Brendan Bassett
5/9/2022

Final Project Report: Optical Music Character Recognition

What is this project about?

Most of the sheet music that musicians typically use is digitized from a physical copy. Often it is a poorly taken photo of a book, or a photocopy, or even a copy of a copy. This means that the vast majority of musicians' digital collections are of poor quality and difficult to read. Originals are often in bad condition or not even present, so the solution is rarely to make a better scan in the first place. Wouldn't it be then nice to convert all of these poor copies to much cleaner, more readable versions?

We could use some image processing techniques to remove artifacts and improve binarization (conversion to black and white), but this only goes so far. Since the dawn of the digital era, musicians have sought a method to automate the "reading" of music itself. This would enable us to extract the musical information so that the sheet music can be reproduced in significantly better visual detail. Think of text and how each letter of the alphabet can be expressed as tiny packets of data, rather than storing an actual image of each letter. Musical information can be stored in similar ways, through formats like Music XML and MIDI. However, given the complexity of musical notation, conversion to one these formats from a physical document has proven to be very difficult.

What I am describing is called Optical Music Recognition (OMR). Only recently has it reached a level of accuracy which is meaningful. Some of the issues that OMR still struggles with is the spacial-relational nature of musical notation. Most notation is written on top of a set of lines call the staff, which has to be removed before individual characters can be recognized. Notes of the same pitch are located at the same vertical location on the staff lines (in general, but not always), so this information must be recognized and stored for analysis. Notes of the same duration can often be grouped together with bars, or with the flag up or down, or with no flag at all. Those details can sometimes be meaningful, and sometimes not. Worst of all for the software developer, much music only exists as handwritten copies, which makes each symbol appear wildly different from composer to composer.

Neural networks are the essential technology that OMR has needed to be remotely useful. It enables the software to accommodate for the vast diversity of musical notation. In this project I take a dataset of common musical characters, already removed from the staffs they are on, and seek to categorize them.

Brendan Bassett
5/9/2022

Which part is your own work/idea, which part is an implementation of other people's idea, which part is using some existing code/examples?

Most of this work, primarily the code, is my own. There are some sections loosely derived from class demos, but I have changed quite a lot to fit my own purposes. The bulk of the work that is not mine is the dataset, which is sourced from an academic paper, "Optical recognition of music symbols: A comparative study" by A. Rebelo and others (citation below). This was extremely valuable in that it enabled me to focus on the neural network itself, rather than spending an exorbitant amount of time on image preprocessing and segmentation. For my project, I pared down the dataset just a bit in order to focus on the categories of characters I felt were the most essential for sheet music notation in general.

The other concept that I used is outlined in "A new optical music recognition system based on combined neural networks" by Cuihong Wen and others. It uses several neural networks in parallel, and then tallies "votes" between each to more accurately categorize each character. This was very useful to me, as nearly every adjustment I made to the hyperparameters of my convolution neural network had miniscule effect on the end results. In the end I used three different machine learning models, and took a "vote" between the three to arrive at the final prediction.

List all the libraries/packages that you used.

This project uses OpenCV, Numpy, Tensorflow Keras, SKLearn, and Matplotlib.

Describe the machine learning algorithms or deep learning algorithms that are used in your project and the reason of selecting such models. Provide necessary discussion or analysis of performance change using different hyperparameters (if there was any).

I used primarily logistic regression and convolutional neural networks. Most OMR software of the 90s and 2000s were based on logistic regression, so I wanted to compare the performance of that with convolutional neural networks. It turns out that for this dataset, logistic regression is actually pretty accurate. However both of the convolution neural networks out performed the logistic regression model by a notable amount.

The convolution neural network 2 is quite simple, yet the most effective. There are only a few layers, with the convolution layer performing most of the meaningful work. After trying a 3x3, 5x5, and 7x7 filter I realized that the 7x7 did the best. This is surprising since the input images are so small. I figured that it would not be as effective as the others, but possibly the padding helps here. Only 4 epochs are needed in order to achieve the best results

The convolution neural network 1 is a test to see if I could get better results using more layers. It uses a 5x5 filter because the performance is still good (but not quite as good as 7x7) and I wanted some more variation between models for more interesting voting later on. This network is much slower to run, and its performance is similar to convolution neural network 2. It needs 5 epochs in order to achieve the best results.

The logistic regression model did shockingly well, though not as good as the other two models. In the final voting process I noticed that if one model differed from the others, it was usually this one.

Anything else that is not listed above and represents your design.

The final voting process is my extra, fun design, though it is fairly straightforward. The idea was to see if I could improve upon the already excellent results of cnn2. The voting process gives equal weight to each model (since the performance of each is so similar), and defaults to the the result from convolution neural network 2 if a three-way tie occurs. This only happened once in the entire testing data, and the cnn2 results were in fact the correct one. There were about 50 instances where predictions between the models did not fully match, and this voting process nearly always chose the correct answer.

Sources

(Included in project folder with dataset)

Cuihong Wen, Ana Rebelo, Jaime Cardoso (2009) “Optical recognition of music symbols: A comparative study” *International Journal on Document Analysis and Recognition (IJDAR) 13(1),* 19-31

Cuihong Wen, Ana Rebelo, Jing Zhanga, Jaime Cardoso (2014) “A new optical music recognition system based on combined neural network” *Pattern Recognition Letters 58,* 1-7

Cuihong Wen, Ana Rebelo, Jaime Cardoso (2012) “Music Symbols Segmentation through Recognition” *INESC TEC and Faculdade de Engenharia Universidade do Porto, Portugal*