FURTHER STEPS TOWARDS A STANDARD TESTBED FOR OPTICAL MUSIC RECOGNITION

Jan Hajič jr.[1] **Jiří Novotný**[2] **Pavel Pecina**[1] **Jaroslav Pokorný**[2]
[1] Charles University, Institute of Formal and Applied Linguistics, Czech Republic
[2] Charles University, Department of Software Engineering, Czech Republic
hajicj@ufal.mff.cuni.cz, novotny@ksi.mff.cuni.cz

ABSTRACT

Evaluating Optical Music Recognition (OMR) is notoriously difficult and automated end-to-end OMR evaluation metrics are not available to guide development. In "Towards a Standard Testbed for Optical Music Recognition: Definitions, Metrics, and Page Images", Byrd and Simonsen recently stress that a benchmarking standard is needed in the OMR community, both with regards to data and evaluation metrics. We build on their analysis and definitions and present a prototype of an OMR benchmark. We do not, however, presume to present a complete solution to the complex problem of OMR benchmarking. Our contributions are: (a) an attempt to define a multi-level OMR benchmark dataset and a practical prototype implementation for both printed and handwritten scores, (b) a corpus-based methodology for assessing automated evaluation metrics, and an underlying corpus of over 1000 qualified relative cost-to-correct judgments. We then assess several straightforward automated MusicXML evaluation metrics against this corpus to establish a baseline over which further metrics can improve.

1. INTRODUCTION

Optical Music Recognition (OMR) suffers from a lack of evaluation standards and benchmark datasets. There is presently no publicly available way of comparing various OMR tools and assessing their performance. While it has been argued that OMR can go far even in the absence of such standards [7], the lack of benchmarks and difficulty of evaluation has been noted on multiple occasions [2, 16, 21]. The need for end-to-end system evaluation (at the final level of OMR when musical content is reconstructed and made available for further processing), is most pressing when comparing against commercial systems such as PhotoScore, [1] SmartScore [2] or SharpEye [3]: these typically perform as "black boxes", so evaluating on the level of individual symbols requires a large amount of human effort for assessing symbols and their locations, as done by Bellini et al. [18] or Sapp [19].

OMR systems have varying goals, which should be reflected in evaluation. Helping speed up transcription should be measured by some cost-to-correct metric; a hypothetical automated score interpretation system could require accurate MIDI, but does not need to resolve all slurs and other symbols; digitizing archive scores for retrieval should be measured by retrieval accuracy; etc. We focus on evaluating transcription, as it is most sensitive to errors and most lacking in evaluation metrics.

Some OMR subtasks (binarization, staff identification and removal, symbol localization and classification) have natural ways of evaluating, but the end-to-end task does not: it is difficult to say how good a semantic representation (e.g., MusicXML) is. Manually evaluating system outputs is costly, slow and difficult to replicate; and aside from Knopke and Byrd [12], Szwoch [20] and Padilla et al. [21], we know of no attempts to even define an automatic OMR evaluation metric, much less define a methodology for assessing how well it actually evaluates.

Our contribution does not presume to define an entire evaluation standard. Instead, we propose a robust, cumulative, data-driven methodology for *creating* one. We collect human preference data that can serve as a gold standard for comparing MusicXML automated evaluation metrics, mirroring how the BLEU metric and its derivatives has been established as an evaluation metric for the similarly elusive task of assessing machine translation based on agreement with human judgements [17]. This "evaluating evaluation" approach is inspired by the Metrics track of the Workshop of Statistical Machine Translation competition (WMT) [3, 5, 14]. To collect cost-to-correct estimates for various notation errors, we generate a set of synthetically distorted "recognition outputs" from a set of equally synthetic "true scores". Then, annotators are shown examples consisting of a true score and a pair of the distorted scores, and they are asked to choose the simulated recognition output that would take them less time to correct.

Additionally, we provide an OMR bechmark dataset prototype with ground truth at the symbol and end-to-end levels.

