PeerJ Preprints NOT PEER-REVIEWED

Hear and See: End-to-end sound classification and visualization of classified sounds

Thomas Miano Corresp. 1

1 Center for Data Science, RTI International, Durham, North Carolina, United States of America

Corresponding Author: Thomas Miano
Email address: tnmiano@gmail.com

Machine learning is a field of study that uses computational and statistical techniques to enable computers to learn. When machine learning is applied, it functions as an instrument that can solve problems or expand knowledge about the surrounding world. Increasingly, machine learning is also an instrument for artistic expression in digital and non-digital media. While painted art has existed for thousands of years, the oldest digital art is less than a century old. Digital media as an art form is a relatively nascent, and the practice of machine learning in digital art is even more recent. Across all artistic media, a piece is powerful when it can captivate its consumer. Such captivation can be elicited through through a wide variety of methods including but not limited to distinct technique, emotionally evocative communication, and aesthetically pleasing combinations of textures. This work aims to explore how machine learning can be used simultaneously as a scientific instrument for understanding the world and as an artistic instrument for inspiring awe. Specifically, our goal is to build an end-to-end system that uses modern machine learning techniques to accurately recognize sounds in the natural environment and to communicate via visualization those sounds that it has recognized. We validate existing research by finding that convolutional neural networks, when paired with transfer learning using out-of-domain data, can be successful in mapping an image classification task to a sound classification task. Our work offers a novel application where the model used for performant sound classification is also used for visualization in an end-to-end, sound-to-image system.

PeerJ Preprints NOT PEER-REVIEWED

Hear and See: End-to-end sound classification and visualization of classified sounds

Thomas N. Miano[1]

[1]**RTI International**

Corresponding author:
Thomas N. Miano[1]

Email address: tnmiano@gmail.com

ABSTRACT

Machine learning is a field of study that uses computational and statistical techniques to enable computers to learn. When machine learning is applied, it functions as an instrument that can solve problems or expand knowledge about the surrounding world. Increasingly, machine learning is also an instrument for artistic expression in digital and non-digital media. While painted art has existed for thousands of years, the oldest digital art is less than a century old. Digital media as an art form is a relatively nascent, and the practice of machine learning in digital art is even more recent. Across all artistic media, a piece is powerful when it can captivate its consumer. Such captivation can be elicited through through a wide variety of methods including but not limited to distinct technique, emotionally evocative communication, and aesthetically pleasing combinations of textures. This work aims to explore how machine learning can be used simultaneously as a scientific instrument for understanding the world and as an artistic instrument for inspiring awe. Specifically, our goal is to build an end-to-end system that uses modern machine learning techniques to accurately recognize sounds in the natural environment and to communicate via visualization those sounds that it has recognized. We validate existing research by finding that convolutional neural networks, when paired with transfer learning using out-of-domain data, can be successful in mapping an image classification task to a sound classification task. Our work offers a novel application where the model used for performant sound classification is also used for visualization in an end-to-end, sound-to-image system.

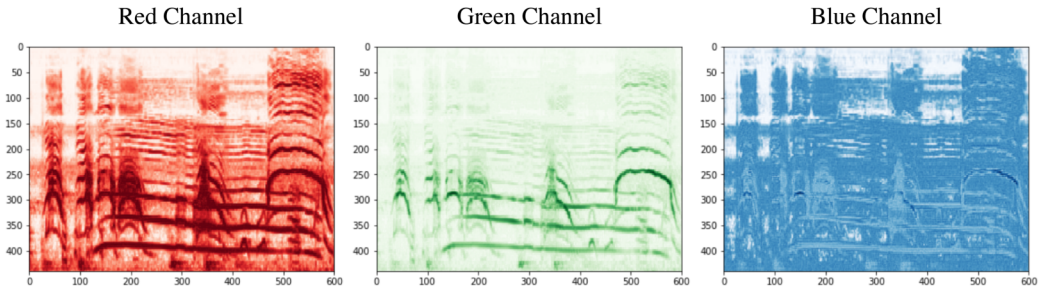

Figure 1. Channels making up a spectrogram generated from a sound in the Children Playing class.

