cheng, Shyamal Anadkat, Simón Posada Fishman, Simon Tobin, Siyuan Fu, Somay Jain, Song Mei, Sonya Egoian, Spencer Kim, Spug Golden, SQ Mah, Steph Lin, Stephen Imm, Steve Sharpe, Steve Yadlowsky, Sulman Choudhry, Sungwon Eum, Suvansh Sanjeev, Tabarak Khan, Tal Stramer, Tao Wang, Tao Xin, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Degry, Thomas Shadwell, Tianfu Fu, Tianshi Gao, Timur Garipov, Tina Sriskandarajah, Toki Sherbakov, Tomer Kaftan, Tomo Hiratsuka, Tongzhou Wang, Tony Song, Tony Zhao, Troy Peterson, Val Kharitonov, Victoria Chernova, Vineet Kosaraju, Vishal Kuo, Vitchyr Pong, Vivek Verma, Vlad Petrov, Wanning Jiang, Weixing Zhang, Wenda Zhou, Wenlei Xie, Wenting Zhan, Wes McCabe, Will DePue, Will Ellsworth, Wulfie Bain, Wyatt Thompson, Xiangning Chen, Xiangyu Qi, Xin Xiang, Xinwei Shi, Yann Dubois, Yaodong Yu, Yara Khakbaz, Yifan Wu, Yilei Qian, Yin Tat Lee, Yinbo Chen, Yizhen Zhang, Yizhong Xiong, Yonglong Tian, Young Cha, Yu Bai, Yu Yang, Yuan Yuan, Yuanzhi Li, Yufeng Zhang, Yuguang Yang, Yujia Jin, Yun Jiang, Yunyun Wang, Yushi Wang, Yutian Liu, Zach Stubenvoll, Zehao Dou, Zheng Wu, and Zhigang Wang. OpenAI GPT-5 system card, 2025. URL `https://arxiv.org/abs/2601.03267`.

[26] Zhilin Wang, Zhe Yang, Yun Luo, Yafu Li, Xiaoye Qu, Ziqian Qiao, Haoran Zhang, Runzhe Zhan, Derek F. Wong, Jizhe Zhou, and Yu Cheng. Towards an AI musician: Synthesizing sheet music problems for musical reasoning, 2025. URL `https://arxiv.org/abs/2509.04059`.

[27] Guang Yang, Muru Zhang, Lin Qiu, Yanming Wan, and Noah A. Smith. Toward a more complete omr solution. In *Proceedings of the 25th International Society for Music Information Retrieval Conference*, pages 930–937. ISMIR, November 2024. doi: 10.5281/zenodo.14877483. URL `https://doi.org/10.5281/zenodo.14877483`.

[28] Guang Yang, Victoria Ebert, Nazif Tamer, Brian Siyuan Zheng, Luiza Pozzobon, and Noah A. Smith. Legato: Large-scale end-to-end generalizable approach to typeset OMR, 2025. URL `https://arxiv.org/abs/2506.19065`.

A Pipeline Details

A.1 Context Management Mechanism

Our VLM is trained to model the probability $P(a_i \mid a_{<i}, u_i)$, where a_i represents the system-level ABC notation and u_i denotes the segmented image of the ith musical system. Because the preceding context $a_{<i}$ can be arbitrarily long while Legato 2 is constrained to a maximum context window of 2,048 tokens, we require a mechanism to manage the sequence length. Specifically, the tokens fed into the decoder are arranged in the following format:

$$\underbrace{\texttt{<|bos|>}\, a_{<i}}_{\text{left-truncated}}\ \texttt{<|bos|><|image|><|bos|>}\ \underbrace{a_i}_{\text{right-truncated}}$$

The `<|image|>` token is a special control token that implicitly instructs the cross-attention module to attend to the visual features, while the image input u_i is fed directly into the encoder. We apply left-truncation to the preceding system-level ABC sequence, as the most recent context is the most relevant for predicting the current system. Specifically, $a_{<i}$ is left-truncated to half of the model's

maximum context length (i.e., 1,024 tokens in Legato 2). While this truncation could potentially degrade the model's performance, the impact is minimal because the most critical musical context is highly localized. Furthermore, the multi-page recognition results presented in §4.2 demonstrate robust performance on lengthy documents.

