LEVERAGING HIERARCHICAL STRUCTURES FOR FEW-SHOT MUSICAL INSTRUMENT RECOGNITION

Hugo Flores Garcia, Aldo Aguilar, Ethan Manilow, Bryan Pardo
{hugofg@u., aldoa@u., ethanm@u., pardo@}northwestern.edu
Interactive Audio Lab, Northwestern University, Evanston, IL, USA

ABSTRACT

Deep learning work on musical instrument recognition has generally focused on instrument classes for which we have abundant data. In this work, we exploit hierarchical relationships between instruments in a few-shot learning setup to enable classification of a wider set of musical instruments, given a few examples at inference. We apply a hierarchical loss function to the training of prototypical networks, combined with a method to aggregate prototypes hierarchically, mirroring the structure of a predefined musical instrument hierarchy. These extensions require no changes to the network architecture and new levels can be easily added or removed. Compared to a non-hierarchical few-shot baseline, our method leads to a significant increase in classification accuracy and significant decrease in mistake severity on instrument classes unseen in training.

1. INTRODUCTION

Musical instrument recognition is a machine learning task that aims to label audio recordings of musical instruments, typically at a fine temporal granularity (second by second) [1–3]. Musical instrument recognition can be viewed as a subtask of Sound Event Detection (SED), which consists of identifying and locating any type of sound event (*e.g.*, car horn, dog bark) in an audio recording [4–6].

Labelling audio tracks is extremely important for organizing the dozens of tracks in a typical Digital Audio Workstation (DAW) recording session [7,8], but manual labelling is a tedious process. Automated musical instrument recognition could enable automated track labeling. Automated second-by-second labeling could go further, enabling navigation through recording projects by traversing musical instrument *labels*, rather than waveform visualizations. This would be especially helpful for audio engineers with low or no vision, as existing interfaces leave accessibility as an afterthought [9] and navigating by visually examining waveforms is not a viable option for them [10].

A barrier to incorporating instrument recognition into DAWs is that most existing deep learning techniques must be trained on instruments that have abundant labeled training data. The datasets that support these systems only focus on the limited set of instrument classes that have sufficient data [11–17]. However, the vast diversity of musical instrument sounds necessitates supporting a broader set of instrument classes [18]. While expanding current datasets with more diverse coverage can ameliorate this issue, collecting human annotations for a large number of audio files is a tedious, time consuming task [19, 20], and there will always be unanticipated sound categories that an end-user would like to automatically label.

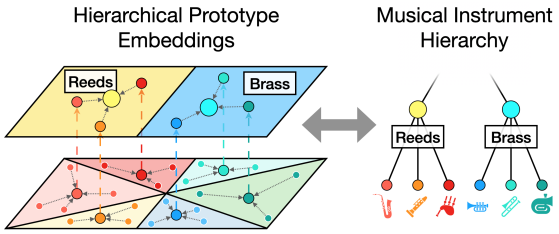

Figure 1. Overview of our method. Prototypes from a set of embedded support examples at a fine-grained level (bottom left) are aggregated to make a set of *metaprototypes* at a coarser-grained level (top left). In this way, we learn a hierarchical set prototypes that corresponds to a musical instrument hierarchy (right).

Therefore, musical instrument recognition systems should be able to dynamically expand their vocabularies after deployment, to conform to end-user needs. This requires an approach that lets a system learn a new sound category given only a few examples that can be provided by an end user, *a la* few-shot learning.

Using a hierarchical system, like the widely-used Hornbostel-Sachs hierarchy [21], to organize and classify musical instruments has broad precedent in many human cultures [22]. We can take advantage of a musical instrument hierarchy, like the widely-used Hornbostel-Sachs hierarchy [21], to improve few-shot learning. A system could learn a feature space meaningful for unseen classes that share hierarchical ancestry with the classes seen during training. For example, the Chinese zhongruan is a plucked string instrument that shares ancestry with other chordophones in the Hornbostel-Sachs hierarchy (like the guitar), which might be more common in datasets of Western instruments. A model could leverage the hierarchical relationship between an instrument it has never been trained on (*e.g.* the zhongruan) and more common instru-

ments seen during training (*e.g.* the guitar) to produce a meaningful representation of the new instrument with only a few support examples.

In this work, we propose a simple extension to prototypical networks [23] that imposes a hierarchical structure on the learned embedding space (Figure 1). We first create prototypes from an initial set of embedded support examples at the most granular level. We then aggregate these initial prototypes into new prototypes corresponding to a coarser hierarchical level, in a manner reminiscent of agglomerative clustering [24]. Repeating this process lets our system represent classes at many granularities of a predefined instrument hierarchy. We also propose a weighted, hierarchical extension of cross-entropy loss to ensure the network learns the hierarchy. Compared to a non-hierarchical few-shot baseline [25], our method shows a significant increase in classification accuracy and significant decrease in mistake severity on unseen instrument classes.

