Title: The Sound of Music

Who: Johnny Ren, Clara Guo, David Moon

Introduction: (Copied From DevPost)

After learning about interpretable CNN models in class, we were interested in applying one to a dataset where interpretable CNNs have not been used extensively. Specifically, we became interested in how such a model could be applied to spectrograms of sound files. We were motivated to explore this topic for two reasons: For one, we are interested in the parallels between image analysis and spectrogram audio analysis. For another, as music lovers, we find the idea of audio and genre classification interesting.

This [paper](https://openaccess.thecvf.com/content_cvpr_2018/papers/Zhang_Interpretable_Convolutional_Neural_CVPR_2018_paper.pdf) outlines an interpretable CNN framework that we are looking to modify for our project. The objective of this paper, which was developed for image classification, was to develop CNNs with filters that identify human-interpretable areas of feature detection/activation. The key difference between the feature maps of a traditional CNN and the one outlined in this paper lies in how loss between filter layers is calculated. In this interpretable framework, it is modified such that filters are incentivized to learn more interpretable parameters that lead to more discernible objects and features identified in an image. An additional strength to this paper is that additional annotations of object parts/text are unneeded under the assumption that repetitive shapes in various regions represent low-level textures.

We chose this paper because it provides a fundamental method for encouraging interpretable behavior in CNNs, which we can adapt to spectrogram classification. Our project ultimately seeks to develop an interpretable CNN model for classifying music of different genre using the GTZAN dataset.

(Update for Week of 11/31)

While we have implemented many of the primary components of the original paper, creating an effective CNN model, implementing model, and an attempted integration of their proprietary loss, we ultimately decided to also explore the work of a different paper as well. This [paper](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6627892/) outlines methods for applying class activation maps (CAMs) for model interpretability to medical data that we are adapting to our spectrograms.

While the previous paper focuses on training to make model activations interpretable, this new paper focuses on modifying the model architecture so that filter activations will be interpretable after combining them through different weighted combinations.

Methodology: (Copied From DevPost, with slight modifications)
We experimented with a 15-15-70 validation-test-training split, working with data with 10 different categories.

As for our specific model architecture, we had an iterative process in mind:
1. We will begin with a vanilla CNN architecture:
Spectrogram -> Convolution Layer 1 -> Convolution Layer 2 -> Convolution Layer 3 -> Output classification
2. Next, we develop the modifications outlined in the paper, namely, adapting loss so that our convolution filters are incentivized to learn meaningful features.
3. After verifying that our interpretability is okay, we approve accuracy:
a. We will consider experimenting with standard ways of improving performance (eg. changing number of filters, filter sizes, and hyperparameters, generally)
b. We will also consider specific CNN architectures that have been successful (such as the one outlined in Related Papers)
4. Iterate our model design until we reach our desired accuracy threshold and have a model with interpretable

The hardest part of this project will most likely include the adaptation of the modified loss function into our model. Additionally, the actual verification of interpretability (eg. conducting

qualitative/quantitative analyses of activated regions in our spectrograms) will likely be a cumbersome process. More broadly, balancing interpretability with performance could be tricky, since papers that have performed well on spectrogram analysis typically don't train for interpretability.

(Update for Week of 11/30)
While we are experimenting with second paper our methodology will be fairly similar. We will be experimenting with progressivly more complex and deep CNN networks to achieve high accuracy. However, with the CAM models, our model architecture must be qualified by the fact that there can only be one dense layer at the end (in order to allow for model interpretability)

The CAM model has a similar iterative methodology, with the crucial caveat that there can only be one dense layer at the end of our model, as this is necessary to generate interpretable CAM maps.

Results:
The baseCNN achieved an average training accuracy of 77.38% over 10 epochs with a validation accuracy of 67.08% and test accuracy of 68.23%. The maskedCNN achieved an average training accuracy of 83.01% over 10 epochs with a validation accuracy of 68.44% and test accuracy of 69.90%. The CAMCNN achieved an average training accuracy of 51.55% over 10 epochs with a validation accuracy of 68.23% and test accuracy of 68.12%. For visualizations and auralisations, we used 10 pieces of data—one per genre—to use across all models. For the base and masked CNN, we created png and wav representations of filter activations for filters 1, 8, 16, and 32 for the first convolutional layer. We also created png and wav representations of the deconvolved signal from the second convolutional layer. A trend we noticed is that the second layer picked up on more high level features, and our masks, when auralized further helped clarify discernible features in the songs. A good example of this is the deconvolved signal for country, which primarily picks up the underlying drum beat of the song. For the CAMCNN, we created class activation maps using a global average pooling layer. We also created wav files by elementwise multiplying the activation

map by the original spectrogram, and auralized them to actually hear what these activations sounded like.

Challenges:

The main challenge we had was in trying to implement the original paper we chose for interpretability. The paper had two main components: masking and a custom loss. The custom loss formula involved different complex variables whose calculations were not fully explained, especially since some of the variables were treated as constants that were updated occasionally, without specifying when they should be updated. The loss was extremely computationally expensive and which layers the custom loss were to be applied to was unclear as well. A less significant challenge was increasing the accuracy of our baseCNN to something we found acceptable. After consulting a paper detailing the impacts of filter size on spectrograms in particular, we began to experiment with different filter sizes and found that accuracy was extremely sensitive to filter size.

Reflection:

How do you feel your project ultimately turned out? How did you do relative to your base/target/stretch goals?

Overall, we were satisfied with the final test accuracies we achieved on all 3 models (~68% on the baseCNN, maskedCNN, and CAMCNN). We were able to achieve our target goals regarding interpretability by visualizing class activation maps (from the CAMCNN) and generating images/audio files of activation mappings from specific layers/filters (from the maskedCNN and baseCNN). While we were not able to meet our original stretch goal of implementing the interpretable loss outlined in this paper (due to ambiguity in its implementation), we were still able to implement the masking function.

Did your model work out the way you expected it to?

All three models did work as intended, achieving their own respective purposes (eg. for the CAMCNN, to generate valid class activation mappings). Specifically for the masked model,

we were satisfied with what bits of information each filter and feature map learned (i.e. for the country genre, some feature maps picked up the rhythm nicely). One thing to note is that for the CAMCNN, we had to adapt from the baseCNN model by adding additional conv2D layers and max pooling layers, as well as a deconvolution layer to implement global average pooling. For the CAMCNN, we were able to generate class activation maps, which told us which regions of a spectrogram were most important for

How did your approach change over time? What kind of pivots did you make, if any? Would you have done differently if you could do your project over again?

One significant pivot we made was in deciding against implementing the interpretable loss function in the original paper and adding global average pooling/class activation maps for interpretability instead. One thing our group would do differently would be to actively seek specifics on the implementation of a particular feature/function before fully committing to its usage in our own model.

What do you think you can further improve on if you had more time?

If there was more time, our group would experiment with other architectures (i.e. increasing the number of Conv2D layers, changing kernel sizes) to see if we could achieve higher test accuracies on our models. We would also add the ability for users to input their own songs and allow our trained model to classify it accordingly.

What are your biggest takeaways from this project/what did you learn?

The main takeaway we learned was that CNNs can be generalized beyond images to domains such as audio processing, highlighting their flexibility. We also learned that there are many forms of interpretability in the deep learning space. For example, we explored masking features, but also worked with methods like CAMs.

Closing Thoughts: Thanks to all the TAs and Professor Ritchie!