A.2 Training Details

YOLO We fine-tuned a YOLOv8 medium model ($\sim$26 million parameters). Training used an input resolution of 800 pixels, batch size 16, and 100 epochs. The optimizer used was AdamW, using a base learning rate of 0.002, betas of 0.9 and 0.999, weight decay of 0.0005, a linear learning-rate decay schedule with final LR factor 0.01, and a 3-epoch warm-up. Mixed-precision training is enabled. Validation was performed after each epoch, and the best checkpoint was selected using validation performance. This model was trained on one Nvidia A100 GPU for 6 hours.

VLM Following Yang et al. [28], our VLM adopts the LLAMA-3.2-11B-VISION [1] architecture. We retain the pre-trained vision encoder but reduce the size of the decoder, training it from scratch. Since only the small decoder and the multimodal projection layer are trained, the total number of trainable parameters is 113.7M. We employ a learning rate of 3×10^{-4} and a batch size of 32. The model is trained on the PDMX-Synth training split for five epochs. Optimization is performed using the AdamW optimizer ($\beta_1 = 0.9, \beta_2 = 0.99, \epsilon = 10^{-6}$) paired with a cosine decay learning rate scheduler and a warm-up ratio of 0.03. This model was trained with 180 Nvidia L40 GPU hours.

A.3 Rule-Based ABC Converter

The ABC converter is a rule-based algorithm designed to translate between standard ABC notation and our system-level ABC representation.

During the processing of the PDMX-Synth training data, we convert standard ABC notation into a system-level format. Given the standard ABC notation a^* , we first parse the composition-level header information and then extract individual voices from the content. For each voice, we segment the notation into systems according to the linebreak symbol (`$`). Finally, for a given system, we merge its corresponding voices and append the layout information (`%%score`). The composition-level header is retained exclusively at the beginning of the first system, a_1.

During inference, after our VLM generates the system-level ABC representations, we convert them back into standard ABC notation. Specifically, we first parse the sequence of system-level outputs $\langle a_1, a_2, \cdots, a_s \rangle$. We then retrieve the composition-level header from a_1. Subsequently, we extract the final layout information from the last system, a_s, and insert it into the composition-level header. Finally, for each voice, we concatenate the content across all systems. Note that this merging process occasionally fails due to ill-formed syntax generated by the VLM. In such cases, we iteratively discard the terminal system-level ABC outputs until the merge succeeds.

B Audiveris Preprocessing

For Audiveris-based optical music recognition, the original score images were preprocessed by resampling them to a higher effective resolution before transcription. The original sample images were tagged at approximately 72 DPI, which cannot be processed by Audiveris. Each image was therefore scaled toward a target effective resolution of 300 DPI. Note that since the input images were rasters, upscaling does not recover details that were not originally present. Rather, more pixels are added to increase DPI so that Audiveris can process it. To avoid exceeding Audiveris's practical page-size limits, output images were capped at 20 million pixels, which caused some large pages to be clamped to a slightly lower effective DPI (250–300). This preprocessing was limited to resolution normalization; no deskewing, denoising, thresholding, or binarization was applied. The purpose was to improve Audiveris's ability to detect musical structure from already-rasterized inputs.

C Evaluating YOLO Segmentation and Robustness

C.1 YOLO Segmentation Performance

We evaluate our YOLO model on a test set of 204 images, sampled from the same data sources as the training set. The model achieved a validation bounding box mAP of

$$\text{mAP}_{50:95} = 0.804$$

with

$$\text{mAP}_{50} = 0.990, \quad \text{Precision} = 0.990, \quad \text{Recall} = 0.994.$$

Note that these metrics likely underestimate the true capability of the YOLO model, since we stretch the left and right side of each bounding box to segment the page, and so the YOLO model only needs to predict the top and bottom borders of the box.

C.2 Robustness to YOLO Errors

To measure the robustness of Legato 2 against YOLO segmentation errors, we conduct the following experiments. For each page in the Rendered OpenScore String Quartets dataset, we artificially pick, with uniform probability, to either (1) merge two neighboring boxes, or (2) delete one of the identified bounding boxes. Then, we make our VLM predict the corrupted samples. We find that after the corruptions, Legato 2 was able to achieve an OMR-NED of 47.4, scoring better than Audiveris and Gemini 3.1 Pro, showing that even in the rare case of YOLO failure, Legato 2 can still transcribe with competitive performance.

