An environment for machine pedagogy: Learning how to teach computers to read music

Gabriel Vigliensoni, Jorge Calvo-Zaragoza, and Ichiro Fujinaga
Schulich School of Music, McGill University, CIRMMT
Montréal, QC, Canada
{gabriel.vigliensonimartin, jorge.calvozaragoza, ichiro.fujinaga}@mcgill.ca

ABSTRACT

We believe that in many machine learning systems it would be effective to create a pedagogical environment where both the machines and the humans can incrementally learn to solve problems through interaction and adaptation.

We are designing an optical music recognition (OMR) workflow system where human operators can intervene to correct and teach the system at certain stages so that they can learn from the errors and the overall performance can be improved progressively as more music scores are processed.

In order to instantiate this pedagogical process, we have developed a series of browser-based interfaces for the different stages of our OMR workflow: image preprocessing, music symbol recognition, musical notation recognition, and final representation construction. In most of these stages we integrate human input with the aim of teaching the computers to improve the performance.

ACM Classification Keywords

H.5.5. Information interfaces and presentation (e.g., HCI): Sound and Music Computing—*Systems*; H.5.2. Information interfaces and presentation (e.g., HCI): User Interfaces—*User-centered design*; I.5.5. Pattern recognition: Implementation—*Interactive systems*

Author Keywords

Optical music recognition; interactive machine learning; artificial pedagogy;

INTRODUCTION

The idea of achieving intellectual development of a machine—or making computers smarter when creating algorithmic models, is not new. Alan Turing stated in the middle of the last century that the interaction of machines with humans would be necessary to adapt machines to the human standard and to achieve intellectual or performance parity with humans [14]. He envisioned that human guidance and feedback are desirable at various points of the machine's process of learning. However, Turing also anticipated that humans can act as a "brake" in fast machine computational processes, and so the places and levels of interaction between machines and humans should be studied and considered carefully.

One of the strengths of current learning machines lies in their ability to recognize complex patterns, provided that there is a large amount of labeled training data (ground truth). In cases where massive ground-truth datasets are not readily available, one solution is to incrementally and interactively train an adaptive system, with gradual exposure of new data. We argue that in these supervised adaptive learning environments, it is important to study how humans impart their knowledge to the machine: what are the different teaching methods (pedagogy) for the machine to achieve a desired performance and how do humans learn these effective strategies.

A Pedagogy for "Learning Machines"

In this paper, we propose the idea of a *pedagogy for learning machines* as the study of the methods and activities of teaching machines. This pedagogy is about creating an environment where humans can learn the art of how to teach machines running learning algorithms in an incremental learning process.

Turing also anticipated [14, p. 472] that learning machines

> will make mistakes at times, and at times they may make new and very interesting statements, and on the whole the output of them will be worth attention to the same sort of extent as the output of a human mind.

Following Turing's vision, we propose to exploit human skills and knowledge to teach machines to optimize their performance. In order to achieve this, we first need to understand how humans interact with a machine-learning component and then we need to build a clever workflow in order to take advantages of the intelligence of the human and the ability to perform fast calculations of the computer.

Bieger et al. proposed a conceptual framework for teaching intelligent systems [1]. They identified the constituent elements of that framework and stated that the interaction between *teachers* (e.g., a human actor) and *learners* (e.g., a computer system) has the goal of teaching the learning system to gain knowledge about something or about a specific task. As a pedagogical strategy, we hypothesize that by knowing the learner, and how the learner reacts to correction and new input, teachers can adapt their teaching tactics to improve the pedagogy.

