# Drum Classification: CSE-6740 Final Project Report

Karl Hiner

May 24, 2024

## 1 Introduction

The drum classification task can be informally posed as follows: Given a raw audio clip, assume the clip is an audio recording of one a predetermined set (e.g., snare, tom, kick, hi-hat, cymbal) of drum instruments being struck. The task is to correctly classify the true drum instrument that generated the audio clip.

Such a classifier could be used as a component in a system for the automatic transcription of raw recorded audio of drum performances into a symbolic musical notation such as MIDI. Coupled with generative models for the individual drum instruments (such as physical audio models), one could further imagine decoding this estimated lower-dimensional latent representation (of time-stamped drum instrument classifications) back into the raw audio domain as a form of semantically meaningful audio compression over the restricted domain of drum performances.

Specifically, let $\mathbf{x} \in [-1, 1]^n$ denote an audio clip composed of $n$ samples, where each sample is a real number in the range $[-1, 1]$, and let $y \in \{1, \ldots, k\}$ be a predefined drum instrument label. [1]

Let $\mathcal{D}$ be the joint distribution of random variables $Z = (X, Y)$, where $X$ and $Y$ denote the audio clip and the drum label distributions, respectively. The goal is to learn a classifier $h : \mathbf{x} \mapsto \hat{y}$. The quality of $h$ is evaluated by the (generalization) risk,

$$R(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[l(h(\mathbf{x}), y)],$$

where $l$ is a loss function.

As a realizable proxy for the generalization risk, we use the empirical risk to evaluate the classifier. Let $S = \{z_i = (\mathbf{x}_i, y_i)\}, 1 \leq i \leq m$ be a labeled training set of $m$ samples. Then, the the empirical risk of classifier $h$ with loss $l$ over $S$ is

$$\hat{R}(h) = \frac{1}{m}\sum_{i=1}^{m} l(h(\mathbf{x}_i), y_i).$$

The objective is to minimize $\hat{R}(h)$ to find a classifier $h$ that generalizes well to unseen data from $\mathcal{D}$.

## 2 Related work

Shortly after convolutional neural networks (CNNs) were shown to be effective for image classification [1], CNNs were successfully applied to audio classification tasks. Applying CNNs to audio spectrogram features, in particular Mel Spectrograms and Mel-frequency cepstral coefficients (MFCCs), quickly became a dominant approach for audio classification [2], [3]. In parallel to this work, many approaches emerged for applying recurrent models to spectrogram-based features to better capture temporal patterns for audio classification and speech recognition tasks, including deep recurrent neural networks (RNNs) [4], convolutional recurrent neural networks (CRNNs) [5], and and long short-term memory (LSTM) networks [6].

Recently, classification accuracy has been dramatically improved by first pretraining self-supervised learning (SSL) models to reconstruct audio across large datasets, and then fine-tuning the learned models on classification tasks [7], [8]. Chen et. al. [9] improved this SSL-pretraining paradigm to predict learned discrete tokens rather than minimizing reconstruction loss, encouraging SSL models to detect high-level audio semantic features and discard redundant details, showing significant improvements in sample efficiency and generalization. Current state-of-the-art models extend this discrete-token SSL-pretraining and finetuning approach with multimodal data (e.g. visual, audio, text and 3D) and transformer architectures [10].

Google's Magenta team has recently investigated the application of CNNs and LSTM models to large MIDI-annotated audio datasets for drum transcription in their *Groove MIDI* project [11], [12].

[1] Note that in this project, we do not consider multi-label classification of *overlapping* drum instrument combinations, and we also do *not* include a `null` class to represent the class of all audio clips that don't correspond to a known drum instrument. All samples at training and test time are assumed to be generated from one of the $k$ known drum instruments.

# 3 Proposed approach

We use the *Groove MIDI* dataset [12] to train a CNN model on mel spectrogram features to classify drum instruments. We use a simple (by modern standards) preprocessing pipeline and model architecture, to focus on fundamental questions of the impact of feature selection and model architecture on classification performance.