INTRODUCTION

Image classification

Convolutional Neural Networks (CNNs) are effective and widely used for detection and classification problems with image data. When digital images are sampled, light intensity is captured at some range along the electromagnetic spectrum, with multiple ranges (i.e., ”channels”) possibly being captured

and intensity values (i.e., pixels) being stored in a two-dimensional matrix for each channel. One way CNNs interact with images is by moving 2-dimensional matrices over them. These moving matrices function as filters, and as they move over groups of pixels in an image, the values inside of these filters are multiplied by the image pixels. In other words, they are functions that allow certain pieces of information to flow from input to output. This process - combined with loss estimation, back-propagation, and other mathematical processing - is a core part of how CNNs are able to learn latent patterns in data (i.e., features) for classifying different inputs.

While CNNs emerged out of the computer vision field, CNNs can also be applied to digital audio. Digital audio is a representation of waveforms in discrete samples where each sample value has some amplitude, and the frequency of the sound is defined by the number of samples that fall within a cycle. Digital audio is stored in a one-dimensional array in the case of mono recordings and in a two-dimensional array in the case of stereo recordings. CNNs have been used on audio far less frequently than they have been used on images, but one way to facilitate the use of CNNs with sound is to convert sound samples into spectrograms - visual representations of the sound sample frequencies. Testing multiple basic neural network architectures and using a very clean dataset of musical instruments, (5) demonstrates that networks trained using sound data represented in the frequency-space perform significantly better than those trained on the same data represented in the pressure-space. (5) also found that a CNN architecture (similar to LeNet) performed the best at 97% accuracy on their test dataset. (13) lays groundwork for use of CNNs in noisier sound data, like natural environments.

Transfer learning

A popular way to use CNNs is to initialize models with weights pre-trained on large datasets requiring vast computing resources and then ”fine-tuning” them by continuing training on a smaller dataset for the last few layers. This process is called ”transfer learning”, and it has been shown to be effective in producing highly performant models that require less new data (16), which reduces the burden on computational and monetary resources. Additionally, transfer learning can be applied across domains and across data modalities. In our case, we can start off using models that have been trained on images of objects and then do relatively minimal training on images of sounds in order to build a performant sound classifier. Several researchers have investigated CNN transfer learning for sound classification, showing overall good results on a range of tasks from environmental sound classification (13) to music classification (2). (2) demonstrates that the strength of CNN-derived features for sound classification comes from the network structure itself and that even non-domain CNN features (e.g., from a vision task) can be useful if there is not a suitable source task for transfer learning. (2) also show that low level features (especially coming from the first and second layers of a CNN) can be important in predicting a class. Additionally, (Gwardys and Grzywczak) uses AlexNet pretrained on ILSVRC-2012 to classify GTZAN, a music genre dataset, at a 78%. These findings are consistent with existing approaches that use manually derived features of low-level information like rhythmic patterns and tempo (8) (15).

Neural activation visualization

In the last few years, researchers and artists have developed methodology for visualizing “activations”, or the features in an image a CNN finds significant. From a research perspective, this is important because it helps us better understand what CNNs look for when they are given a task. From an artistic perspective, this opens up a new medium for creative expression. The visualization of CNN activations was most famously pioneered by (10). Since then, groups of researchers and artists have advanced the understanding and artistic use of CNNs. (11) provided a clear explanation of CNN visualizations for activations at range of viewpoints including individual neurons, channels, and layers. (6) demonstrates and provides instructions for ”neural synthesis”, or visualization of combined activations at different points in a network. (1) provides infrastructure and tools to facilitate network interpretability and visualization .

METHODS

Data

In this work, we took advantage of the research demonstrating the success of transfer learning outside of the task domain. In order to build our sound classifier, we used a pre-trained ResNet-50 model and fine-tuned it on the UrbanSound8K dataset (14). UrbanSound8K contains 8732 labeled sound excerpts (4 seconds or less) of 10 classes. For most classes, the sample size is 1000 observations. The exceptions to

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27280v1 | CC BY 4.0 Open Access | rec: 15 Oct 2018, publ: 15 Oct 2018