The main contributions of our work are:

- A corpus-based "evaluating evaluation" methodol-

[1] http://www.neuratron.com/photoscore.htm
[2] http://www.musitek.com/index.html
[3] http://www.visiv.co.uk

ogy that enables iteratively improving, refining and fine-tuning automated OMR evaluation metrics.
- A corpus of 1230 human preference judgments as gold-standard data for this methodology, and assessments of example MusicXML evaluation metrics against this corpus.
- Definitions of ground truths that can be applied to Common Western Music Notation (CWMN) scores.
- MUSCIMA++. A prototype benchmark with multiple levels of ground truth that extends a subset of the CVC-MUSCIMA dataset [9], with 3191 annotated notation primitives.

The rest of this paper is organized as follows: in Sec. 2, we review the state-of-the-art on OMR evaluation and datasets; in Sec. 3, we describe the human judgment data for developing automated evaluation metrics and demonstrate how it can help metric development. In Sec. 4, we present the prototype benchmark and finally, in Sec. 5, we summarize our findings and suggest further steps to take. [4]

2. RELATED WORK

The problem of evaluating OMR and creating a standard benchmark has been discussed before [7, 10, 16, 18, 20] and it has been argued that evaluating OMR is a problem as difficult as OMR itself. Jones et al. [10] suggest that in order to automatically measure and evaluate the performance of OMR systems, we need (a) a standard dataset and standard terminology, (b) a definition of a set of rules and metrics, and (c) definitions of different ratios for each kind of errors. The authors noted that distributors of commercial OMR software often claim the accuracy of their system is about 90 %, but provide no information about how that value was estimated.

Bellini et al. [18] manually assess results of OMR systems at two levels of symbol recognition: low-level, where only the presence and positioning of a symbol is assessed, and high-level, where the semantic aspects such as pitch and duration are evaluated as well. At the former level, mistaking a beamed group of 32nds for 16ths is a minor error; at the latter it is much more serious. They defined a detailed set of rules for counting symbols as recognized, missed and confused symbols. The symbol set used in [18] is quite rich: 56 symbols. They also define *recognition gain*, based on the idea that an OMR system is at its best when it minimizes the time needed for correction as opposed to transcribing from scratch, and stress *verification cost*: how much it takes to verify whether an OMR output is correct.

An extensive theoretical contribution towards benchmarking OMR has been made recently by Byrd and Simonsen [7]. They review existing work on evaluating OMR systems and clearly formulate the main issues related to evaluation. They argue that the complexity of CWMN is the main reason why OMR is inevitably problematic, and suggest the following stratification into levels of difficulty:

1. Music on one staff, strictly monophonic,
2. Music on one staff, polyphonic,
3. Music on multiple staves, but each strictly monophonic, with no interaction between them,
4. "Pianoform": music on multiple staves, one or more having multiple voices, and with significant interaction between and/or within staves.

They provide 34 pages of sheet music that cover the various sources of difficulty. However, the data does not include handwritten music and no ground truth for this corpus is provided.

Automatically evaluating MusicXML has been attempted most significantly by Szwoch [20], who proposes a metric based on a top-down MusicXML node matching algorithm and reports agreement with human annotators, but how agreement was assessed is not made clear, no implementation of the metric is provided and the description of the evaluation metric itself is quite minimal. Due to the complex nature of MusicXML (e.g., the same score can be correctly represented by different MusicXML files), Szwoch also suggests a different representation may be better than comparing two MusicXML files directly.

More recently, evaluating OMR with MusicXML outputs has been done by Padilla et al. [21]. While they provide an implementation, there is no comparison against gold-standard data. (This is understandable, as the paper [21] is focused on recognition, not evaluation.) Aligning MusicXML files has also been explored by Knopke and Byrd [12] in a similar system-combination setting, although not for the purposes of evaluation. They however make an important observation: stems are often mistaken for barlines, so the obvious simplification of first aligning measures is not straightforward to make.

No publicly available OMR dataset has ground truth for end-to-end recognition. The CVC-MUSCIMA dataset for staffline identification and removal and writer identification by Fornés et al. [9] is most extensive, with 1000 handwritten scores (50 musicians copying a shared set of 20 scores) and a version with staves removed, which is promising for automatically applying ground truth annotations across the 50 versions of the same score. Fornés et al. [8] have also made available a dataset of 2128 clefs and 1970 accidentals.