## 3.1 Data and preprocessing

All data for training and evaluation is derived from the *Expanded Groove MIDI Dataset* (E-GMD) [12]. This is a dataset of human drum performances containing 444 hours of MIDI-annotated audio recordings from 43 electronic drum kits, with predifined train/test/validation splits. MIDI performance data is collected from live performances on a Roland TD-17 electronic drum kit, and aligned with corresponding audio recordings.

To prepare the data for training and evaluation, we perform the following pipeline to find 0.5s labeled segments of single-drum audio clips:

- Filter out sessions recorded with unconventional drum kits, to ease the learning task. 6 manually-selected kits out of 43 were excluded from training.
- Find the $k = 5$ drum instrument classes with the highest note-onset occurrences across this reduced dataset.
- Using heuristic methods, search across the reduced MIDI session dataset to find sections of the recordings that are likely to contain audio belonging to only *one* of the most common $k$ drum instrument classes. [2].
- Run a cleanup pass to account for imprecise MIDI alignment by removing rows with clips whose peak audio onset arrives late in the clip.

Figure 1 shows the distribution of unique MIDI note occurrences in the "slimmed" dataset by instrument.

Figure 2 shows the highly unbalanced distribution of class labels in the "chopped" dataset. We do not explore model/training adaptations or augmentation strategies to explicitly account for this class imbalance in this project.

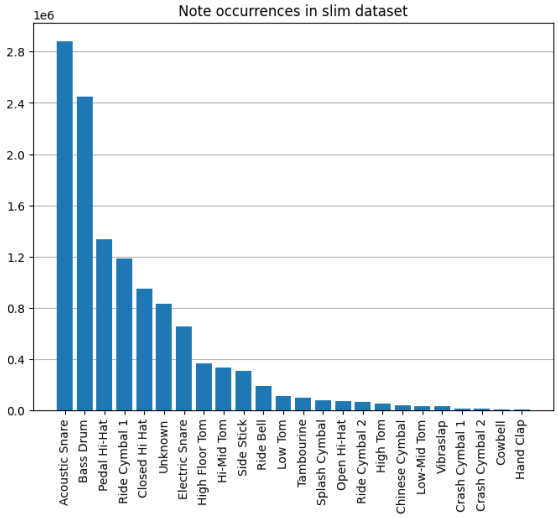


Figure 1: Distribution of unique MIDI note occurrences in the "slimmed" dataset, by instrument name

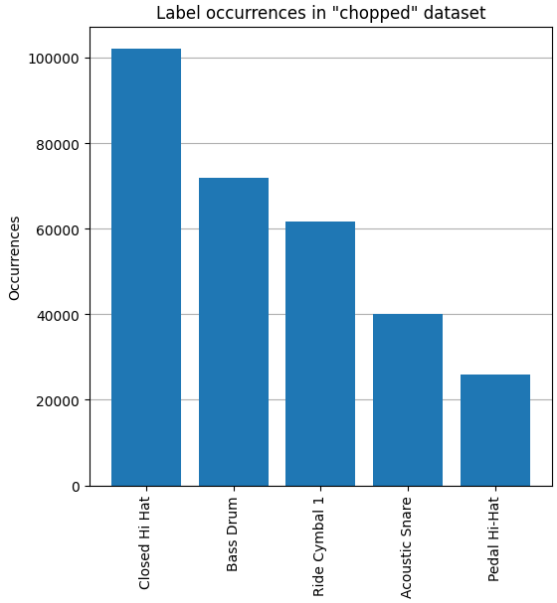


Figure 2: Distribution of drum instrument classes in the "chopped" set.

[2]More details can be found in the Data Preparation section of the accompanying project repo

### 3.2 Audio feature extraction

Preprocessing produces a dataset of 301,640 0.5s 44.1 kHz monophonic audio clips, each labeled with a single drum instrument class.