2. RELATED WORK

Musical instrument recognition can be performed in single-source contexts [26–29], where only a single sound source may be active at any given time, as well as in multi-source contexts [13–15, 30, 31], where multiple sound sources may be active at the same time. We consider the single-source case, as the vast majority of audio in a studio music production workflow is single-source.

Hierarchical structures have shown to be effective for many machine learning tasks, such as text classification [32] and image classification [33, 34]. In fact, Bertinetto *et al.* [35] propose a hierarchical image classification approach that uses a similar exponentially weighed hierarchical loss function to the one proposed here, although they do not focus on a few-shot setting, as we do, and they favor learning broader classes, whereas we are also interested in finer classes. Hierarchical structure was explored for musical instrument recognition by using fixed signal processing feature extraction techniques [29, 36, 37]. Here, we use deep learning methods to flexibly learn a feature space that mirrors musical instrument hierarchies.

Recent work has studied how hierarchical structures can be incorporated into neural network models for different tasks. In the automatic speech recognition (ASR) domain, CTC-based hierarchical ASR models [38–40] employ hierarchical multitask learning techniques, particularly by using intermediate representations output by the model to perform intermediate predictions in a coarse-to-fine scheme. Manilow *et al.* [41] have shown that hierarchical priors can have significant benefits for performing source separation of musical mixtures. None of these systems, however, were designed for few-shot learning.

Previous deep learning systems have been proposed for multilevel audio classification [42–44]. However, none of these systems work in a few-shot setting and they require either specialized network architectures or complex data pipelines to learn a hierarchy. Our approach is a simple extension to incorporate hierarchy into an established few-shot learning paradigm.

Recent work in audio tagging and sound event detection tasks has explored few-shot learning in the audio domain [19, 25, 45–47], though none of this work assumed any hierarchical structure.

Here, we propose a method for hierarchical representation learning in a few-shot setting, leveraging the increased flexibility of both hierarchy and few-shot methods for musical instrument recognition.

3. BACKGROUND

3.1 Few-shot Learning

In a few-shot classification setting, we consider a target class $k \in \mathcal{K}$ for a set of target classes, $\mathcal{K}$, of size $|\mathcal{K}|$. Let x_s be a single support example drawn from a set of examples $\mathcal{S}$, called the support set. Assume N labeled support examples (*i.e.*, shots) per class k, totalling $N \times |\mathcal{K}|$ labeled examples. We define $\mathcal{S}_k$ as the subset of $\mathcal{S}$ containing the examples of class k.

We are provided an unlabeled query set $\mathcal{Q}$ of M unlabeled examples. The goal of the task is to label each query example $x_q \in \mathcal{Q}$ with a target class $k \in \mathcal{K}$. A neural network model f_θ projects both the support and query sets into a discriminative embedding space. The query is assigned to the class of the support set it is closest to, according to distance metric d.

3.2 Prototypical Networks

Prototypical networks [23] compute an embedding vector for each instance in $\mathcal{S}_k$. The prototype, c_k, for class k is the mean vector of all the support embeddings belonging to class k:

$$c_k = \frac{1}{|S_k|} \sum_{x_s \in \mathcal{S}_k} f_\theta(x_s). \tag{1}$$

Using a distance function d, we can produce a probability distribution over the set of classes $\mathcal{K}$ for a given query x_q by applying a softmax over the negated distances from the query to each class prototype:

$$p(\hat{y}_q = k | x_q) = \frac{\exp\left(-d(f_\theta(x_q), c_k)\right)}{\sum_{c'_k} \exp\left(-d(f_\theta(x_q), c'_k)\right)}. \tag{2}$$

We use the Euclidean distance as d in this work.

4. METHOD

Musicologists have long categorized musical instruments into hierarchical taxonomies, such as the Hornbostel-Sachs system [21], which classifies musical instruments into a hierarchy corresponding to their sound producing mechanisms. We can improve upon existing few-shot models by leveraging the hierarchical structure intrinsic to musical instrument taxonomies. To do this, we extend prototypical networks by training on a multitask scenario composed of multiple classification tasks, one for each level of a class tree, where the prototype for a parent node in the class tree is defined as the mean of the prototypes for each of the parent node's children.

We impose hierarchical structure on our few-shot task by constructing a tree, T, with height H, starting from a set of leaf nodes. We define the leaf nodes as the same set of classes, $\mathcal{K}$, that we defined for our standard few-shot setup in Sec. 3.1. We then define the parents of the leaf nodes by aggregating classes, $k \in \mathcal{K}$. For musical instrument recognition, we aggregate classes according to a predefined instrument hierarchy (*e.g.*, Hornbostel-Sachs). We iteratively aggregate child classes up to the max height of the tree H. We index the tree as $T_{i,h}$, where $i \in \mathcal{K}_i$ indexes over the set of sibling classes at level h, for $h = 0, \dots H$, with level 0 containing the most specific classes and level H containing the broadest. In our notation $H = 0$ describes a tree with no hierarchy and is equivalent the non-hierarchical prototypical network defined in Sec. 3.2. $H = 1$ has two levels, and so on.