D System Prompts

D.1 Sheet Music Recognition

We use the following prompts for evaluating Gemini 3.1 Pro on standard OMR benchmarks:

```
You will be given an image of a sheet of music.
Transcribe it into valid ABC 2.1 notation. Try your best to transcribe and
↪  make a reasonable guess if the image is not clear.
You will be given three in context examples of image-transcription pairs
Output only the ABC (no explanations), preferably inside a single ```abc
↪  fenced block.
IMPORTANT: NEVER give outputs like: 'Unable to transcribe from the provided
↪  image due to insufficient resolution/clarity.' If you can't tell, give
↪  your best guess.
!!!ALWAYS OUTPUT VALID ABC, DO NOT GIVE ANY ENGLISH OUTPUT. GIVE YOUR BEST
↪  GUESS IF YOU ARE NOT SURE!!!

Transcribe this score to ABC.
{icl_1_img}
{ICL_1_ABC}


Transcribe this score to ABC.
{icl_2_img}
{ICL_2_ABC}


Transcribe this score to ABC.
{icl_3_img}
{ICL_3_ABC}


Transcribe this score to ABC.
{image}
```

D.2 Sheet Music Understanding

D.2.1 MusiXQA

Baseline As a starting point for our investigation, our baseline prompt follows the format introduced by Chen et al. [7].

```
You are an AI assistant specializing in Optical Music Recognition
(OMR) and Optical Character Recognition (OCR) for music sheets.
Your task is to accurately analyze images of music notation and
provide structured responses to visual question-answering (VQA)
tasks.
You will process printed music sheet images and answer both
OCR and OMR-related questions with high accuracy.
1 OCR-Based Tasks (Text Extraction)
- Extract the title and composer from the music sheet.
- Identify and extract the tempo marking (in BPM).
- Recognize and return the time signature.
- Extract explicitly labeled chord names from the sheet.
2 OMR-Based Tasks (Music Symbol Recognition)
- Identify the number and type of clefs (e.g., treble, bass).
- Count the number of bars (measures) in the music sheet.
- Recognize repeat sections based on notation symbols.
- Extract note durations (e.g., quarter, eighth, dotted notes, tied notes)
↪  for a given bar.
- Identify note pitches within a given bar.
- Return a structured representation of pitch, duration for a given bar in
↪  JSON string of list of python dictionaries without indent.
- Use kern representation for duration.
- If no explicit chord labels exist, infer the chord based on the notes in
↪  a given bar.
3 Response Format
- Provide structured, precise, and as concise as possible answers.
- Use structured JSON output without indent, when applicable for easy
↪  parsing.
4 Additional Considerations
- Ensure responses are notation-aware, considering key signatures,
↪  accidentals, and note relationships.
- Handle staff line separation correctly, ensuring multi-clef scores are
↪  properly analyzed.
- Avoid hallucinating missing information; only extract what is present in
↪  the image.
Follow music engraving conventions and OMR best practices to provide
↪  accurate, structured answers. If the requested information is not
↪  visible in the image, respond with '"Information not found"' instead of
↪  making assumptions.

Question:
{question}
"""
```

Transcription as Context Provider To investigate the capability of VLMs when given transcriptions from OMR models as additional context, we provide a transcription from either Legato 1 [28] or Legato 2 as optional context.

```
You are an AI assistant specializing in Optical Music Recognition
(OMR) and Optical Character Recognition (OCR) for music sheets.
Your task is to accurately analyze images of music notation and
provide structured responses to visual question-answering (VQA)
tasks.
You will process printed music sheet images and answer both
```

```
OCR and OMR-related questions with high accuracy.
1 OCR-Based Tasks (Text Extraction)
- Extract the title and composer from the music sheet.
- Identify and extract the tempo marking (in BPM).
- Recognize and return the time signature.
- Extract explicitly labeled chord names from the sheet.
2 OMR-Based Tasks (Music Symbol Recognition)
- Identify the number and type of clefs (e.g., treble, bass).
- Count the number of bars (measures) in the music sheet.
- Recognize repeat sections based on notation symbols.
- Extract note durations (e.g., quarter, eighth, dotted notes, tied notes)
↪  for a given bar.
- Identify note pitches within a given bar.
- Return a structured representation of pitch, duration for a given bar in
↪  JSON string of list of python dictionaries without indent.
- Use kern representation for duration.
- If no explicit chord labels exist, infer the chord based on the notes in
↪  a given bar.
3 Response Format
- Provide structured, precise, and as concise as possible answers.
- Use structured JSON output without indent, when applicable for easy
↪  parsing.
4 Additional Considerations
- Ensure responses are notation-aware, considering key signatures,
↪  accidentals, and note relationships.
- Handle staff line separation correctly, ensuring multi-clef scores are
↪  properly analyzed.
- Avoid hallucinating missing information; only extract what is present in
↪  the image.
Follow music engraving conventions and OMR best practices to provide
↪  accurate, structured answers. If the requested information is not
↪  visible in the image, respond with '"Information not found"' instead of
↪  making assumptions.

Question:
{question}
Optional generated transcription for this image:
{transcription}
You may use the transcription to answer the question.
```