The HOMUS musical symbol collection for online recognition [11] consists of 15200 samples (100 musicians, 32 symbol classes, 4-8 samples per class per musician) of individual handwritten musical symbols. The dataset can be used for both online and offline symbol classification.

A further dataset of 3222 handwritten and 2521 printed music symbols is available upon request [1]. Bellini et al. [18] use 7 selected images for their OMR assessment; unfortunately, they do not provide a clear description of the database and its ground truth, and no more information is publicly available. Another staffline removal dataset is Dalitz's database, [5] consisting of 32 music pages that cov-

[4] All our data, scripts and other supplementary materials are available at `https://github.com/ufal/omreval` as a git repository, in order to make it easier for others to contribute towards establishing a benchmark.

[5] `http://music-staves.sourceforge.net`

ers a wide range of music types (CWMN, lute tablature, chant, mensural notation) and music fonts. Dalitz et al. [6] define several types of distortion in order to test the robustness of the different staff removal algorithms, simulating both image degradation and page deformations. These have also been used to augment CVC-MUSCIMA.

There are also large sources such as the Mutopia project [6] with transcriptions to LilyPond and KernScores [7] with HumDrum. The IMSLP database [8] holds mostly printed scores, but manuscripts as well; however, as opposed to Mutopia and KernScores, IMSLP generally only provides PDF files and no transcription of their musical content, except for some MIDI recordings.

3. EVALUATING EVALUATION

OMR lacks an automated evaluation metric that could guide development and reduce the price of conducting evaluations. However, an automated metric for OMR evaluation needs itself to be evaluated: does it really rank as better systems that *should* be ranked better?

Assuming that the judgment of (qualified) annotators is considered the gold standard, the following methodology then can be used to assess an automated metric:

1. Collect a corpus of annotator judgments to define the expected gold-standard behavior,
2. Measure the agreement between a proposed metric and this gold standard.

This approach is inspired by machine translation (MT), a field where comparing outputs is also notoriously difficult: the WMT competition has an evaluation track [5, 14], where automated MT metrics are evaluated against human-collected evaluation results, and there is ongoing research [3, 15] to design a better metric than the current standards such as BLEU [17] or Meteor [13]. This methodology is nothing surprising; in principle, one could machine-learn a metric given enough gold-standard data. However: how to best design the gold-standard data and collection procedure, so that it encompasses what we in the end want our application (OMR) to do? How to measure the quality of such a corpus – given a collection of human judgments, how much of a gold standard is it?

In this section, we describe a data collection scheme for human judgments of OMR quality that should lead to comparing automated metrics.

3.1 Test case corpus

We collect a corpus C of *test cases*. Each test case $c_1 \dots c_N$ is a triplet of music scores: an "ideal" score I_i and two "mangled" versions, $P_i^{(1)}$ and $P_i^{(2)}$, which we call *system outputs*. We asked our K annotators $a_1 \dots a_K$ to choose the less mangled version, formalized as assigning $r_a(c_i) = -1$ if they preferred $P_i^{(1)}$ over $P_i^{(2)}$, and $+1$ for the opposite preference. The term we use is to "rank" the predictions. When assessing an evaluation metric against this corpus, the test case rankings then constrain the space of well-behaved metrics. [9]

The exact formulation of the question follows the "cost-to-correct" model of evaluation of [18]:

> "Which of the two system outputs would take you less effort to change to the ideal score?"

3.1.1 What is in the test case corpus?

We created 8 ideal scores and derived 34 "system outputs" from them by introducing a variety of mistakes in a notation editor. Creating the system outputs manually instead of using OMR outputs has the obvious disadvantage that the distribution of error types does not reflect the current OMR state-of-the-art. On the other hand, once OMR systems change, the distribution of corpus errors becomes obsolete anyway. Also, we create errors for which we can assume the annotators have a reasonably accurate estimate of their own correction speed, as opposed to OMR outputs that often contain strange and syntactically incorrect notation, such as isolated stems. Nevertheless, when more annotation manpower becomes available, the corpus should be extended with a set of actual OMR outputs.