The impact of human supervision in the loop of supervised machine learning workflows has been also empirically studied. For example, Fails and Olsen built a system for creating image

MILC '18, March 11, 2018, Tokyo, Japan

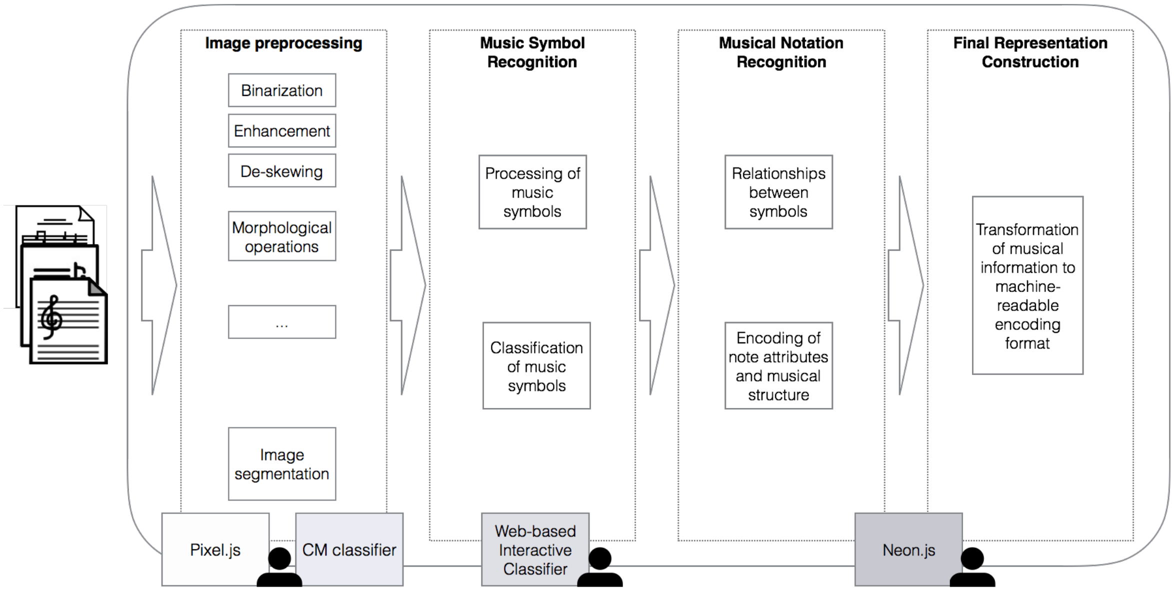
Image preprocessing
Binarization
Enhancement
De-skewing
Morphological operations
...
Image segmentation
Music Symbol Recognition
Processing of music symbols
Classification of music symbols
Musical Notation Recognition
Relationships between symbols
Encoding of note attributes and musical structure
Final Representation Construction
Transformation of musical information to machine-readable encoding format
Pixel.js
CM classifier
Web-based Interactive Classifier
Neon.js

Figure 1. Our end-to-end optical music recognition (OMR) workflow. Places where human intervention or human-entered data is needed are indicated by a human icon. The interfaces that humans use to visualize intermediate outputs of the system as well as to teach the system are shaded in grey.

classifiers and proposed the concept of *interactive machine learning* [7] for those environments where human teachers evaluate a model created by a learning machine, then edit the training data, and retrain the model according to their expert judgment to improve the performance of the system in the given task. Also, Fiebrink et al. studied evaluation practices of human actors interactively building supervised learning systems for gesture analysis [8].

In the next section we will detail how we have incorporated interactive checkpoints between human teachers and learning-machine systems in the development of an intelligent interface for encoding symbolic music, so that people can access cultural music heritage in an unprecedented manner.

TEACHING MACHINES HOW TO READ MUSIC SCORES

Our aim is to read and extract the content from digitized images of music documents. This process is called optical music recognition (OMR) and, despite more than 30 years of research, it remains to be a difficult problem. The slow development in OMR, particularly when dealing with older music documents, lies mainly in the large variability of musical sources (i.e., degradation, bleed-through, handwriting and notation style, among others). Since most approaches for extracting the musical content in the different layers of these manuscripts (e.g., musical notes, lyrics, staff lines, ornamental letters, etc.) have been developed using heuristic approaches, they rely on specific characteristics of the documents, and so these methods usually do not generalize well to music documents of a different type or era.