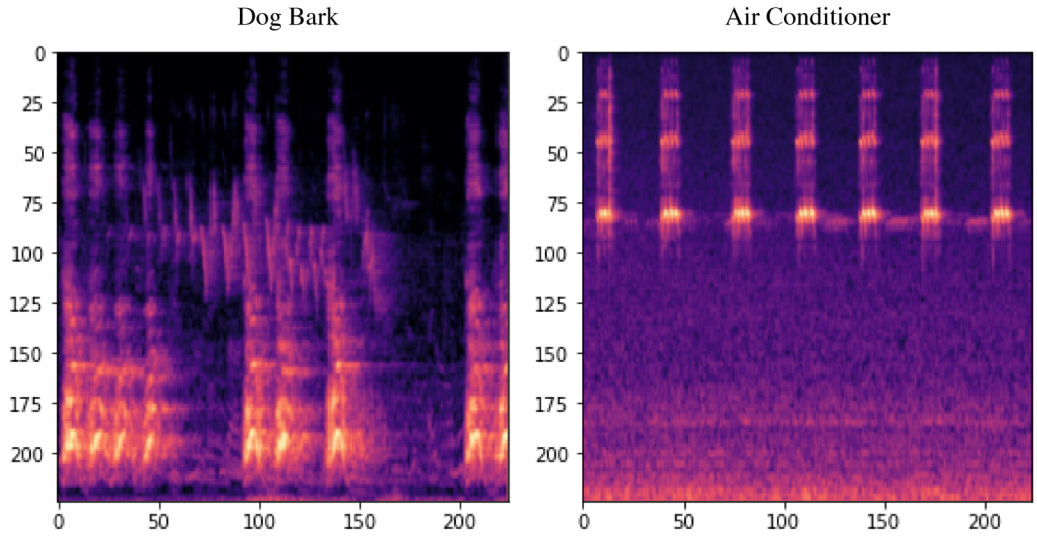

Figure 2. Spectrograms are scaled to 224x224 pixels before being passed into the classification model.

this are Car Horn (n=429), Gun Shot (n=374), and Siren (n=929). The files are pre-sorted into 10 folds, where each fold sub-directory has a random sample of the different sound classes. We then restructured the data so that each file belonging to a particular class occupied its respective class directory. With this baseline dataset, we randomly split the data into a training set, validation set, and final test set – comprising 80%, 10%, and 10% of the data for each class, respectively.

One problem with converting the baseline audio dataset to images is that the length of the samples among the classes are inconsistent, such that some classes have a higher concentration of longer samples whereas other classes have a higher concentration of shorter samples. The duration of the samples has a direct impact on the resolution of the spectrograms that are generated from the samples. For example, when we convert our sounds to spectrograms we will not take the length of our sample into account for determining the dimensions of the output spectrogram, and when our CNN sees a spectrogram it will see it as a 224x224 pixel image (re-scaled from 440x600 pixels), as shown in Figure 2. Because of this, sound samples of different durations will have different image resolutions, potentially biasing the model (4). To address this problem, we created a second dataset by adding sound padding to samples that were less than 4 seconds so that all of the samples ended up having a length of 4 seconds. The frequency level of the padding was set to equal to the frequency value at the beginning and end of each sound sample.

Generating Spectrograms

We then converted all of the sound samples in our baseline and padded datasets to mel-spectrograms. In our implementation, we used LibROSA's mel-spectrogram function, passing the sample rate for each of our files and otherwise using the default parameter values. We experimented with some of the parameters, but we found this could lead to poorly rendered spectrograms when the value of the parameter actually turned out to be inappropriate. In comparison, we found that producing the spectrograms using the default parameters led to outputs without any obvious visual artifacts.

Data Augmentation

As shown in Figure 3, some classes of sounds only appear consistently at certain locations in time. For example, across all samples in the Gun Shot class there is an initial impulse at the beginning of the sample and then a decay. The CNN could also identify this as a feature in distinguishing the different classes. Given that the CNN is being trained on spectrograms, this could create an effect where the model might be unable to identify a gun shot when the impulse comes towards the end of the sample. To address this problem, we used a common image data augmentation technique called "translation" (12), augmenting the padded dataset by shifting (or rolling) the samples forward and backward in time by 25%, 50%, and 75%. This resulted in a dataset with seven times as many observations. Thus, we ended up with a third "rolled" dataset.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27280v1 | CC BY 4.0 Open Access | rec: 15 Oct 2018, publ: 15 Oct 2018