Judge Prompt G-Acc is calculated using LLM-as-a-Judge techniques; we provide the prompt for the judge model here:

```
You are grading a model answer for a music-sheet QA task.
Given Question, GroundTruth, and Prediction, output:
- 1 if Prediction is semantically correct for the Question
- 0 otherwise

Rules:
- Be strict on musically meaningful symbols (pitch letter, accidental,
↪  octave, duration, tie, chord root/quality).
- Ignore insignificant formatting differences (extra whitespace, minor
↪  punctuation/casing).
- Output only one character: 1 or 0.
```

D.2.2 SSMR-Bench

Baseline We provide here the prompt used to test VLMs on SSMR-Bench.

```
Answer this multiple-choice music reading question.
Output exactly one capital letter: A, B, C, or D.
```

```
Do not provide any explanation.
Question: {example['question']}
```

Transcription as Context Provider As with MusiXQA (§D.2.1) we augment our baseline prompt with optional context in the form of transcriptions from Legato 1 [28] or Legato 2.

```
Answer this multiple-choice music reading question.
Output exactly one capital letter: A, B, C, or D.
Do not provide any explanation.
You may optionally use the provided Legato transcriptions alongside the
↪  image(s).
The transcriptions may be imperfect, so prefer the image if they conflict.
Question: {example['question']}
ABC context:
{example['abc_context']}

Main score image:
[inline image: sheet_music_img]

Optional Legato transcription for main score:
{main_score_legato_transcription}

Choices:
A. {choice_A}
B. {choice_B}
C. {choice_C}
D. {choice_D}
```

In the case where the answer choices themselves contain images, we use the following prompt:

```
Answer this multiple-choice music reading question.
Output exactly one capital letter: A, B, C, or D.
Do not provide any explanation.
You may optionally use the provided Legato transcriptions alongside the
↪  image(s).
The transcriptions may be imperfect, so prefer the image if they conflict.
Question: {example['question']}
ABC context:
{example['abc_context']}

Main score image:
[inline image: sheet_music_img]

Optional Legato transcription for main score:
{main_score_legato_transcription}

Choices:

A.
[inline image: choice_A]
Optional Legato transcription for A:
{choice_A_legato_transcription}

B.
[inline image: choice_B]
Optional Legato transcription for B:
{choice_B_legato_transcription}

C.
[inline image: choice_C]
Optional Legato transcription for C:
```

```
{choice_C_legato_transcription}

D.
[inline image: choice_D]
Optional Legato transcription for D:
{choice_D_legato_transcription}
```

E Evaluation Metric

E.1 OMR Normalized Edit Distance (OMR-NED)

We employ the standard OMR-NED metric for evaluating OMR quality. Compared to traditional metrics such as Symbol Error Rate, OMR-NED offers several benefits. Rather than comparing predicted and reference encodings purely as strings, OMR-NED evaluates discrepancies at the level of rendered musical notation, enabling assessment across a broad range of musical symbols and layout-dependent features. Specifically, the metric represents a composition as a sequence of measures $(m_1, m_2, \cdots, m_n)$ and computes the sequence edit distance to the ground truth. Each measure is defined as a set of musical symbols, and the distance between two measures is computed using a set edit distance. By quantifying the error as the normalized number of insertions and deletions required to transform a predicted sheet music into the reference, OMR-NED offers a robust and efficient measure of transcription quality.

E.2 OMR-NED Breakdown by Error Category

OMR-NED sums edit operations over error categories, so the score can be split into the contribution of each category. Table 5 shows this split for Legato 2 on OpenScore String Quartets. Note errors account for most of the remaining error on both rendered and camera-captured pages. The largest increase from rendered to camera-captured pages comes from inserted or deleted measures and staves, which indicates that the model loses track of the score structure more often on photographed pages. Accidental and barline errors each contribute less than 0.3 points.

Table 5: **OMR-NED of Legato 2 on OpenScore String Quartets, split by error category.** Each entry is the category's contribution to OMR-NED; rows sum to the totals in Table 1 up to rounding. Category names follow musicdiff.