Fully manual OMR projects have been developed to overcome the large degree of variability in handwritten music scores. *Allegro*, for example, is a recently developed web-based crowdsourcing tool to transcribe and encode scores of a corpus of folk songs in Common Western Music Notation [2].

In order to work at a larger scale, we have taken a different route to OMR of Medieval and Renaissance music by using a machine learning-based approach. Instead of using heuristics and features that take advantage of specific characteristics of the documents, we teach the computer to classify the different elements in a music score by training it with a large number of examples for each category to be classified. The computer learns the regularities in these examples and creates a model of the data. Once a model is created, it is used to classify new examples that the computer has not yet seen. In other words, the computer *learns by examples* from the teacher.

In the standard OMR workflow, a human intervention is required to correct the errors generated by the automated process. Hence, we can take advantage of this by incorporating the previously corrected scores, as ground truth, for subsequent processing in an adaptive OMR system [9]. Pugin et al. experimented with this idea by building book-adaptive OMR models for music from microfilms [12]. Their experiments showed that human editing costs were substantially reduced and that the approach was especially well suited to handle the various degradation levels of music documents from typographic prints.

Our entire OMR workflow is depicted in Figure 1. This process is divided into four stages: *image preprocessing*, *music symbol recognition*, *musical notation recognition*, and *final representation reconstruction*. Digitized music scores are the input to the system and image preprocessing is applied to segment the constituent parts of the music document into layers. The recognition of the music symbols and the analysis of their relationship is achieved once the symbols are isolated and classified in the found layers. Finally, the retrieved musical information is encoded into a machine-readable format. We want to automate the process of extracting and digitizing the content of music scores. However, since we know that this process is not error free, and the errors generated in previous steps are carried forward to the next ones, we want to learn about the type of errors that the computer makes in each stage in order to: (i) provide better ground-truth data to improve the performance of the computer and (ii) let users (teachers) of the system understand and know where computers make mistakes in order to modify their behavior. To facilitate these tasks, we have implemented interactive checkpoints in the OMR workflow.

In the next two subsections we present the interactive interfaces we have developed for teaching the machine how to perform tasks in the first two stages of the OMR workflow.

Teaching machines for image segmentation

The first stage in our OMR workflow is *image preprocessing*. In this step, all pixels of the music score image are classified into different, pre-defined layers. Since we need training data as example for recognizing the different layers within an image, and creating ground truth from scratch is onerous and expensive, we have tested a few approaches for teaching the computer to perform the image preprocessing. So far, we have found that we can drastically reduce the time and effort needed to build ground truth by preprocessing a small number of images with a pre-existing model, usually a model learned in pages of similar characteristics. If no model achieves a meaningful result (i.e., if the output is not significantly better than random), we use a heuristic method. Then, we correct the coarse errors in the output of the previous stage with a pixel-level editor. In this step, we only spend the amount of time required to correct the major errors in order to have a reasonable set of corrected data, but not perfect. Finally, we iterate over the two previous steps until desired performance is achieved. We assume that perfect performance can not be achieved because, at pixel-level, even for humans it is hard to discriminate to what layer a pixel belongs to, especially at the boundaries.

Most image preprocessing techniques (based on heuristic or machine learning techniques) output a non-negligible amount of misclassified pixels, and so we developed *Pixel.js*, an open source, web-based, pixel-level classification application to correct the output of image segmentation processes [13]. We use this tool interactively with a convolutional neural network-based classifier [4], to create ground-truth data incrementally.

A conventional machine learning approach would work under the assumption that training and tuning will be performed a few times and need not be interactive. Hence, one reasonable strategy for improving supervised learning systems using human interaction is enabling the user to evaluate a model, then edit its training dataset based on his or her judgments of how the model should improve.