4.1 Hierarchical Prototypical Networks

We define our proposed hierarchical prototypical network by extending typical prototypical networks [23] to a hierarchical multitask learning scenario, where we wish to label each query example, $x_q \in \mathcal{Q}$, at multiple levels of our class tree, T. Here, labeling at each level is a separate task.

Like a normal prototypical network, we use a network f_θ to produce embeddings for every example in the support set. The mean of these embedded support examples creates an initial set of prototypes (Eq. 1). We deviate from the typical setup by considering this initial set of prototypes as the lowest level of our tree, T, and aggregating these initial prototypes *again* to make another set of prototypes representing the next level. The prototypes at this higher level are, thus, prototypes of prototypes, or *metaprototypes*, and define a hierarchy according to the structure of our tree, T. We continue to iteratively aggregate prototypes in this fashion for all levels of our tree. The prototype for each parent class at level $h+1$ is notated $c_{T_{i,h+1}}$ and is the mean of the members of its support set $\mathcal{S}_{T_{i,h}}$. For levels $h > 0$, each example $\hat{x}_s$, is itself a prototype:

$$c_{T_{i,h+1}} = \frac{1}{|\mathcal{S}_{T_{i,h}}|} \sum_{\hat{x}_s \in \mathcal{S}_{T_{i,h}}} f_\theta(\hat{x}_s), \tag{3}$$

This process is shown in Figure 1.

Given a query example x_q, we use the network to create an embedding $f_\theta(x_q)$ and measure its distance to each class prototype or metaprototype $c_{T_{i,h}}$ at a given level h. Given these distances, we output H probability distributions, one for each level in our class tree:

$$p(T_{i,h}|x_q) = \frac{\exp\left(-d(f_\theta(x_q), c_{T_{i,h}})\right)}{\sum_{c'_{T_{i,h}}} \exp\left(-d(f_\theta(x_q), c'_{T_{i,h}})\right)}. \tag{4}$$

We note that Eqs. 1 and 2 are special cases of the proposed Eqs. 3 and 4, evaluated at $h = 0$. Our generalization allows multi-task few-shot classification at multiple levels of a hierarchical class tree.

Our proposed method does not require any specific network architecture. Instead, it provides a hierarchical label structure for support examples x_s to be aggregated together, forming fine-to-coarse representations (*i.e.*, $c_{T_{i,h}}$) that we can leverage and optimize with. This exposes the potential for a model to be trained with multiple concurrent hierarchies, a direction for future work.

4.2 Multi-Task Hierarchical Loss

We now set up a learning objective, where we minimize the cross-entropy loss between the predicted distribution and the ground truth class for each level in the class tree. The intuition behind our approach is that we can use a hierarchically structured objective to encourage our model to produce an embedding space with discriminative properties at both coarse and fine granularities, allowing some of these coarse features to generalize beyond the training set of fine grained leaf classes to their unseen siblings in the class tree. We use an exponentially decaying sum of loss terms for each level in the hierarchy [35]:

$$\mathcal{L}_{hierarchical} = \sum_{h=0}^{H} e^{-\alpha \cdot h} \mathcal{L}_{CE}^{(h)}, \tag{5}$$

where $\mathcal{L}_{CE}^{(h)}$ denotes the cross-entropy loss for the classification task at height h, and α is a hyperparameter that determines the decay of each loss term w.r.t height. Setting $\alpha > 0$ places more more weight on finer-grained tasks, $\alpha < 0$ places more weight on coarser-grained tasks, and $\alpha = 0$ weighs all tasks equally. We note that $H = 0$ reduces to the non-hierarchical (baseline) definition of the problem, where we only optimize for the fine-grained task.

5. EXPERIMENTAL DESIGN

We evaluated our proposed hierarchical prototypical approach using a non-hierarchical prototypical method [25] as a baseline. We evaluated all models on a few-shot musical instrument recognition task, measuring standard classification metrics (F1) as well as mistake severity. We conducted ablations for class tree height, choice of class hierarchy, and proposed loss function.

5.1 Datasets

For all experiments, we trained and evaluated using isolated tracks from the MedleyDB [48] and MedleyDB 2.0 [49] datasets. MedleyDB contains multi-track recordings of musical instruments and vocals. We excluded recordings that do not have fine-grained instrument labels (*e.g.*, "brass" was excluded because the audio could be of trumpets, trombones, etc.). Additionally, we considered sections of a single instrument to be the same class as the instrument itself (*e.g.* "violin section" and "violin" both belong to the class "violin"). Altogether, the dataset consists of 63 different instruments, with 790 tracks in total.

For training and evaluation, we removed the silent regions of each audio track. We then split the remainder of the track into 1 second segments with a hop size of 0.5 seconds, where each 1 second segment is an input example to the model. All audio was downsampled to 16kHz. For

each example, we compute a 128-bin log-Mel spectrogram with a 32ms window and an 8ms hop. After preprocessing, our training and evaluation datasets contained 539k and 56k 1-second examples, respectively. We performed silence removal using `pysox` [50].