Error category	Rendered	Camera
Wrong note	11.5	17.5
Entire measure insertion/deletion	1.9	6.6
Entire staff insertion/deletion	0.4	2.0
Wrong slur	0.7	1.2
Wrong direction	0.5	1.0
Wrong dynamic	0.4	0.5
Wrong flag/beam	0.3	0.5
Other categories	1.3	2.2
Total	17.1	31.6

E.3 Embedded Text Evaluation Metric

We evaluate embedded text recognition using different metrics depending on the output format of each model. For OMR systems that produce symbolic music, we first convert all predictions to ABC notation. We then extract textual fields from the ABC header, including title, composer, and inline textual annotations within the music body, which are represented in ABC as quoted strings. Location indicators are removed, and the resulting text items are concatenated in a canonical order using double underscores as separators. We compute character error rate (CER) between the predicted and reference strings using Levenshtein distance. The reference text is obtained by converting the ground-truth MusicXML of each page to ABC, with the title, composer, and staff names corrected by

Table 6: **Ablation study on model architecture.** D-Dec and W-Dec refer to the Deep Decoder and Wide Decoder configurations. Qwen refers to using QWEN-3-VL-2B-INSTRUCT model architecture.

Dataset (# of Pages)	OMR-NED ↓			
	D-Dec	W-Dec	Qwen	Default
PDMX-Synth Test Set (411)	**21.4**	24.3	25.6	23.5
Rendered OpenScore String Quartets (252)	18.9	22.6	20.0	**17.1**
Camera OpenScore String Quartets (252)	33.2	41.5	43.7	**31.6**
Rendered OpenScore Lieder (64)	**26.7**	28.0	29.4	27.6
Camera OpenScore Lieder (64)	**43.0**	47.7	45.0	43.6
IMSLP Piano Scores (32)	**33.1**	41.2	38.4	34.2

hand to match the text printed on that page. For text recognition, we run Audiveris 5.11.0 with its OCR enabled; the 2 of 252 pages on which it produces no output are counted as empty predictions.

For OCR systems, outputs consist of unordered bounding boxes with associated text. Because these predictions cannot be reliably converted into the same canonical ordering, we evaluate them with an unordered character error rate (Unordered-CER). Specifically, we compute a set edit distance between predicted and reference text boxes, where inserting or deleting a box costs the length of its text and substituting one box for another costs their Levenshtein distance. This allows OCR outputs to be evaluated without imposing an arbitrary reading order.

F Additional Ablation Studies

F.1 Model Architecture

Following the Legato 1 architecture, our default model is initialized from LLAMA-3.2-11B-VISION and configured with a reduced decoder size. We investigate the potential of scaling up the model capacity and exploring alternative architectures. Specifically, we examine: (i) whether a deeper or wider decoder enhances performance, and (ii) whether architectures utilizing different pretrained vision encoders yield improvements. To address (i), we construct two variants—one with double the number of layers, and another with a larger hidden dimension. For both variants, the total number of trainable parameters is controlled to approximately twice that of the default model. To address (ii), we adapt QWEN-3-VL-2B-INSTRUCT [3] by reducing its decoder to match our default size, while maintaining an identical training pipeline. As shown in Table 6, despite the much higher computational cost of the wide and deep decoder variants, only the deep decoder yields an improvement in performance over the default model. However, we note that this improvement is primarily localized to the PDMX-Synth test set, whereas gains on the remaining datasets are marginal. This discrepancy points to a distribution shift between the synthetic PDMX-Synth data and the real-world datasets, indicating that scaling up the decoder capacity largely results in better fitting the synthetic training distribution. Furthermore, the model with QWEN-3-VL-2B-INSTRUCT architecture underperforms the default model. We think that this is because its architecture assigns an individual image token to each small image patch, which rapidly exhausts the available context window, particularly when processing large images.