The ideal scores (and thus the derived system outputs) range from a single whole note to a "pianoform" fragment or a multi-staff example. The distortions were crafted to cover errors on individual notes (wrong pitch, extra accidental, key signature or clef error, etc.: micro-errors on the semantic level in the sense of [16, 18]), systematic errors within the context of a full musical fragment (wrong beaming, swapping slurs for ties, confusing staccato dots for noteheads, etc.), short two-part examples to measure the tradeoff between large-scale layout mistakes and localized mistakes (e.g., a four-bar two-part segment, as a perfect concatenation of the two parts into one vs. in two parts, but with wrong notes) and longer examples that constrain the metric to behave sensibly at larger scales.

Each pair of system outputs derived from the same ideal score forms a test case; there are 82 in total. We also include 18 control examples, where one of the system outputs is identical to the ideal score. A total of 15 annotators participated in the annotation, of whom 13 completed all 100 examples; however, as the annotations were voluntary, only 2 completed the task twice for measuring intra-annotator agreement.

3.1.2 Collection Strategy

While Bellini et al. [18] define how to count individual errors at the level of musical symbols, assign some cost to each kind of error (miss, add, fault, etc.) and define the overall cost as composed of those individual costs, our methodology does not assume that the same type of error has the same cost in a different *context*, or that the overall cost can be computed from the individual costs: for instance, a sequence of notes shifted by one step can be in

[6] http://www.mutopiaproject.org
[7] http://humdrum.ccarh.org
[8] http://imslp.org

[9] We borrow the term "test case" from the software development practice of unit testing: test cases verify that the program (in our case the evaluation metric) behaves as expected on a set of inputs chosen to cover various standard and corner cases.

most editors corrected simultaneously (so, e.g., clef errors might not be too bad, because the entire part can be transposed together).

Two design decisions of the annotation task merit further explanation: why we ask annotators to compare examples instead of rating difficulty, and why we disallow equality.

Ranking. The practice of ranking or picking the best from a set of possible examples is inspired by machine translation: Callison-Burch et al. have shown that people are better able to agree on which proposed translation is better than on how good or bad individual translations are [4]. Furthermore, ranking does not require introducing a cost metric in the first place. Even a simple 1-2-3-4-5 scale has this problem: how much effort is a "1" on that scale? How long should the scale be? What would the relationship be between short and long examples?

Furthermore, this annotation scheme is fast-paced. The annotators were able to do all the 100 available comparisons within 1 hour. Rankings also make it straightforward to compare automated evaluation metrics that output values from different ranges: just count how often the metric agrees with gold-standard ranks using some measure of monotonicity, such as Spearman's rank correlation coefficient.

No equality. It is also not always clear which output would take less time to edit; some errors genuinely are equally bad (sharp vs. flat). These are also important constraints on evaluation metrics: the costs associated with each should not be too different from each other. However, allowing annotators to explicitly mark equality risks overuse, and annotators using *underqualified* judgment. For this first experiment, therefore, we elected not to grant that option; we then interpret disagreement as a sign of uncertainty and annotator uncertainty as a symptom of this genuine tie.

3.2 How gold is the standard?

All annotators ranked the control cases correctly, except for one instance. However, this only accounts for elementary annotator failure and does not give us a better idea of systematic error present in the experimental setup. In other words, we want to ask the question: if all annotators are performing to the best of their ability, **what level of uncertainty should be expected under the given annotation scheme?** (For the following measurements, the control cases have been excluded.)

Normally, inter-annotator agreement is measured: if the task is well-defined, i.e., if a gold standard *can* exist, the annotators will tend to agree with each other towards that standard. However, usual agreement metrics such as Cohen's κ or Krippendorf's α require computing *expected agreement*, which is difficult when we do have a subset of examples on which we do *not* expect annotators to agree but cannot *a priori* identify them. We therefore start by defining a simple agreement metric L. Recall:

