Winter Break Journal

Hugh Zabriskie

January 2, 2016

1 Initial Experiments

1.1 12/28

The initial data cleaning and feature scraping is complete. The corpus consists of 326 4-voice chorales, each with very consistent formatting. Only chorales in '4/4' and '3/4' time were found. Each beat contains four notes. Currently, the features selected from each beat are as follows:

1. The melody pitch
2. The beat strength, appropriately weighted. In '4/4' for example, the beat weights are (1.0, 0.25, 0.5, 0.25).
3. The presence of a fermata, a binary feature. Indicates a cadence.
4. The distance until the next fermata (or cadence). This is 0 if there is a cadence on this beat.
5. The offset (in beats) from the beginning of the piece.
6. The offset (in beats) from the end of the piece.
7. The time signature.
8. The number of accidentals in the key signature (C major = 0, sharps are 1-7, flats are 8-14).
9. The mode of the key signature (i.e. 'major', 'minor').

The chorales were divided into training and test sets, initially with a 90/10 test split. In further experiments, a cross-validation set should be included. Each chorale is then written to a *separate* HDF5 file to be processed in torch. The isolation of data points into chorales is meant to facilitate the evaluation of results later on, so that the results might be re-converted to a musical score.

One question is: in order to convert from the results (an array of chord indices) into actual chords, the dictionary mapping indices to chords will need to be stored...not sure if this happens yet. Check metadata.md. From there the process is:

$$\text{indices} \rightarrow \text{MIDI tuple} \rightarrow \text{note tuple} \rightarrow \text{musical score (via music21)}$$

1.2 12/29

Goal: The goal is build a vanilla neural network using TORCH that can accept the batch of chorale files, evaluate them and update its parameters, and then output a final evaluation metric (even a rudimentary one, like overall chord accuracy). Also, to set up the process journal.
Result: The process journal (this journal) is up and running!

- `load.lua` created to consolidate the training chorales into one dataset, stored in `data/train.hdf5`.
- `eval.lua` created to create the model and criterion for the first experiment.
- I tried a range of parameters to see which improved performance. The parameters were epoch, embedding, hidden size, and learning rate, and they were evaluated based on negative log likelihood error. The best parameter set from the below options was $\{10, 10, 199.34527809156, 15, 0.001\}$.
  - embedding sizes = $\{10, 20, 30, 40, 80\}$
  - hidden sizes = $\{5, 15, 30, 60\}$
  - epochs = $\{10, 30\}$
  - learning rates = $\{0.1, 0.01, 0.001, 0.0001, 0.00001\}$

- The results were disappointing. While the training error converged, test error continued to increase. This suggests high variance, which could be adjusted by reducing the feature set (maybe eliminating `offset-start` as a feature?) or adding regularization.

- See `bach_code/experiment1.txt` for more results from initial tests.

1.3 12/30

Goal: The goal today is to see improve if we can improve on the results from the initial experiment performed yesterday. Potential methods are altering or deleting features, as well as adding regularization. A secondary goal is consider better methods for evaluation. **Result:**

- Based on some sense of success from the first trial I did about a month ago, I tried deleting the offset-start feature to see if that was causing some redundancy due to the offset-end feature. This showed no significant improvements.

- The next step was to play around with other means of dealing with high variance. I introduced dropout to add regularization. I also introduced a second hidden layer and experimented with several different hidden layer sizes. Again, no significant improvement was seen.

- I believe that the largest problem is that either the model needs to be segmented into sequential nets, or there needs to be a better representation for chords.

1.4 12/31

Goal: The goal today is to develop a new model that more closely models how chorales are composed today as 4-voice counterpoint exercises. This approach is based on both HARMONET (1992) and Schmidhuber (2015). There will be two LSTMs **Result:**

- Mostly a thinking day. While HARMONET is an old model, it is remarkably well thought-out, and papers within the last five year that make use of the chorales either do not discuss harmonization in any detail or do not use harmonization as a machine learning task. I think

using the general approach of HARMONET along with the improvement of LSTMs, will represent a strong baseline for moving forward with the inventions.

- Emailed Sasha - haven't heard back.

1.5 1/1

Goal: The goal today is to get the first experiment up and running, that being a harmonization task using the basic neural network (non-recurrent).

Result:

- `featurize.py` was completely re-done to be oriented towards the task of learning harmonization by way of roman numeral analysis. The whole script runs in about an hour. It's a big improvement, considering that the original approach had 1500 possibilities for chords, and it's been reduced to 57. This approach also recognizes the similarity between chords by identifying the harmony before the voicing.

- Currently, this only uses a 3-layer non-recurrent neural network. After playing around with parameters, I found some success with a learning rate of 0.00001, an embedding size of 20, and a hidden layer of size 75. The minimum error rate on the test set was around 86. A validation set is needed to test parameters, but that should only be done for the recurrent models. This is merely preliminary.

- This model will be saved so that it can be compared against once results are gathered for recurrent models.

- **Experiment 1 results:** Due to the lack of context in making decisions, the network is unable to develop well-structured harmonizations. It does, however, immediately note the importance of the tonic and dominant chords, and the vast majority of each produced harmonization is simply a combination of I and V. Occasionally, the minor i is chosen. There is also a snapshot of where it manages to exactly identify a few dominant harmonies (but misses several others). Generally, the machine learns to spot moments of *development* - that is, more

complicated harmonic passages - which it typically labels as a dominant chord. Thus, an emerging recognition of tonic-dominant motion is emerging with very little context!

- See `results/experiment1b.txt` for an example of a more successful test.
- After this task is working properly, I'll move on to learning how to select the inner voices.

1.6 1/2

Goal: The goal today is get an LSTM working *well* for the harmonization task. If this succeeds, we should see similar success on the task of inner voices.

Result:

-
-