F.2 Data Distribution Shift

To investigate the hypothesis that the distribution gap between the training and evaluation datasets hinders the performance gains of the scaled models, we plot the distributions of two key structural metrics across all datasets: (1) the number of measures per page, and (2) the number of musical elements per measure. We remove outliers, defined as any datapoint 1.5 times the inter-quartile range above or below the first or third quartile, before generating plots. As illustrated in Figure 5, the evaluation datasets contain significantly more measures per page (with the exception of OpenScore Lieder, where vocal staves are masked, leaving blank spaces) and a higher density of musical elements per measure. Because PDMX-Synth is rendered from PDMX—a symbolic music dataset sourced from the MuseScore forum—the visual density of the resulting images is often low and sparse, frequently featuring empty pages or simplistic, single-voice compositions. This fundamental

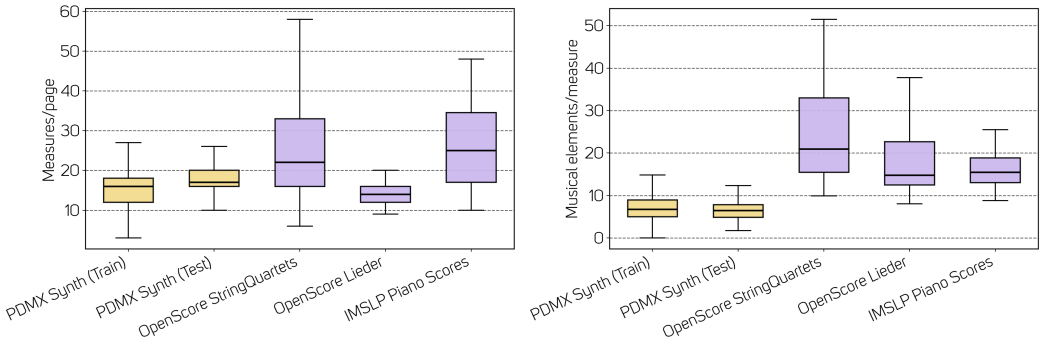


Figure 5: **Data distribution shift.** We plot the distribution of number of measures per page (left) and number of elements per measure (right). denotes PDMX-Synth and denotes evaluation datasets. Sheet music in evaluation datasets is denser than PDMX-Synth.

Table 7: **Evaluation of model with naive BPE tokenizer.** Performances are significantly worse than both Legato 1 and our proposed models, indicating that Legato 1's training approach is ineffective when directly applied to textual data.

Dataset	Raw 2048 Tokenizer
PDMX-Synth Test	48.3
Camera OpenScore String Quartets	92.1
Rendered OpenScore String Quartets	87.4
Camera OpenScore Lieder	74.5
Rendered OpenScore Lieder	64.4
IMSLP Piano Scores	77.2

difference in content density accounts for the observed distribution shift. Consequently, we argue that the curation of large-scale, real-world datasets is the primary bottleneck for training more robust OMR models and narrowing this generalization gap.

F.3 Naive BPE Tokenizer

We present the evaluation results for a model utilizing a tokenizer naively trained on the text-inclusive PDMX-Synth dataset. The goal of this model was to achieve text awareness by directly applying Legato's training recipe. However, as shown in Table 7, it performs significantly worse than both Legato 1 and 2, indicating that Legato 1's training approach is ineffective when directly applied to textual data.

G Validation Performances

We detail the grid search conducted to identify the optimal inference hyperparameters for each of our ablation models, and our baseline. These hyperparameters are selected based on model performance on our manually curated validation set. Table 8 reports the validation OMR-NED for Legato 2. Table 8 reports the validation OMR-NED for our baseline Legato 1 model. Note that for Legato 1, the setting with the lowest validation loss also happened to be the setting reported in the original paper (10 beams with 1.1 repetition penalty) [28]. In the following subsections, we present the validation performance of all models evaluated in our ablation studies.

G.1 System Segmentation and Byte Fallback

In our ablation studies concerning system segmentation and byte fallback, we report the performance of three models in the main text: Legato 1, Legato 1 with system segmentation (`SS`), and Legato 1 with both system segmentation and byte fallback (`SS + BF`). For the Legato 1 model, the validation

Table 8: **OMR-NED of Legato 1 and Legato 2 on validation set.**

	Repetition Penalty		
# Beams	1.0	1.1	1.2
1	72.8	71.2	72.1
2	70.7	70.4	70.4
5	71.3	70.9	71.8
10	71.8	**70.2**	70.6

(a) Legato 1

	Repetition Penalty		
# Beams	1.0	1.1	1.2
1	50.8	**46.8**	56.2
2	51.2	49.0	47.9
5	54.5	50.6	51.6
10	58.5	54.3	52.6

(b) Legato 2

performance is reported in Table 8. For the other two models, we report the validation performance in Table 9.