In our approach for image segmentation, the output of a learning system is used by a human teacher to further inform the system about the performance of the task. As a result, we are implementing an incremental and adaptive workflow based on tactics and strategies by which human teachers modify their actions depending on the outcome of a task given to learning machines. Preliminary implementations of these pedagogical strategies and actions have permitted us to considerably reduce the amount of effort when creating ground truth for image preprocessing for OMR by 40 percent. Importantly, we have not only obtained similar performance than using ground truth created from scratch, but we have also achieved higher user satisfaction [5]. We are currently increasing the iteration rate between training, correction, and retraining to see if even better results can be obtained.

Once the image preprocessing step has been performed, our OMR system outputs a number of image files *per* original score image, where each file contains a layer representing different type of musical information. For example, these layers may contain notes, staff lines, lyrics, annotations, or ornamental letters.

Teaching machines to recognize musical symbols

Our application for the second stage of the OMR workflow, music symbol recognition, is called *Interactive Classifier* (IC). IC is a web-based version of the Gamera classifier [6]. In this stage, the connected components of a specific layer of the original image are automatically grouped into *glyphs*. Then, a human teacher has to manually label the classes of a number of musical glyphs. IC will extract a set of features for describing each of the glyphs, and will classify the data based on the k-nearest neighbors classifier.

An attractive aspect of IC is that it can be used in an incremental learning fashion [11]. That is, as new data is entered by a human teacher into the system, IC will learn from new information and will accommodate the classes while preserving previously acquired knowledge without building a new classifier. In other words, the IC module for music symbol recognition is designed in a way that human teachers do not have to start over and over from scratch if new data or classes are entered into the learning system. Instead, they can use a previously trained classifier of glyphs and labels for the initial classification. Then, they can manually correct the glyphs that were misclassified and perform a reclassification. By repeating this process, IC will learn the corrections at each iteration and will build a better classifier until the teacher is satisfied with the results.

An interesting characteristic of IC is that how well the machine learns depends on how well the human teaches it. In fact, the human, through interaction, can gradually learn how to teach the machine better. Furthermore, human teachers do not need to know the intricacies of machine learning or need to be a domain expert because, for humans, these are simple visual tasks. We strongly believe that this interaction is important for developing a pedagogy for machines that learn.

Non-pedagogical OMR stages

The last two stages of our OMR workflow, *musical notation recognition* and *final representation construction* have a common interactive breakpoint for visualizing and correcting the output of the automatized OMR process. This human-driven checkpoint is embedded as a web-based interface called *Neume Editor Online* (Neon) [3]. Neon allows a user to inspect differences between the original music score image and the rendered version of the output of the OMR process. By visual inspection of the two overlaid scores, the user can observe their difference and manually add, edit, or delete music symbols in the browser. So far, however, corrections entered by the user are not fed back into the learning system, but they change the encoded music file output.

Our OMR workflow management system

Since our workflow requires a human operator to teach the learning system, we need to be able to create interactive checkpoints where the system stops a process and waits for user input. As a result, all the constituent parts of our OMR workflow are handled by Rodan, a distributed, collaborative, and networked adaptive workflow management system [10] that allows to specify *interactive* and *non-interactive* tasks.

FINAL REMARKS AND FUTURE WORK

The end goal of our project is not only to segment images and to recognize music symbols, but to create a final music representation that can be browsable and searchable by humans and computers by many different means. We envision this interface as an intelligent, music-score-searching tool for the 21st century. We are currently investigating the available infrastructure for creating this interface. Among them, we are making use of the International Image Interoperability Framework (IIIF) and IIIF manifests, which allows for the display of high-resolution images directly from the institutions having the rights to these images. We also make use of visualization interfaces (e.g., Diva.js document image viewer) that take advantage of IIIF and the Music Encoding Initiative (MEI) music encoding format (e.g., Verovio music notation engraving library). We hope that this infrastructure, in combination with the proper teaching strategies and tactics developed by human teachers in the interfaces for training the OMR system, will enable the end-to-end recognition and encoding of music from music score images.