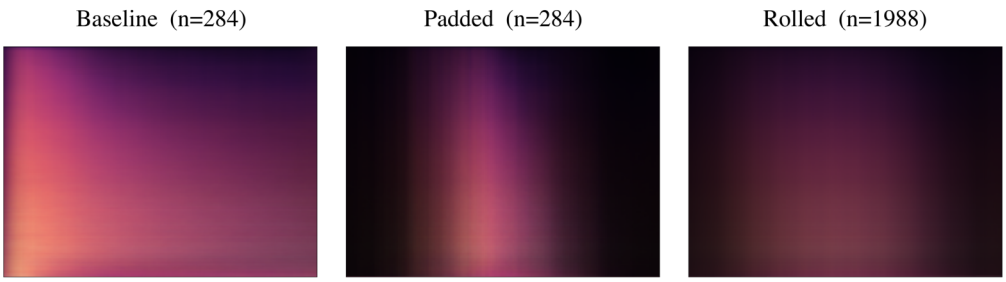

Figure 3. Image averages of the Gun Shot class for each dataset.

Model

To create our environmental sound classifier we fine-tuned the pre-trained PyTorch implementation of ResNet-50, using a single Pascal Titan-X graphics processing unit (GPU). The ResNet-50 model was initialized with weights trained on ImageNet. The only change we made to the architecture of the model was to adjust the final fully-connected layer so that it would produce a probability vector with 10 elements corresponding to the 10 sound classes in the UrbanSound8K dataset. We re-scaled our input images to make them 224x224 pixels, and we used a batch size of 16. We used the Stochastic Gradient Descent (SGD) optimizer, a learning rate of 0.001, momentum of 0.9, weight decay of 0.0005, and 5 training epochs.

RESULTS

Sound Classification Results

Using the parameters identified above, we trained and tested three models on our baseline, padded, and rolled datasets, and we named the models after the datasets that they were trained on. Table 1 shows the testing results for each of our models on all of our testing datasets. Given the sample imbalance of our classes noted earlier, we used F1-Score as our evaluation metric.

Model	Baseline (n=807)	Padded (n=807)	Rolled (n=5649)	Mean
Baseline	95.9%	87.7%	88.0%	90.5%
Padded	83.1%	95.8%	95.9%	91.6%
Rolled	89.4%	97.7%	97.5%	94.9%
Mean	89.5%	93.7%	93.8%	92.3%

Table 1. F1-Scores for models on test datasets.

Overall, we found that each of our models performed well, demonstrating the strength of using a pre-trained model. The discrepancies that we found in the performances were expected. For example, the baseline model performed the worst on the padded dataset at 87.7%. Given that the padded dataset has samples that are at different resolutions for classes that the baseline model was trained on, it was anticipated that it would misclassify some of these. Figure 4 provides a more detailed view of the baseline model's performance on the padded data. One observation is that 40% of the Car Horn test samples were misclassified as Gun Shot. This could be due to the samples being similarly short bursts. Additionally, both of these classes were among the classes with a smaller sample size. Additionally, About 11% of the Jack Hammer samples were misclassified as Drilling. Listening to examples from each, this is not that surprising since both have observations that make continuous humming sounds at a wide range of frequencies that are sometimes met with abrupt starting and stopping.

We expected the rolled model to perform the best on the padded data, and this turned out to be the case with a classification F1-Score of 97.6%. The rolled model was trained on 7x more data, so it had the opportunity to develop better features. It also learned on data with classes in a variety of positions, which would make it more robust and less biased to observations needing to be in a certain position to be recognized. The baseline model performed about 8% worse on the padded and rolled datasets, and it

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27280v1 | CC BY 4.0 Open Access | rec: 15 Oct 2018, publ: 15 Oct 2018

performed about 10% worse on these datasets as compared to the rolled model. This suggests that training on the rolled, padded data is worthwhile. Additionally, while the padded and rolled models performed worse on the baseline data than the baseline model, this is likely due to the image resolution (i.e., sound duration) problems with the baseline data that were mentioned previously.