5.2 Network Architecture

The backbone network architecture used in all experiments was based on the prototypical network described in Wang *et al.* [47]. It uses a log-Mel spectrogram as input, and consists of four CNN blocks, where each convolutional filter has a kernel size of 3×3, followed by a batch normalization layer, a ReLU activation, and a 2×2 maxpooling layer. After the last convolutional block, we applied maxpooling over the time dimension, to obtain a 1024-dimensional embedding. Finally, we added a linear projection layer that reduces the 1024-dimensional embedding to 128 dimensions.

5.3 Hornbostel-Sachs Class Tree

We used a musical instrument hierarchy inspired by the Hornbostel-Sachs [21] taxonomy, [1] (maximum height of 4) which is organized by the sound production mechanisms of each instrument. Since similar sound production mechanisms can lead to similar sounds, we believe this is a natural organization that our model can leverage to learn discriminative features at different levels of a class hierarchy.

5.4 Episodic Training and Evaluation

We have a musical instrument hierarchy tree, where individual instrument classes are leaf nodes (e.g. violin, guitar). Nodes at higher levels ($h > 0$) are instrument families, (e.g. bowed strings, plucked strings). Our goal is to observe classification performance on previously-unseen leaf classes (e.g. zhongruan, erhu). Therefore, we created a data split of 70% train, 30% evaluation, with no overlap between train and evaluation classes at the leaf instrument level ($h = 0$). We further added the constraint that the classes in both testing and evaluation sets be distributed evenly among the instrument families ($h > 0$). This avoids a problem where, for example, the train set consists only of percussion and the evaluation set consists only of chordophones. All experiments shared a train/evaluation split.

For each experiment, we trained every model in a few-shot learning scenario using episodic training. Each model was presented with a unique $|\mathcal{K}|$–way, N–shot learning task (an episode) with M queries per leaf class at each training step. We constructed an episode by sampling a set of $|\mathcal{K}|$ instrument classes from the training data. For each of these $|\mathcal{K}|$ classes, we sampled $N + M$ audio examples. Here, for each class k, $N = |S_k|$ is the number of "shots" in the support set and M is the size of the query set.

We trained all models using the same random initialization for a maximum of 60,000 steps with early stopping after the evaluation loss stopped improving for 4500 steps, using the Adam optimizer and a learning rate of 0.03. During training, we set $|\mathcal{K}| = 12$, $N = 4$, and $M = 12$. We evaluated each trained model on episodes constructed from the test data. For each evaluation, we made 100 episodes, with $|\mathcal{K}| = 12$, $M = 120$. All hyperparameters were fixed except those we ablated, as described below.

5.5 Evaluation Metrics

We used the F1-score as our primary classification metric, reporting the distribution of F1 scores computed for each episode, evaluated for predictions made at the finest level of the hierarchy.

Similar to Bertinetto *et al.* [35], we used the hierarchical distance of a mistake as a metric indicative of a model's mistake severity. Given a class tree, the hierarchical distance of mistake is defined as the height of the lowest common ancestor (LCA) between the prediction node and ground truth node when the input is misclassified (that is, when the model makes a mistake).We report the average hierarchical distance of a mistake over all evaluation episodes.

For all hierarchical models, we measured mistake severity with respect to its own hierarchy. For the non-hierarchical model, we evaluated with respect to our proposed 4-level version of the Hornbostel-Sachs hierarchy, as we believe that its organization is meaningful.

6. EXPERIMENTS

We now describe specific experiments to measure the effects of different design choices. We trained and evaluated all models using the procedure described in Section 5. Our experiment code is available online [2].

6.1 Tree Height

To observe the effect of tree height on classification, we constructed shorter trees from the Hornbostel-Sachs class tree by removing every leaf node's parent until the desired max height of the tree is met. We trained and evaluated five models using our proposed class tree, shortened to different heights $H \in \{0, 1, 2, 3, 4\}$, where $H = 0$ is the baseline, non-hierarchical case inspired by Wang *et al.* [25]. Each model was trained with $\alpha = 1$ and evaluated with $N = 8$ support examples per class, at inference.

Results are shown in Figure 2. All variations of the proposed model achieved a better classification performance than the baseline. The best F1 score was seen at $H = 1$, with a mean value of .8111 over all evaluation episodes. Compared to the baseline mean score of .7792, this is a 4% improvement. A Wilcoxon signed-rank test showed that all of our proposed models achieve a statistically significant improvement when compared to the baseline, with $p < 10^{-7}$ for all hierarchies. These results show that incorporating our method into a prototypical network can lead to statistically significant improvements in classification performance under few-shot learning conditions.

[1] See: https://en.wikipedia.org/wiki/Hornbostel-Sachs

[2] https://github.com/hugofloresgarcia/music-trees

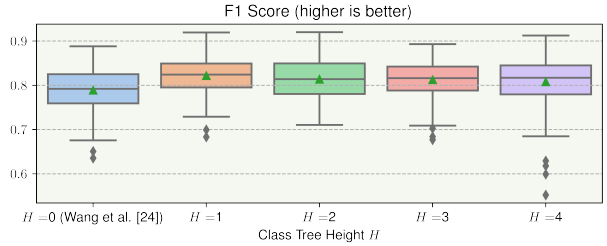

Figure 2. F1 scores for models trained with class trees of varying height H, evaluated over 100 episodes. Means are shown as green triangles. Note that $H = 0$ is our baseline model (Wang *et al*. [25]), as it is trained without a class tree.