ACKNOWLEDGMENTS

This research has been supported by the Social Sciences and Humanities Research Council of Canada. Important parts of this work used ComputeCanada's High Performance Computing resources.

REFERENCES

1. Jordi Bieger, Kristinn R. Thórisson, and Bas R. Steunebrink. 2017. The pedagogical pentagon: A conceptual framework for artificial pedagogy. In *International Conference on Artificial General Intelligence (Lecture Notes in Computer Science, vol 10414)*, Tom Everitt, Ben Goertzel, and Alexey Potapov (Eds.). Springer, Cham, 212–222.

2. Manuel Burghardt and Sebastian Spanner. 2017. Allegro: User-centered design of a tool for the crowdsourced transcription of handwritten music scores. In *Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage*. 15–20.

3. Gregory Burlet, Alastair Porter, Andrew Hankinson, and Ichiro Fujinaga. 2012. Neon. js: Neume Editor Online. In *Proceedings of the 13th International Society for Music Information Retrieval Conference*. 121–126.

4. Jorge Calvo-Zaragoza, Gabriel Vigliensoni, and Ichiro Fujinaga. 2017a. Pixel-wise binarization of musical documents with convolutional neural networks. In *Proceedings of the 15th IAPR International Conference on Machine Vision Applications*. Nagoya, Japan, 362–365.

5. Jorge Calvo-Zaragoza, Ké Zhang, Zeyad Saleh, Gabriel Vigliensoni, and Ichiro Fujinaga. 2017b. Music document layout analysis through machine learning and human feedback. In *Proceedings of 12th IAPR International Workshop on Graphics Recognition*. Kyoto, Japan.

6. Michael Droettboom, Karl MacMillan, and Ichiro Fujinaga. 2003. The Gamera framework for building custom recognition systems. In *Proceedings of the 2003 Symposium on Document Image Understanding Technologies*. Greenbelt, MD, 275–286.

7. Jerry Alan Fails and Dan R. Olsen Jr. 2003. Interactive machine learning. In *Proceedings of the 8th International Conference on Intelligent User Interfaces*. Miami, FL, 39–45.

8. Rebecca Fiebrink, Perry R. Cook, and Dan Trueman. 2011. Human model evaluation in interactive supervised learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 147–156.

9. Ichiro Fujinaga. 1996. *Adaptive optical music recognition*. PhD Dissertation. McGill University, Montréal, QC.

10. Andrew Hankinson. 2015. *Optical Music Recognition Infrastructure for Large-scale Music Document Analysis*. Ph.D. Dissertation. McGill University, Montréal, QC.

11. Robi Polikar, Lalita Upda, Satish S. Upda, and Vasant Honavar. 2001. Learn++: An incremental learning algorithm for supervised neural networks. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews* 31, 4 (2001), 497–508.

12. Laurent Pugin, John Ashley Burgoyne, Douglas Eck, and Ichiro Fujinaga. 2007. Book-adaptive and book-dependent models to accelerate digitization of early music. In *Proceedings of the NIPS Workshop on Music, Brain, and Cognition*. Whistler, BC, 1–8.

13. Zeyad Saleh, Ké Zhang, Jorge Calvo-Zaragoza, Gabriel Vigliensoni, and Ichiro Fujinaga. 2017. Pixel.js: Web-based pixel classification correction platform from ground truth creation. In *Proceedings of the 12th IAPR International Workshop on Graphics Recognition*. Kyoto, Japan.

14. Alan M. Turing. 2004. Intelligent machinery, a heretical theory. In *The Essential Turing: Seminal Writings in Computing, Logic, Philosophy, Artificial Intelligence, and Artificial Life: Plus The Secrets of Enigma*, B. Jack Copeland (Ed.). Oxford University Press, Oxford, United Kingdom, Chapter 12, 472–475.