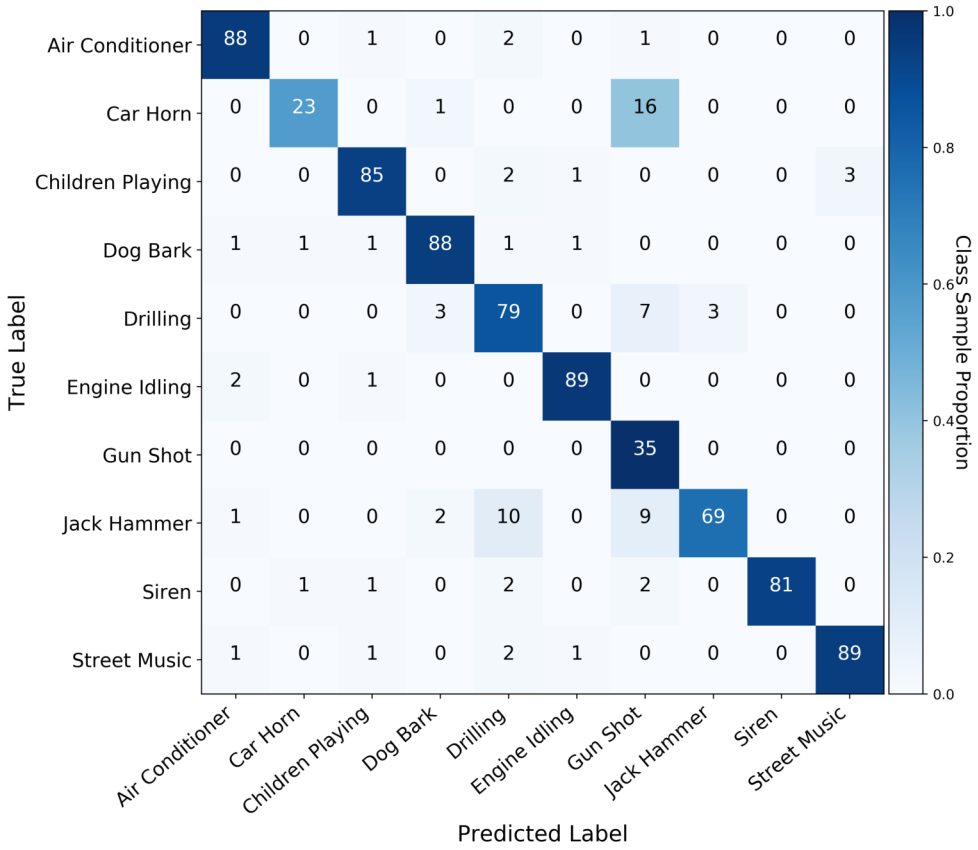

Figure 4. Normalized confusion matrix of the baseline model tested on padded data.

Neural Activation Visualization

We explored two approaches to visualizing classified sound. First, we attempted "real-time" visualization by rendering an image after each optimization step on the GPU. Second, we turned to a "pre-rendered" visualization, where, for every class, we generated 100 images that had been optimized in our model and wrote them to disk. In the first and second case, we used seed images that corresponded to each class (e.g., a picture of a car representing Car Horn). Each image was a CC0 Creative Commons image collected from Pixabay. All of the source images can be found in the Github repository (9) corresponding to this project.

In order to create the neural activation visualization, we began with an existing implementation (7) of Deep Dream in PyTorch, expanding and customizing the code for our purposes. The goal in creating a neural activation visualization was to create self-recursive fractals of the predicted image, such that if the classifier predicted Car Horn, an image of a car would be passed into the model and the model would update the pixels in the image so that more cars would appear. To do this, we take the matrix-wise dot product of the activations for a source image and the activations for the current image at a given time step. We take the argmax of this product, and then the results are back-propagated to the current image. Next, we use a zooming effect where we crop a parameterized number of border pixels around the current image and then scale the image back up to its original size using linear interpolation. This resultant image is then fed back into the network and this process is repeated. Figure 5 shows what this looks like. As the content of the source image begins to disappear into the optimization noise, components of the source image begin to reappear. In the fifth frame of our Car Horn example, car wheels begin to emerge. In the

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27280v1 | CC BY 4.0 Open Access | rec: 15 Oct 2018, publ: 15 Oct 2018

Figure 5. Visualizations of Car Horn (top) and Street Music (bottom) classes.

fourth frame of our Street Music example, clusters of the violin peg box emerge, and in the fifth frame a derivative of the source violinist's face and bowing hand reemerge.

Another goal was to use the probabilities in the sound prediction vector as well as other sonic properties of the input sample (e.g., amplitude) as arguments for the neural visualization hyper-parameters, like optimization iterations and learning rate. Optimization iterations determines the number of times an image will be fed back into the network, and learning rate affects how much the image changes for each iteration. Our preliminary experiments demonstrated that this was possible, but our implementation lacked the efficiency required for a high-fidelity experience. All of the GPU computation was being performed on a remote server, and the visualizations were being rendered locally. This meant that there was some amount of delay between classifying the sound locally, sending information to the server, and receiving data back from the server. Additionally, while Python performs well as a general purpose language, its graphics rendering capabilities are limited as compared to other languages and tools.