Surprisingly, a shallow tree with only the coarsest categories and the leaf nodes ($H = 1$) achieved the highest increase in performance. We believe this is due to the small number of classes encountered in a training episode (in our case, 12). At a given level of the tree, at least 2 of the classes in the support set need to have a parent node in common for our method to be able to compute a meaningful metaprototype that can be leveraged by our loss. As a class tree gets deeper, the number of nodes at a given level can grow exponentially, meaning that our support set of 12 classes has a lower chance of finding meaningful groupings at deeper levels. This indicates that loss terms for levels closer to the leaf nodes are more likely be identical to the non-hierarchical loss. Though the loss term for the coarsest level is still present in these deeper trees, it has a smaller impact on the gradient of the primary loss function, as loss terms are weighted to decay exponentially as the height increases. We believe training with a higher $|\mathcal{K}|$ can help leverage deeper hierarchies better. However, we leave this for future work.

6.2 Number of Support Examples

We evaluated our best proposed model ($H = 1$, $\alpha = 1$) as well as our baseline model by varying the number of support examples N provided to the model, where $N \in \{1, 4, 8, 16\}$. Results are shown in Figure 3 (left). We notice that increases in performance are greater when more support examples are provided, with the smallest increase (+2.17% in the mean relative to baseline) occurring when $N = 1$. Our model achieved a statistically significant improvement on all test cases ($p < 10^{-4}$ for all N).

As shown in Figure 3 (right), our model achieved a lower hierarchical distance of a mistake, on average. A Wilcoxon signed-rank test indicates that all improvements are statistically significant ($p < .0005$). This means that, when making incorrect predictions, our method was more likely to make predictions that are closer to the ground truth in terms of the class hierarchy (*i.e.*, lower mistake severity). We believe it is fair to assume that mistake severity from a sound production perspective (as in our class hierarchy) is related to mistake severity in predictions made by humans. That is, a human is more likely to confuse a viola for a violin than to confuse a viola for a drum.

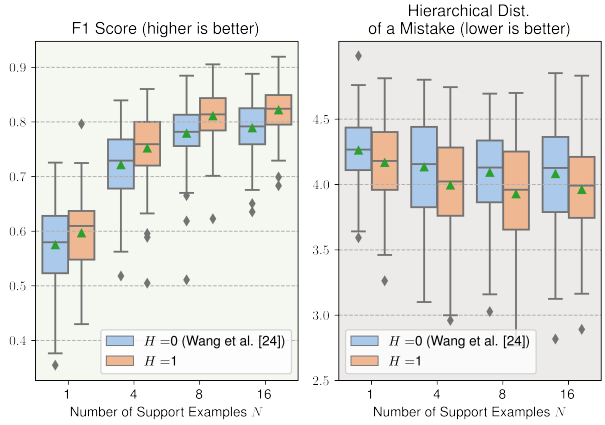

Figure 3. Model comparison between the baseline model and our best proposed model ($H = 1$), evaluated under conditions with a different number of shots (support examples) provided during inference.

6.3 Arbitrary Class Trees

To understand how the choice of hierarchy affects the results of our model, we evaluated the same prototypical network architecture trained using the Hornbostel-Sachs hierarchy and also 10 randomly generated class trees. We generated each tree by performing random pairwise swaps between leaf nodes in our original class tree, doing so 1000 times for each node. For this experiment, all trees were trained with ($H = 3$, $\alpha = 1$), and evaluated with $N = 16$.

Results for our evaluation of random class hierarchies are shown in Figure 5. Our best performing random hierarchy in terms of classification performance ("random-best") achieves an F1 score comparable to our proposed hierarchy ($p > 0.05$) though with a larger spread. Additionally, "random-best" obtains much worse mistake severity relative to the hierarchy it was trained on. This indicates that the model was not able to generalize the hierarchical structure it was trained on to out-of-distribution classes. On the other hand, our worst performing random hierarchy, "random-worst", caused a statistically significant deterioration in both classification performance and mistake severity compared to the baseline ($p < 0.005$). Even though the random-best model fairs comparably to Hornbostel-Sachs model, it is impossible to know *a priori* whether any random tree will produce good results, therefore for practical uses (*i.e.*, within a DAW), we find Hornbostel-Sachs to be a suitable choice.