Table 9: **OMR-NED of models for system segmentation and byte fallback on validation set.**

	Repetition Penalty		
# Beams	1.0	1.1	1.2
1	57.9	49.6	55.3
2	50.7	50.7	49.4
5	51.1	50.8	**48.5**
10	51.1	51.2	48.8

(a) Legato 1 + `SS`

	Repetition Penalty		
# Beams	1.0	1.1	1.2
1	57.1	50.0	52.6
2	51.2	49.9	**48.5**
5	50.4	50.2	52.1
10	55.1	53.6	50.6

(b) Legato 1 + `SS` + `BF`

G.2 Vocabulary Size

In Table 10, we report the validation OMR-NED scores for the model with a vocabulary size of 8,192, which corresponds to the rightmost column of Table 4. The model with a vocabulary size of 2,048 is identical to the "Legato 1 + `SS` + `BF`" configuration, and its validation errors are reported in Table 9. Finally, the model with a vocabulary size of 4,096 represents our final model, whose validation errors are detailed in Table 8. Note that all evaluated models utilize a text-aware tokenizer featuring byte fallback; the vocabulary size is the sole independent variable.

Table 10: **OMR-NED of model with vocabulary size of 8,192 on validation set.**

	Repetition Penalty		
# Beams	1.0	1.1	1.2
1	56.0	54.4	62.0
2	55.5	**52.8**	53.8
5	56.2	55.4	55.3
10	56.2	54.9	54.5

G.3 Model Architecture

Table 11 presents the validation performance for the models used in the architecture ablation studies in Appendix F.1. The evaluated configurations include a deep decoder, a wide decoder, and the QWEN-3-VL-2B-INSTRUCT architecture. Consistent with our earlier approach, we select the hyperparameters that achieve the lowest validation OMR-NED.

Table 11: **OMR-NED of models with various architecture on validation set.** D-Dec and W-Dec refer to the Deep Decoder and Wide Decoder configurations. Qwen refers to using QWEN-3-VL-2B-INSTRUCT model architecture.

	Repetition Penalty		
# Beams	1.0	1.1	1.2
1	62.5	59.9	61.6
2	61.1	61.7	61.1
5	61.6	**59.8**	60.1
10	64.7	62.8	60.7

(a) Qwen

	Repetition Penalty		
# Beams	1.0	1.1	1.2
1	55.1	52.1	53.3
2	51.6	50.6	51.8
5	50.5	51.4	50.9
10	55.1	52.1	**46.4**

(b) D-Dec

	Repetition Penalty		
# Beams	1.0	1.1	1.2
1	60.0	**55.3**	**55.3**
2	56.2	55.9	**55.3**
5	58.3	**55.3**	**55.3**
10	61.8	59.6	57.5

(c) W-Dec

H Qualitative Examples

H.1 Sheet Music Understanding

In Figure 6, we present an example from SSMR-Bench alongside Gemini's reasoning process under different context-provider settings. Note that for this example, we ask Gemini to output its reasoning process, instead of directly outputting the answer like we do during the actual evaluation on SSMR-Bench. The prompt asks for the correct barline placement within the provided sheet music. When relying exclusively on the image, Gemini successfully recognizes the time signature but fails to extract more fine-grained visual details. Consequently, it hallucinates five note groups, assigning an arbitrary number of beats to each. As for Legato 1, in this specific example, it generates superfluous notes at the end of its transcription. As a result, when Legato 1 serves as the context provider, Gemini is misled; it hallucinates eight groups, although it correctly calculates the durations for them (e.g., groups 4 and 5). In contrast, Legato 2 accurately transcribes the sheet music into ABC notation. By grounding its reasoning in this precise symbolic transcription, Gemini correctly parses the groups and computes the proper beat placements. In general, this example demonstrates how Legato 2 effectively provides reliable symbolic context to assist frontier language models in downstream sheet music understanding tasks.

I Limitations and Future Work

In this paper, we introduced a novel OMR approach that segments sheet music into individual systems and processes them autoregressively. This method achieves state-of-the-art performance in both single- and multi-page recognition, while uniquely integrating the extraction of embedded textual metadata. Furthermore, we demonstrated that Legato 2 effectively serves as an upstream context provider, supplying precise symbolic data to general-purpose VLMs to enhance their reasoning on domain-specific musical tasks.