- C stands for the corpus, which consists of N examples $c_1 \dots c_N$,
- A is the set of K annotators $a_1 \dots a_K$, $a, b \in A$;
- r_a is the *ranking function* of an annotator a that assigns +1 or -1 to each example in c,

$$L(a,b) = \frac{1}{N} \sum_{c \in C} \frac{|r_a(c) + r_b(c)|}{2}$$

This is simply the proportion of cases on which a and b agree: if they disagree, $r_a(c)+r_b(c) = 0$. However, we expect the annotators to disagree on the genuinely uncertain cases, so some disagreements are not as serious as others. To take the existence of legitimate disagreement into account, we modify $L(a,b)$ to weigh the examples according to how certain the other annotators $A \setminus \{a,b\}$ are about the given example. We define weighed agreement $L_w(a,b)$:

$$L_w(a,b) = \frac{1}{N} \sum_{c \in C} w^{(-a,b)}(c) \frac{|r_a(c) + r_b(c)|}{2}$$

where $w^{(-a,b)}$ is defined for an example c as:

$$w^{(-a,b)}(c) = \frac{1}{K-2} \left| \sum_{a' \in A \setminus a,b} r_{a'}(c) \right|$$

This way, it does not matter if a and b disagree on cases where no one else agrees either, but if they disagree on an example where there is strong consensus, it should bring the overall agreement down. Note that while maximum achievable $L(a,b)$ is 1 for perfectly agreeing annotators (i.e., all the sum terms equal to 1), because $w(c) \leq 1$, the maximum achievable $L_w(a,b)$ will be less than 1, and furthermore depends on the choice of a and b: if we take notoriously disagreeing annotators away from the picture, the weights will increase overall. Therefore, we finally adjust $L_w(a,b)$ to the proportion of maximum achievable $L_w(a,b)$ for the given (a,b) pair, which is almost the same as $L_w(a,a)$ with the exception that *b must also be excluded from computing the weights*. We denote this maximum as $L^*_w(a,b)$, and the adjusted metric $\hat{L}_w$ is then:

$$\hat{L}_w(a,b) = L_w(a,b) / L^*_w(a,b)$$

This metric says: "What proportion of achievable weighed agreement has been actually achieved?" The upper bound of $\hat{L}_w$ is therefore 1.0 again; the lower bound is agreement between two randomly generated annotators, with the humans providing the consensus.

The resulting pairwise agreements, with the lower bound established by averaging over 10 random annotators, are visualized in Fig. 1. The baseline agreement $\hat{L}_w$ between random annotators weighed by the full human consensus was close to 0.5, as expected. There seems to be one group of annotators relatively in agreement (green and above, which means adjusted agreement over 0.8), and then several individuals who disagree with everyone – including among themselves (lines 6, 7, 8, 11, 12, 14).

Interestingly, most of these "lone wolves" reported significant experience with notation editors, while the group more in agreement not as much. We suspect this is because

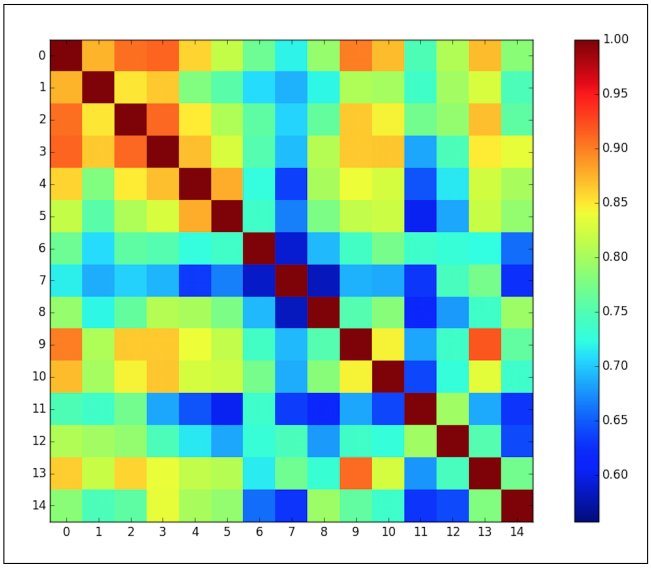

Figure 1. Weighed pairwise agreement. The cell $[a, b]$ represents $\hat{L}_w(a, b)$. The scale goes from the average random agreement (ca. 0.55) up to 1.

with increasing notation editor experience, users develop a personal editing style that makes certain actions easier than others by learning a *subset* of the "tricks" available with the given editing tools – but each user learns a different subset, so agreement on the relative editing cost suffers. To the contrary, inexperienced users might not have spent enough time with the editor to develop these habits.