Real-time visualization

In experiments, we found that it took about 4.5 seconds to capture a live 4-second windowed time segment, convert it to a spectrogram, and generate a class probability vector. At the fastest, it took about .2 seconds to generate an optimized image on a 200x200 pixel image and about .8 seconds to generate an optimized image on a 400x400 pixel image. The render time varied greatly depending on the number of optimization iterations, the number of octaves (sub-iterations), the number of network layers the image passes through, and the number of pixels in the image resolution. In all of these cases, fewer meant a faster render time. Table 2 shows the render times for several scenarios.

Pre-rendered visualization

In order to improve the frame rate fidelity and viewer experience, we pre-rendered neural visualization image sequences for each sound class. Figure 6 shows a snapshot of what a sound looks like when it is classified and visualized using this method. In this case, we attempted to approximate the sound of drilling by vocalizing a humming sound. We created our visualizations by running our sound classifier in one Jupyter Notebook and our neural visualization in another Jupyter Notebook. This approach worked well, because it created two separate kernels, which allowed the notebooks to run in tandem without having to write a parallel process.

DISCUSSION

Before training any models, we listened to samples from each class. We noticed several audio-content characteristics that we thought could pose problems for classification. First, most of the UrbanSound8K samples were record "in the field", which leads to frequent background noise. We were initially concerned that the noise would drown out the signal in some cases and prevent the classifier from being able to reliably identify it. Related to this, we were concerned that the Air Conditioning class, which can sound like background noise, would be difficult to classify. In the end, the Air Conditioner class was typically classified correctly and few of the other classes were misclassified as Air Conditioner.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27280v1 | CC BY 4.0 Open Access | rec: 15 Oct 2018, publ: 15 Oct 2018

Seconds	Resolution	Iterations	Octaves	Layers
4.7	200x200	1	1	1
4.8	200x200	1	1	2
5.0	200x200	1	1	3
4.9	200x200	2	1	1
5.0	200x200	3	1	1
5.9	200x200	2	2	3
6.5	200x200	3	2	3
5.3	400x400	1	1	1
5.7	400x400	1	1	2
6.0	400x400	1	1	3
6.3	400x400	2	1	1
6.8	400x400	2	1	2
7.6	400x400	2	1	3
6.0	400x400	1	2	1
6.3	400x400	1	3	1

Table 2. Impact of neural visualization parameters on run-time speed.

We envision the best application of this work as a toy for a child, where instead of pushing a button of an animal and hearing the sound it makes, the child could make the sound of that animal and see an image of it shown to them. The visualization system in its current form is insufficient for this. First, the activation noise in the visualized images can be overwhelming, and some of the artifacts that appear – like faces made up of faces – can be disconcerting. Second, while the pre-rendered visualizations solve the fidelity problem, they inhibit control over the visualization parameters. After a while, once you have seen a visualization for each class, the system becomes repetitive. Improving the fidelity for the real-time visualization would prove much more captivating for someone using the system.

In the future, we would like to reduce the amount of time needed to capture a sound sample for classification. To do this would require creating a training dataset that has shorter time segments for our classes of interest. This dataset could be created from the UrbanSound8K dataset, or an entirely new dataset could be created using better recording conditions. Additionally, we would also like to replace our neural activation visualizer with a generative adversarial network. This would be one approach for eliminating the activation noise in our current visualizations and for creating visualizations that appear clearer and more natural.