During training, we transform each raw audio clip into a lower-dimensional representation that aligns more closely with human auditory perception. We first downsample the audio clips [3], and extract Mel spectrogram features from each audio clip. Mel spectrograms transform the audio signal into a lower-dimensional time-frequency representation of signal energy, with frequencies scaled to better match the perceptual frequency response of human hearing [3]. In our final model, we use a 256-bin Mel spectrogram with a 50% overlap between frames, producing a 256x32x1 tensor for each audio clip, for a total dimensionality reduction $\approx 160\%$ [4].

### 3.3 Loss function

We use cross-entropy loss as the loss function $l$. The cross-entropy loss is defined for a single sample as

$$l(h(\mathbf{x}), y) = -\sum_{c=1}^{k} y_c \log(h(\mathbf{x})_c),$$

where $y_c = 1$ iff the sample $y$ belongs to class $c$, and $h(\mathbf{x})_c$ is the predicted probability of the sample $\mathbf{x}$ belonging to class $c$ under classifier $h$.

This loss quantifies the error between the predicted and the true class labels, and can be interpreted as yielding a probability distribution of prediction confidence over all labels.

The empirical risk $\hat{R}(h)$ with the cross-entropy loss over the training set $S$ is then given by:

$$\hat{R}(h) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{c=1}^{k} y_{i,c} \log(h(\mathbf{x}_i)_c)$$

### 3.4 Model architecture

For our model architecture, we use a simple CNN over the Mel spectrograms, with no additional temporal modeling. The model, shown above 3, consists of three convolutional layers with max pooling, followed by a dropout layer, and two fully connected layers.

[3] In our final model, we downsample to 32 kHz, since we found further downsampling hinders performance, likely due to the loss of high-frequency information crucial in disambiguating between drum instruments, especially cymbals and the two high-hat classes

[4] Empirically, we found that loss drops slowly with greater dimensionality reduction, but that the reduction in overfitting is negligent below these values.

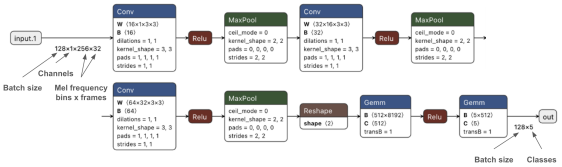


Figure 3: Model architecture.

## 4 Results

We explored the impact of various model architecture and feature extraction choices on the model's training and validation loss curves, with a goal of minimizing error while avoiding overfitting. In particularly, we explored varying the number of convulutional layers and dimensions, resampling rate, STFT window size and hop length (which affects the temporal dimension size), learning rate, and also the audio clip length and the impact of the final data preprocessing step of removing audio clips with "late peaks" (described above). Loss curves for various configurations can be found in the appendix.

We allow training to continue after validation loss begins increasing, and then select the model checkpoint at the epoch with the lowest validation loss.

Our final model correctly classifies 39,356 of 42,122 drum clips over the held out test set, acheiving an accuracy of 93.4%. Below, we show a confusion matrix of the classification results over the test set, and the training and validation loss curves for our final model. The incorrect predictions align well with intuition:

- The *least* common misclassification is between a true label of *pedal hi-hat*, which is short and composed of high-frequency noise, and a mispredicted class of *bass drum*, which tend to be longer, mostly sinusoidal low-frequency decaying tones.

- The *most* common misclassification is between a true label of *closed hi-hat* and a prediction of *pedal hi-hat*. Anecdotally, we have found that it can be difficult or impossible to distinguish between these two classes by ear, with similarity highly dependent on the drum kit type and the velocity of the hits.

In our accompanying repo, we provide a notebook to explore the dataset, including an interactive visualization and preview, as well as a variable-length supercut generator for each class.

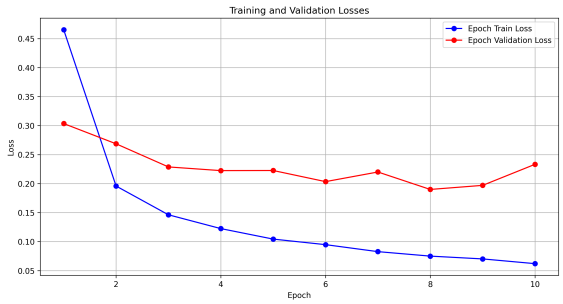


Figure 4: Training and validation loss curves for final model.