3.3 Assessing some metrics

We illustrate how the test case ranking methodology helps analyze these rather trivial automated MusicXML evaluation metrics:

1. Levenshtein distance of XML canonization (**c14n**),
2. Tree edit distance (**TED**),
3. Tree edit distance with `<note>` flattening (**TEDn**),
4. Convert to LilyPond + Levenshtein distance (**Ly**).

c14n. Canonize the MusicXML file formatting and measure Levenshtein distance. This is used as a trivial baseline.

TED. Measure Tree Edit Distance on the MusicXML nodes. Some nodes that control auxiliary and MIDI information (`work`, `defaults`, `credit`, and `duration`) are ignored. Replacement, insertion, and deletion all have a cost of 1.

TEDn. Tree Edit Distance with special handling of `note` elements. We noticed that many errors of TED are due to the fact that while deleting a note is easy in an editor, the edit distance is higher because the `note` element has many sub-nodes. We therefore encode the notes into strings consisting of one position per `pitch`, `stem`, `voice`, and `type`. Deletion cost is fixed at 1, insertion cost is 1 for non-note nodes, and 1 + length of code for notes. Replacement cost between notes is the edit distance between their codes; replacement between a note and non-note costs 1 + length of code; between non-notes costs 1.

Metric	r_s	$\hat{r}_s$	ρ	$\hat{\rho}$	τ	$\hat{\tau}$
c14n	0.33	0.41	0.40	0.49	0.25	0.36
TED	0.46	0.58	0.40	0.50	0.35	0.51
TEDn	**0.57**	**0.70**	0.40	0.49	**0.43**	**0.63**
Ly	0.41	0.51	0.29	0.36	0.30	0.44

Table 1. Measures of agreement for some proposed evaluation metrics.

Ly. The LilyPond [10] file format is another possible representation of a musical score. It encodes music scores in its own LaTeX-like language. The first bar of the "Twinkle, twinkle" melody would be represented as `d'8[ d'8] a'8[ a'8] b'8[ b'8] a'4 |` This representation is much more amenable to string edit distance. The **Ly** metric is Levenshtein distance on the LilyPond import of the MusicXML system output files, with all whitespace normalized.

For comparing the metrics against our gold-standard data, we use nonparametric approaches such as Spearman's r_s and Kendall's τ, as these evaluate monotonicity without assuming anything about mapping values of the evaluation metric to the $[-1, 1]$ range of preferences . To reflect the "small-difference-for-uncertain-cases" requirement, however, we use Pearson's ρ as well [14]. For each way of assessing a metric, its maximum achievable with the given data should be also estimated, by computing how the metric evaluates the consensus of one group of annotators against another. We randomly choose 100 splits of 8 vs 7 annotators, compute the average preferences for the two groups in a split and measure the correlations between the average preferences. The expected upper bounds and standard deviations estimated this way are:

- $r_s^* = 0.814$, with standard dev. 0.040
- $\rho^* = 0.816$, with standard dev. 0.040
- $\tau^* = 0.69$, with standard dev. 0.045

We then define $\hat{r_s}$ as $\frac{r_s}{r_s^*}$, etc. Given a cost metric $\mathcal{L}$, we get for each example $c_i = (I_i, P_i^{(1)}, P_i^{(2)})$ the cost difference $\ell(c_i) = \mathcal{L}(I_i, P_i^{(1)}) - \mathcal{L}(I_i, P_i^{(2)})$ and pair it with the gold-standard consensus $r(c_i)$ to get pairwise inputs for the agreement metrics.

The agreement of the individual metrics is summarized in Table 1. When developing the metrics, we did *not* use the gold-standard data against which metric performance is measured here; we used only our own intuition about how the test cases should come out.

4. BENCHMARK DATASET PROTOTYPE

A benchmark dataset should have ground truth at levels corresponding to the standard OMR processing stages, so that sub-systems such as staff removal, or symbol localization can be compared with respect to the end-to-end pipeline they are a part of. We also suspect handwritten music will remain an open problem much longer than printed music. Therefore, we chose to extend the **CVC-MUSCIMA** dataset instead of Byrd and Simonsen's pro-

[10] `http://www.lilypond.org`

posed test bed [7] because of the extensive handwritten data collection effort that has been completed by Fornés et al. and because ground truth for staff removal and binarization is already present. At the same time, CVC-MUSCIMA covers all the levels of notational complexity from [7], as well as a variety of notation symbols, including complex tuples, less common time signatures (5/4), C-clefs and some symbols that could very well expose the differences between purely symbol-based and more syntax-aware methods (e.g., tremolo marks, easily confused for beams). We have currently annotated symbols in printed scores only, with the perspective of annotating the handwritten scores automatically or semi-automatically.