6.4 Hierarchical Loss Functions

To measure the impact of our proposed multi-task hierarchical loss, we compared it to a reasonable baseline "flat" loss. As our baseline approach, we treated hierarchical classification as a single-task, multilabel classification problem, where the ground truth is a multi-hot vector, with 1s for the leaf ground truth node and all of its ancestors in the tree, and 0s otherwise. Furthermore, we minimized the binary cross entropy between each individual predicted node and ground truth node. Note that this required us to

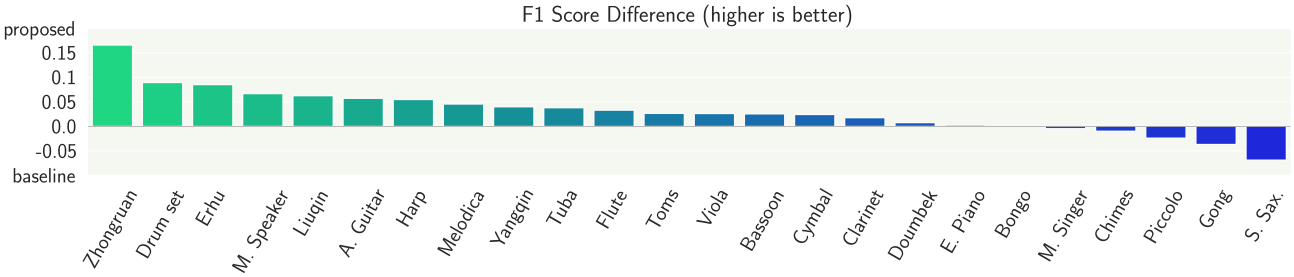

Figure 4. Difference in F1-score between our best proposed model ($H = 1$, $\alpha = 1$) and the baseline (Wang *et al.* [25]) on all instruments in the test set. Both models were evaluated with $N = 8$.

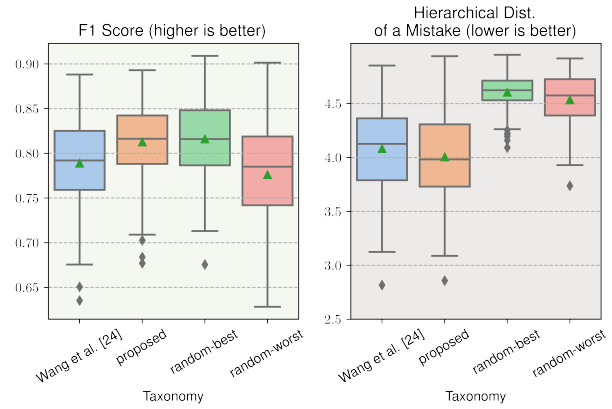

Figure 5. Comparison between the best and worst performing models trained on random hierarchies. The hierarchical distance of a mistake is calculated using the hierarchy the model was trained on. For the baseline (Wang *et al.* [25]) , we calculated the hierarchical distance of a mistake using the Hornbostel-Sachs hierarchy.

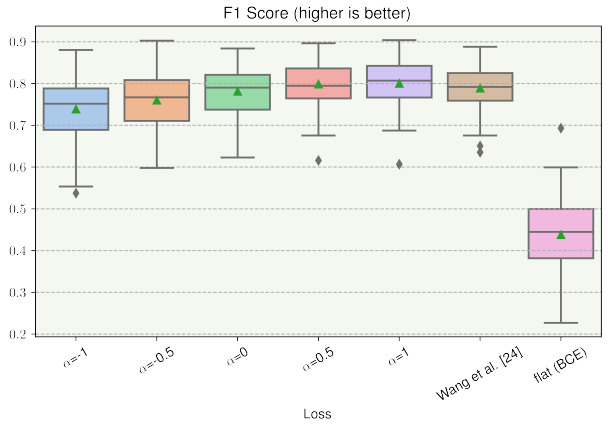

Figure 6. Evaluating the loss function. We vary α in our proposed hierarchical loss from negative (emphasize loss on broader categories) to positive (emphasize loss on finer categories) and additionally compare to a "flat" binary cross entropy (BCE) baseline.

use a sigmoid function instead of Eq. 2, which uses a softmax function. Additionally, we performed a hyperparameter search to find the best value of the α parameter for our proposed loss function (Section 6.4) using the search space $\alpha \in \{-1, -0.5, 0, 0.5, 1\}$. For this experiment, all trees were trained with $H = 4$ and evaluated with $N = 16$.

Results are shown in Figure 6. We observe that only the models with $\alpha > 0$ cause an improvement over Wang *et al*. [25]. Moreover, the flat loss causes a severe degradation in classification performance. This may be because training prototypical networks using a binary, one-vs-all formulation could yield a much less discriminative embedding space. Wang et al. [25] found a similar result: training prototypical networks with a binary formulation did not yield performance improvements.

6.5 Examining All Instrument Classes

In Figure 4, we examine the classification performance of every instrument in our test set. We compare our best model ($H = 1$, $\alpha = 1$) to the baseline model from Wang *et al.* [25], evaluated with $N = 8$. For clarity, we report the difference in F1 Score between the models. Our model beats the baseline on 18 of the 24 classes in the test set. In particular, our model shows a substantial improvement ($+16.56\%$) in F1 Score when classifying *zhongruan*, which may be rarely seen in a dataset composed of Western music. Figure 4 demonstrates that, overall, our hierarchical few-shot model is better at identifying a wider range of instrument classes than the baseline. This is important if we desire to make systems that are more robust to biases in the training data and, thus, can classify more a diverse set of instrument types.