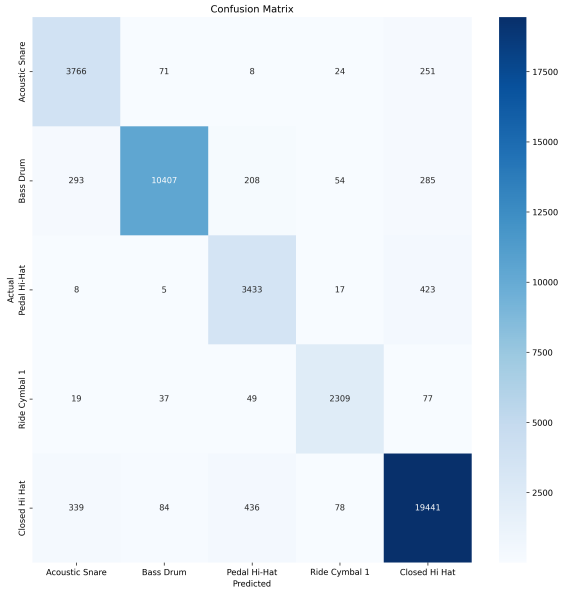


Figure 5: Confusion matrix of classification over test set.

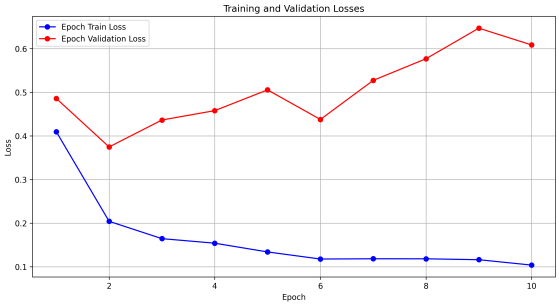


Figure 6: 1 second audio clips, 3 conv layers

# 5 Conclusion

It should be noted that conventional CNN architecture, such as the one employed here, may be ill suited to capture the relationships in audio spectrograms. After all, unlike with images captured with cameras, which have highly similar relationships across the horizontal and vertical image dimensions, the time and frequency dimensions of audio spectrograms have completely different physical interpretations. Despite this inherent limitation, this simple architecture has been shown to be effective for audio classification tasks [2], [3]. The model may accomplished this, in part, by learning highly asymmetric kernels for differential propagation of information across the temporal and frequency dimensions. Future work could investigate the distribution of the values of individul learned kernels to see if this is indeed the case.

Additionally, the model is hindered by poor segmentation in some instances, with accidentally overlapping drum instruments. This could be improved by including multi-label classification. Finally, we acknowledge that a more straightforward data acquisition process could be achieved by simply downloading multiple sets of pre-labelled drum sample packs. However, the MIDI-annotated audio dataset used here is more representative of real-world drum performances, and allows for future incorporation of velocity information into the model.

# 6 Appendix

We provide the training and loss curves for various model configurations below.