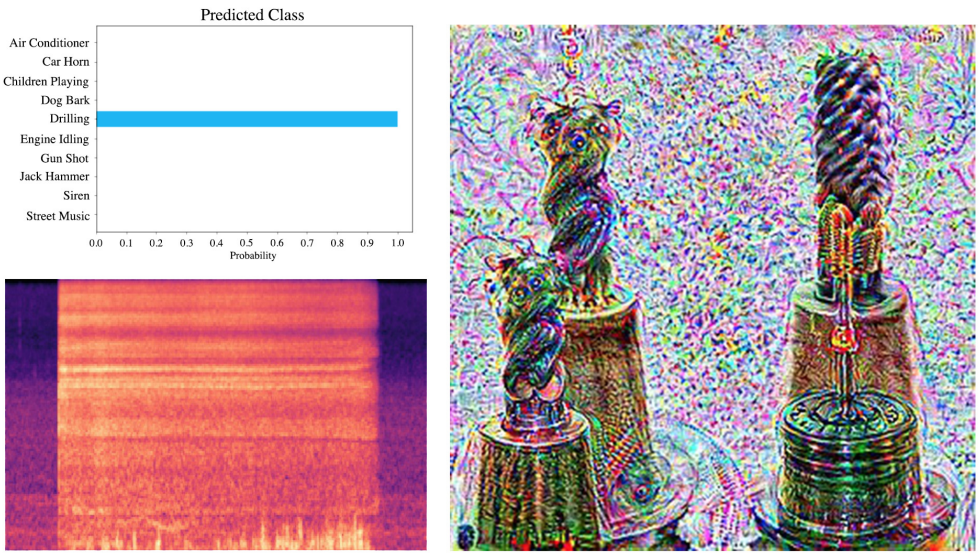

Figure 6. Classification and pre-rendered visualization of vocalized Drilling.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27280v1 | CC BY 4.0 Open Access | rec: 15 Oct 2018, publ: 15 Oct 2018

CONCLUSION

In this work presented an approach for end-to-end sound classification and visualization using a convolutional neural network pre-trained on a dataset outside of the task domain. We verified previous results that CNNs can be used effectively on sound classification when the sounds are converted to spectrograms, and we demonstrated this using ResNet-50 and UrbanSound8K. We illustrated methodology for approaching sound classification from a computer vision perspective, and we provided an open-source repository with clear instructions for reproducing our work, which will enable others to verify our results and to explore their own interests thus further expanding the research and artistic space. In the end, we think machine learning is a tool that will open up new and unexplored methods for creative interaction with the surrounding world.

ACKNOWLEDGMENTS

Thank you to RTI International for providing the GPU computing resources used to develop the models, to Dr. Georgiy Bobashev for providing the funding to write this paper, and to Rob Chew for reviewing the manuscript. Also, thank you to Dr. Vicente Ordonez for providing guidance on the methodology.

REFERENCES

[1] Authors, T. L. (2018). Lucid.

[2] Choi, K., Fazekas, G., Sandler, M., and Cho, K. (2017). Transfer learning for music classification and regression tasks. *arXiv preprint arXiv:1703.09179*.

[Gwardys and Grzywczak] Gwardys, G. and Grzywczak, D. Deep image features in music information retrieval. *International Journal of Electronics and Telecommunications*, 60(4):321–326.

[4] He, K., Zhang, X., Ren, S., and Sun, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European conference on computer vision*, pages 346–361. Springer.

[5] Juliani, A. (2016). Recognizing sounds (a deep learning case study).

[6] Kogan, G. (2017). Neural synthesis.

[7] Lin, X. (2017). Deepdream pytorch.

[8] Marchand, U. and Peeters, G. (2016). The extended ballroom dataset.

[9] Miano, T. (2018). Hear and see (github repository).

[10] Mordvintsev, A., Olah, C., and Tyka, M. (2015). Inceptionism: Going deeper into neural networks. *Google AI Blog*. https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html.

[11] Olah, C., Mordvintsev, A., and Schubert, L. (2017). Feature visualization. *Distill*. https://distill.pub/2017/feature-visualization.

[12] Perez, L. and Wang, J. (2017). The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*.

[13] Piczak, K. J. (2015). Environmental sound classification with convolutional neural networks. In *Machine Learning for Signal Processing (MLSP), 2015 IEEE 25th International Workshop on*, pages 1–6. IEEE.

[14] Salamon, J., Jacoby, C., and Bello, J. P. (2014). A dataset and taxonomy for urban sound research. In *22nd ACM International Conference on Multimedia (ACM-MM'14)*, pages 1041–1044, Orlando, FL, USA.

[15] STURM, B. et al. (2016). Revisiting priorities: improving mir evaluation practices.

[16] Weiss, K., Khoshgoftaar, T. M., and Wang, D. (2016). A survey of transfer learning. *Journal of Big Data*, 3(1):9.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27280v1 | CC BY 4.0 Open Access | rec: 15 Oct 2018, publ: 15 Oct 2018