7. CONCLUSION

We presented an approach for incorporating hierarchical structures in a few-shot learning model for the purpose of improving classification performance on classes outside of the training distribution. Our method builds on top of prototypical networks by computing prototypical representations at fine and coarse granularities, as defined by a class hierarchy. We showed that our proposed method yields statistically significant increases in classification performance and significant decreases mistake severity when evaluated on a classification task composed of unseen musical instruments. Moreover, we found that the choice of hierarchical structure is not arbitrary, and using a hierarchy based on the sound production mechanisms of musical instruments had the best results. We hope our work enables users with diverse cultural backgrounds with the ability to classify diverse collections of musical instruments. Future directions include examining new types of hierarchies, learning multiple hierarchies simultaneously, and the unsupervised discovery of hierarchies from unlabeled data.

8. ACKNOWLEDGEMENTS

This work was funded, in part, by USA National Science Foundation Award 1901456.

9. REFERENCES

[1] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "A survey of audio-based music classification and annotation," *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 303–319, 2010.

[2] A. Eronen and A. Klapuri, "Musical instrument recognition using cepstral coefficients and temporal features," in *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100)*, vol. 2. IEEE, 2000, pp. II753–II756.

[3] A. Krishna and T. V. Sreenivas, "Music instrument recognition: from isolated notes to solo phrases," in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 4. IEEE, 2004, pp. iv–iv.

[4] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.

[5] ——, "Tut database for acoustic scene classification and sound event detection," in *2016 24th European Signal Processing Conference (EUSIPCO)*. IEEE, 2016, pp. 1128–1132.

[6] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[7] S. Savage, *The art of digital audio recording: A practical guide for home and studio*. Oxford University Press, 2011.

[8] B. Owsinski, *The mixing engineer's handbook*. Nelson Education, 2013.

[9] A. Saha and A. M. Piper, "Understanding audio production practices of people with vision impairments," in *The 22nd International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '20), October 26–28, 2020, Virtual Event, Greece.* IEEE, 2020.

[10] A. Tanaka and A. Parkinson, "Haptic wave: A cross-modal interface for visually impaired audio producers," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 2016, pp. 2150–2161.

[11] J. J. Bosch, J. Janer, F. Fuhrmann, and P. Herrera, "A comparison of sound segregation techniques for predominant instrument recognition in musical audio signals." in *International Society for Music Information Retrieval (ISMIR) Conference*. Citeseer, 2012, pp. 559–564.

[12] E. Humphrey, S. Durand, and B. McFee, "Openmic-2018: An open data-set for multiple instrument recognition." in *International Society for Music Information*

Retrieval (ISMIR) Conference, Paris, France, 2018, pp. 438–444.

[13] Y.-N. Hung and Y. Yang, "Frame-level instrument recognition by timbre and pitch," in *Proceedings of the 20th International Society for Music Information Retrieval Conference*. Paris, France: ISMIR, 2018.

[14] S. Gururani, M. Sharma, and A. Lerch, "An attention mechanism for musical instrument recognition," in *International Society for Music Information Retrieval (ISMIR) Conference*, 2019.

[15] Y. Hung, Y. Chen, and Y. Yang, "Multitask learning for frame-level instrument recognition," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 381–385.

[16] M. Taenzer, J. Abeßer, S. I. Mimilakis, C. Weiß, M. Müller, and H. Lukashevich, "Investigating cnn-based instrument family recognition for western classical music recordings." in *Proceedings of the 20th International Society for Music Information Retrieval Conference*. ISMIR, 2019, pp. 612–619.

[17] A. Kratimenos, K. Avramidis, C. Garoufis, A. Zlatintsi, and P. Maragos, "Augmentation methods on monophonic audio for instrument classification in polyphonic music," in *2020 28th European Signal Processing Conference (EUSIPCO)*. IEEE, 2021, pp. 156–160.

[18] V. Lostanlen, J. Andén, and M. Lagrange, "Extended playing techniques: the next milestone in musical instrument recognition," in *Proceedings of the 5th International Conference on Digital Libraries for Musicology*, 2018, pp. 1–10.

[19] B. Kim and B. Pardo, "I-sed: An interactive sound event detector," in *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, ser. IUI '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 553–557.

[20] M. Cartwright, G. Dove, A. E. Méndez Méndez, J. P. Bello, and O. Nov, "Crowdsourcing multi-label audio annotation tasks with citizen scientists," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–11.

[21] E. M. von Hornbostel and C. Sachs, "Classification of musical instruments: Translated from the original german by anthony baines and klaus p. wachsmann," *The Galpin Society Journal*, vol. 14, pp. 3–29, 1961.

[22] M. J. Kartomi, *On concepts and classifications of musical instruments*. University of Chicago Press Chicago, 1990.