# References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances*

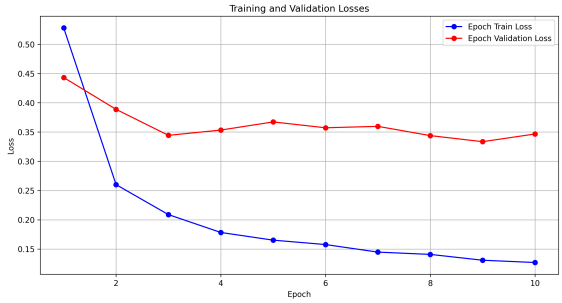


Figure 7: 0.5 secondclips, 2 conv layers

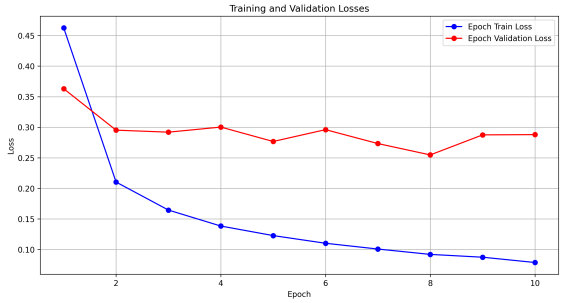


Figure 8: 0.5 secondclips, 3 conv layers

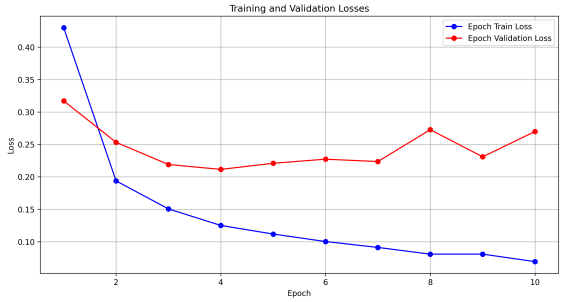


Figure 9: 0.5 secondclips, 3 conv layers, clean peaks

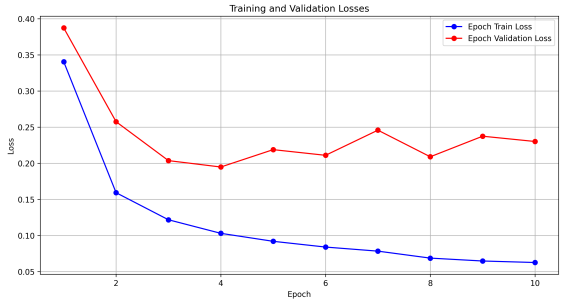


Figure 10: 0.5 second audio, 3 conv layers, clean peaks, 22,500 hz resample

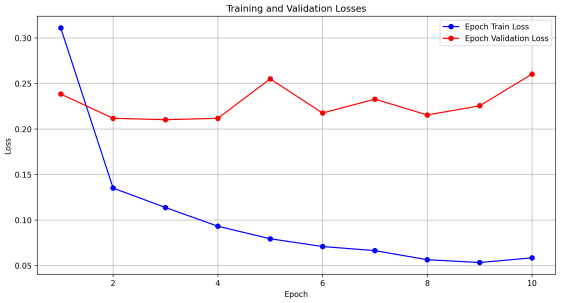


Figure 11: 0.5 second audio, 3 conv layers, clean peaks, 22,500 hz resample, half-hop Mel spectrogram

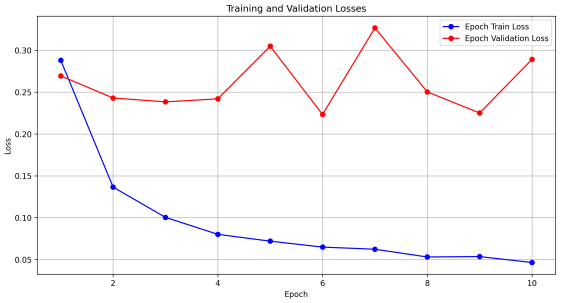


Figure 12: 0.5 second audio, 3 conv layers, clean peaks, 320,00 hz resample


Figure 13: 0.5 second audio, 3 conv layers, clean peaks, 32,000 hz resample, 2048 nfft, halfhop, 256 mel

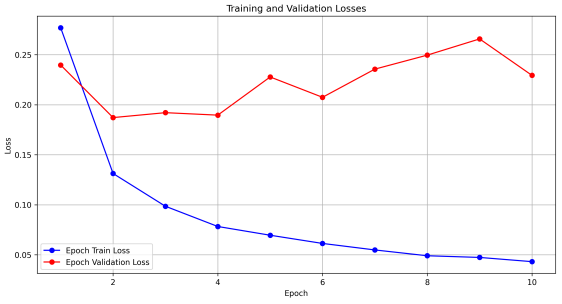


Figure 14: 0.5 second audio, 3 conv layers, more data, clean peaks, 32,000 hz resample, 2048 nfft, halfhop, 256 mel