We selected a subset of scores that covers the various levels of notational complexity: single-part monophonic music (F01), multi-part monophonic music (F03, F16), and pianoform music, primarily based on chords (F10) and polyphony (F08), with interaction between staves.

4.1 Symbol-level ground truth

Symbols are represented as bounding boxes, labeled by symbol class. In line with the low-level and high-level symbols discussed by [7], we differentiate symbols at the level of *primitives* and the level of *signs*. The relationship between primitives and signs can be one-to-one (e.g., clefs), many-to-one (composite signs: e.g. notehead, stem, and flag form a note), one-to-many (disambiguation: e.g., a sharp primitive can be part of a key signature, accidental, or an ornament accidental), and many-to-many (the same beam participates in multiple beamed notes, but each beamed note also has a stem and notehead). We include individual numerals and letters as notation primitives, and their disambiguation (tuplet, time signature, dynamics...) as signs.

We currently define 52 primitives plus letters and numerals, and 53 signs. Each symbol can be linked to a MusicXML counterpart. [11] There are several groups of symbols:

- Note elements (noteheads, stems, beams, rests...)
- Notation elements (slurs, dots, ornaments...)
- Part default (clefs, time and key signatures...)
- Layout elements (staves, brackets, braces...)
- Numerals and text.

We have so far annotated the primitive level. There are 3191 primitives marked in the 5 scores. Annotation took about 24 hours of work in a custom editor.

4.2 End-to-end ground truth

We use MusicXML as the target representation, as it is supported by most OMR/notation software, actively maintained and developed and available under a sufficiently permissive license. We obtain the MusicXML data by manually transcribing the music and postprocessing to ensure each symbol has a MusicXML equivalent. Postprocessing mostly consists of filling in default barlines and correcting staff grouping information. Using the MuseScore notation editor, transcription took about 3.5 hours.

[11] The full lists of symbol classes are available in the repository at `https://github.com/ufal/omreval` under `muscima++/data/Symbolic/specification`.

5. CONCLUSIONS AND FUTURE WORK

We proposed a corpus-based approach to assessing automated end-to-end OMR evaluation metrics and illustrated the methodology on several potential metrics. A gold standard annotation scheme based on assessment of relative cost-to-correct of synthetic "system outputs" was described that avoids pre-defining any cost metric, and the resulting corpus of 1230 human judgments was analyzed for inter-annotator agreement, taking into account the possibility that the compared system outputs may not be clearly comparable. This preference-based setup avoids the need to pre-define any notion of cost, requires little annotator training, and it is straightforward to assess an evaluation metric against this preference data.

Our results suggest that the central assumption of a single ground truth for preferences among a set of system outputs is weaker with increasing annotator experience. To make the methodology more robust, we recommend:

- Explicitly control for experience level; do not assume that more annotator experience is better.
- Measure actual cost-to-correct (in time and interface operations) through a notation editor, to verify how much human *estimation* of this cost can be relied on.
- Develop models for computing expected agreement for data where the annotations may legitimately be randomized (the "equally bad" cases). Once expected agreement can be computed, we can use more standard agreement metrics.

The usefulness of the test case corpus for developing automated evaluation metrics was clear: the TEDn metric that outperformed the others by a large margin was developed through analyzing the shortcomings of the TED metric on individual test cases (before the gold-standard data had been collected). As Szwoch [20] suggested, modifying the representation helped. However, if enough human judgments are collected, it should even be possible to sidestep the difficulties of hand-crafting an evaluation metric through machine learning; we can for instance try learning the insertion, deletion, and replacement costs for individual MusicXML node types.

An OMR environment where different systems can be meaningfully compared, claims of commercial vendors are verifiable and progress can be measured is in the best interest of the OMR community. We believe our work, both on evaluation and on a dataset, constitutes a significant step in this direction.