[23] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[24] O. Maimon and R. Lior, *Data Mining and Knowledge Discovery Handbook*. Springer, 2006, ch. Clustering methods.

[25] Y. Wang, J. Salamon, N. J. Bryan, and J. Pablo Bello, "Few-shot sound event detection," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 81–85.

[26] E. Benetos, M. Kotti, and C. Kotropoulos, "Musical instrument classification using non-negative matrix factorization algorithms and subset feature selection," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 5, 2006, pp. V–V.

[27] A. Eronen and A. Klapuri, "Musical instrument recognition using cepstral coefficients and temporal features," in *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*, vol. 2, 2000, pp. II753–II756 vol.2.

[28] V. Lostanlen and C.-E. Cella, "Deep convolutional networks on the pitch spiral for music instrument recognition," in *International Society for Music Information Retrieval (ISMIR) Conference*, 2016.

[29] S. Essid, G. Richard, and B. David, "Hierarchical classification of musical instruments on solo recordings," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 5, 2006, pp. V–V.

[30] Y. Han, J. Kim, K. Lee, Y. Han, J. Kim, and K. Lee, "Deep convolutional neural networks for predominant instrument recognition in polyphonic music," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 25, no. 1, p. 208–221, Jan. 2017.

[31] S. Gururani, C. Summers, and A. Lerch, "Instrument activity detection in polyphonic music using deep neural networks," in *Proceedings of the 20th International Society for Music Information Retrieval Conference*. Paris, France: ISMIR, 2018.

[32] R. A. Stein, P. A. Jaques, and J. F. Valiati, "An analysis of hierarchical text classification using word embeddings," *Information Sciences*, vol. 471, pp. 216–232, 2019.

[33] A. Dhall, A. Makarova, O. Ganea, D. Pavllo, M. Greeff, and A. Krause, "Hierarchical image classification using entailment cone embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 836–837.

[34] S. Sun, Q. Sun, K. Zhou, and T. Lv, "Hierarchical attention prototypical networks for few-shot text classification," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 476–485.

[35] L. Bertinetto, R. Mueller, K. Tertikas, S. Samangooei, and N. A. Lord, "Making better mistakes: Leveraging class hierarchies with deep networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 506–12 515.

[36] S. Essid, G. Richard, and B. David, "Instrument recognition in polyphonic music based on automatic taxonomies," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 68–80, 2005.

[37] T. Kitahara, M. Goto, and H. G. Okuno, "Category-level identification of non-registered musical instrument sounds," in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 4, 2004, pp. iv–iv.

[38] S. Fernández, A. Graves, and J. Schmidhuber, "Sequence labelling in structured domains with hierarchical recurrent neural networks," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI 2007*, 2007.

[39] R. Sanabria and F. Metze, "Hierarchical multitask learning with ctc," in *2018 IEEE Spoken Language Technology Workshop (SLT)*, 2018, pp. 485–490.

[40] K. Krishna, S. Toshniwal, and K. Livescu, "Hierarchical multitask learning for ctc-based speech recognition," *arXiv preprint arXiv:1807.06234*, 2018.

[41] E. Manilow, G. Wichern, and J. Le Roux, "Hierarchical musical instrument separation," in *International Society for Music Information Retrieval (ISMIR) Conference*, Oct. 2020, pp. 376–383.

[42] Y. Xu, Q. Huang, W. Wang, and M. D. Plumbley, "Hierarchical learning for dnn-based acoustic scene classification," *arXiv preprint arXiv:1607.03682*, 2016.

[43] A. Jati, N. Kumar, R. Chen, and P. Georgiou, "Hierarchy-aware loss function on a tree structured label space for audio event detection," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6–10.

[44] J. Cramer, V. Lostanlen, A. Farnsworth, J. Salamon, and J. P. Bello, "Chirping up the right tree: Incorporating biological taxonomies into deep bioacoustic classifiers," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 901–905.

[45] K. Cheng, S. Chou, and Y. Yang, "Multi-label few-shot learning for sound event recognition," in *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, 2019, pp. 1–5.

[46] B. Shi, M. Sun, K. C. Puvvada, C.-C. Kao, S. Matsoukas, and C. Wang, "Few-shot acoustic event detection via meta learning," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 76–80.

[47] Y. Wang, J. Salamon, M. Cartwright, N. J. Bryan, and J. P. Bello, "Few-shot drum transcription in polyphonic music," in *International Society for Music Information Retrieval (ISMIR) Conference*, 2020.

[48] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. Bello, "Medleydb: A multitrack dataset for annotation-intensive mir research," in *15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.

[49] R. Bittner, J. Wilkins, H. Yip, and J. P. Bello, "Medleydb 2.0 audio," Aug. 2016. [Online]. Available: https://doi.org/10.5281/zenodo.1715175

[50] R. Bittner, E. Humphrey, and J. Bello, "Pysox: Leveraging the audio signal processing power of sox in python," in *Proceedings of the International Society for Music Information Retrieval Conference Late Breaking and Demo Papers*, 2016.