Figure 15: 0.5 second audio, 3 conv layers, more data, clean peaks, 32,000 hz resample, 2048 nfft, halfhop, 256 mel, reduced lr

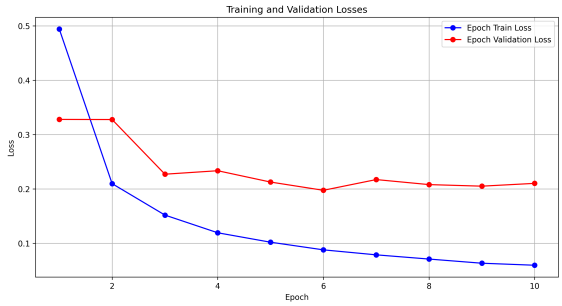


Figure 16: 0.5 second audio, 3 conv layers, more data, clean peaks, no resample, 2048 nfft, halfhop, 256 mel, reduced lr

*in Neural Information Processing Systems*, vol. 25, Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html (visited on 12/07/2023).

[2] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, ISSN: 2378-928X, Sep. 2015, pp. 1–6. DOI: 10.1109/MLSP.2015.7324337. [Online]. Available: https://ieeexplore.ieee.org/document/7324337 (visited on 12/07/2023).

[3] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, *CNN Architectures for Large-Scale Audio Classification*, arXiv:1609.09430 [cs, stat], Jan. 2017. [Online]. Available: http://arxiv.org/abs/1609.09430 (visited on 12/07/2023).

[4] A. Graves, A.-r. Mohamed, and G. Hinton, *Speech Recognition with Deep Recurrent Neural Networks*, arXiv:1303.5778 [cs], Mar. 2013. [Online]. Available: http://arxiv.org/abs/1303.5778 (visited on 12/07/2023).

[5] K. Choi, G. Fazekas, M. Sandler, and K. Cho, *Convolutional Recurrent Neural Networks for Music Classification*, arXiv:1609.04243 [cs], Dec. 2016. [Online]. Available: http://arxiv.org/abs/1609.04243 (visited on 12/07/2023).

[6] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and Classification of Acoustic Scenes and Events," en, *IEEE Transactions on Multimedia*, vol. 17, no. 10, pp. 1733–1746, Oct. 2015, ISSN: 1520-9210, 1941-0077. DOI: 10.1109/TMM.2015.2428998. [Online]. Available: http://ieeexplore.ieee.org/document/7100934/ (visited on 12/07/2023).

[7] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, *Wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations*, arXiv:2006.11477 [cs, eess], Oct. 2020. DOI: 10.48550/arXiv.2006.11477. [Online]. Available: http://arxiv.org/abs/2006.11477 (visited on 12/07/2023).

[8] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, J. Wu, L. Zhou, S. Ren, Y. Qian, Y. Qian, J. Wu,

M. Zeng, X. Yu, and F. Wei, "WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, Oct. 2022, arXiv:2110.13900 [cs, eess], ISSN: 1932-4553, 1941-0484. DOI: 10.1109/JSTSP.2022.3188113. [Online]. Available: http://arxiv.org/abs/2110.13900 (visited on 12/07/2023).

[9] S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen, and F. Wei, *BEATs: Audio Pre-Training with Acoustic Tokenizers*, arXiv:2212.09058 [cs, eess] version: 1, Dec. 2022. [Online]. Available: http://arxiv.org/abs/2212.09058 (visited on 12/07/2023).

[10] S. Srivastava and G. Sharma, *OmniVec: Learning robust representations with cross modal sharing*, arXiv:2311.05709 [cs] version: 1, Nov. 2023. [Online]. Available: http://arxiv.org/abs/2311.05709 (visited on 12/07/2023).

[11] J. Gillick, A. Roberts, J. Engel, D. Eck, and D. Bamman, "Learning to groove with inverse sequence transformations," in *International Conference on Machine Learning (ICML)*, 2019.

[12] L. Callender, C. Hawthorne, and J. Engel, *Improving perceptual quality of drum transcription with the expanded groove midi dataset*, 2020. arXiv: 2004.00188 [cs.SD].