Despite these advances, Legato 2 has notable limitations. First, resource constraints precluded extensive ablation studies on individual pipeline components. Specifically, we have yet to formally quantify how segmentation errors from the YOLO-based detector propagate and affect downstream reasoning when the OMR output is used as VLM context. Second, model performance is constrained by distribution shifts between synthetic training data and the diverse reality of physical sheet music,

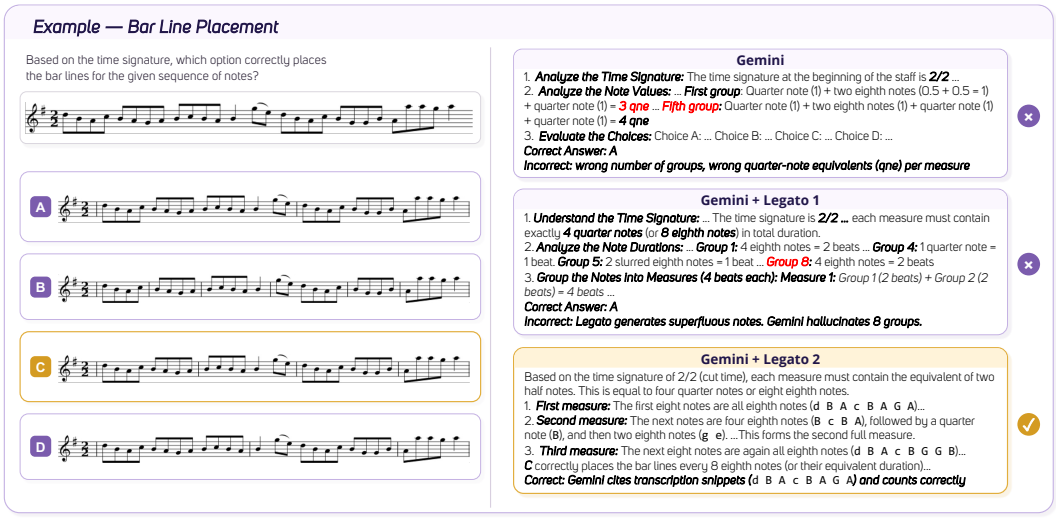


Figure 6: **Qualitative example from SSMR-Bench.** Left: The question and corresponding choices, both accompanied by sheet music. Right: Gemini's reasoning process and final answer under different context providers.

which varies widely in engraving style, scan quality, historical notation, and handwritten markings. These domain shifts can degrade both transcription accuracy and the reliability of the resulting symbolic context.

To address these limitations, future work must broaden training distributions and establish diagnostic benchmarks to precisely measure error propagation in downstream tasks. Our results show that, for printed sheet music, combining system-level decomposition with a specialized OMR model improves frontier VLMs. In future work, we envision a framework where diverse, specialized models—including OMR, audio transcription, and structural analysis—act as integrated context providers, assisting frontier VLMs in solving complex, open-ended musical problems.

Additionally, from a broader societal perspective, better OMR systems may also increase the risk of music piracy by making it easier to convert copyrighted sheet music scans or photographs into clean, editable symbolic formats that can be redistributed, modified, or rendered into new editions. This could reduce the ability of composers, publishers, and archives to control access to copyrighted works, especially if OMR tools are integrated into large-scale digitization or search pipelines. To mitigate this risk, releases should include clear acceptable-use policies, copyright-aware dataset curation, provenance tracking for generated transcriptions, and mechanisms such as watermarking or metadata retention where appropriate. Platforms deploying OMR at scale could also limit bulk transcription of copyrighted materials, support takedown workflows, and encourage use on public-domain, licensed, or user-owned scores.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We carefully write the paper to make sure that every claim in the abstract and introduction is addressed by the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We include a limitations and future works section

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper's main contributions are not theoretical.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We fully disclose all the training data and procedures we ran our experiments with.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Our code, model, and evaluation data are released at `https://github.com/guang-yng/legato`.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We fully disclose all our experimental settings, training details, and hyperparameter tuning procedures.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide information on confidence intervals for our main realistic, long context transcription experiment. For our page-level results, we use deterministic decoding and so the variance would be 0.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide training details for the YOLO segmenter and the main model.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our experiments are conducted on fully open datasets and do not breach the ethics guidelines in any way.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Our paper discusses music recognition and understanding, both of which are of high value to interdisciplinary communities. We also discuss the potential negative societal consequences of piracy enabled by better OMR models in our Limitations and Future Work section.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our pipeline has little risk of misuse, and the datasets are all sourced from open datasets.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have properly cited all the datasets we used and this paper. All the data are sourced from open domains.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We properly document all new inventions and assets of this paper.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: We do not use crowdsourcing or human participation in our paper.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: We do not research human subjects in this paper.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We do not use LLM in a core setting for our research

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.