6. ACKNOWLEDGMENTS

This research is supported by the Czech Science Foundation, grant number P103/12/G084. We would also like to thank our annotators from the Janáček Academy of Music and Performing Arts in Brno and elsewhere, and Alicia Fornés for providing additional background and material for the CVC-MUSCIMA dataset.

7. REFERENCES

[1] 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. *Int J Multimed Info Retr*, 1(3):173–190, Mar 2012.

[2] Baoguang Shi, Xiang Bai, and Cong Yao. An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition. *CoRR*, abs/1507.05717, 2015.

[3] Ondřej Bojar, Miloš Ercegovčević, Martin Popel, and Omar F. Zaidan. A Grain of Salt for the WMT Manual Evaluation. *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 1–11, 2011.

[4] Chris Callison Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. (Meta-) Evaluation of Machine Translation. *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, 2007.

[5] Chris Callison Burch, Philipp Koehn, Christof Monz, Kay Peterson, Mark Przybocki, and Omar F. Zaidan. Findings of the 2010 Joint Workshop on Statistical Machine Translation and Metrics for Machine Translation. *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 17–53, 2010.

[6] Christoph Dalitz, Michael Droettboom, Bastian Pranzas, and Ichiro Fujinaga. A Comparative Study of Staff Removal Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):753–766, May 2008.

[7] Donald Byrd and Jakob Grue Simonsen. Towards a Standard Testbed for Optical Music Recognition: Definitions, Metrics, and Page Images. *Journal of New Music Research*, 44(3):169–195, 2015.

[8] Alicia Fornés, Josep Lladós, Gemma Sánchez, and Horst Bunke. Writer Identification in Old Handwritten Music Scores. *Proceeedings of Eighth IAPR International Workshop on Document Analysis Systems*, pages 347–353, 2008.

[9] Alicia Fornés, Anjan Dutta, Albert Gordo, and Josep Lladós. CVC-MUSCIMA: a ground truth of handwritten music score images for writer identification and staff removal. *International Journal on Document Analysis and Recognition (IJDAR)*, 15(3):243–251, 2012.

[10] Graham Jones, Bee Ong, Ivan Bruno, and Kia Ng. Optical Music Imaging: Music Document Digitisation, Recognition, Evaluation, and Restoration. *Interactive Multimedia Music Technologies*, pages 50–79, 2008.

[11] Jorge Calvo-Zaragoza and Jose Oncina. Recognition of Pen-Based Music Notation: The HOMUS Dataset. *22nd International Conference on Pattern Recognition*, Aug 2014.

[12] Ian Knopke and Donald Byrd. Towards Musicdiff : A Foundation for Improved Optical Music Recognition Using Multiple Recognizers. *International Society for Music Information Retrieval Conference*, 2007.

[13] Alon Lavie and Abhaya Agarwal. Meteor: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, 2007.

[14] Matouš Macháček and Ondřej Bojar. Results of the WMT14 Metrics Shared Task. *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 293–301, 2014.

[15] Matouš Macháček and Ondřej Bojar. Evaluating Machine Translation Quality Using Short Segments Annotations. *The Prague Bulletin of Mathematical Linguistics*, 103(1), Jan 2015.

[16] Michael Droettboom and Ichiro Fujinaga. Microlevel groundtruthing environment for OMR. *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR 2004)*, pages 497–500, 2004.

[17] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318, 2002.

[18] Pierfrancesco Bellini, Ivan Bruno, and Paolo Nesi. Assessing Optical Music Recognition Tools. *Computer Music Journal*, 31(1):68–93, Mar 2007.

[19] Craig Sapp. OMR Comparison of SmartScore and SharpEye. `https://ccrma.stanford.edu/~craig/mro-compare-beethoven`, 2013.

[20] Mariusz Szwoch. Using MusicXML to Evaluate Accuracy of OMR Systems. *Proceedings of the 5th International Conference on Diagrammatic Representation and Inference*, pages 419–422, 2008.

[21] Victor Padilla, Alan Marsden, Alex McLean, and Kia Ng. Improving OMR for Digital Music Libraries with Multiple Recognisers and Multiple Sources. *Proceedings of the 1st International Workshop on Digital Libraries for Musicology - DLfM '14*, pages 1–8, 2014.