SOUND EVENT CLASSIFICATION USING ONTOLOGY-BASED NEURAL NETWORKS

Abelino Jiménez[⋆], *Benjamin Elizalde*[⋆], *Bhiksha Raj*

Carnegie Mellon University, ECE Department
Email: abjimenez@cmu.edu, bmartin1@andrew.cmu.edu, bhiksha@cs.cmu.edu

ABSTRACT

State of the art sound event classification relies in neural networks to learn the associations between class labels and audio recordings within a dataset. These datasets typically define an ontology to create a structure that relates these sound classes with more abstract super classes. Hence, the ontology serves a source of domain knowledge representation of sounds. However, the ontology information is rarely considered, and specially under explored to model neural network architectures. We propose ontology-based neural network architectures for sound event classification. We defined a framework to design simple network architectures that preserve an ontological structure. The networks are trained and evaluated using the MSoS dataset. Results show an improvement in accuracy demonstrating the benefits of the ontology.

1. METHOD

In this section we present a framework to deal with ontological information using deep learning architectures.

1.1. Framework and Assumptions

The framework is defined to make use of the ontology structure and to model the neural network architectures. It should be noted that we considered ontologies with two levels, which are the most common in sound event datasets. Nevertheless, the presented framework can be easily generalized to more levels.

In our framework, we considered the training data $\{(\mathbf{x}_1, \mathbf{y}_1), ..., (\mathbf{x}_n, \mathbf{y}_n)\}$, where $\mathbf{x}_i \in \mathcal{X}$ is an audio representation, which is associated to a set of labels given by the ontology $\mathbf{y}_i \in C_1 \times C_2 \times ... \times C_k$. In this case, C_i is the set of possible classes at i-level. Assuming a hierarchical relation, we can consider that each possible class in C_i is mapped to one element in C_{i+1}. The higher the value of i, the higher the level in the ontology.

For an example, consider the an ontology where $k = 2$, $C_1 = \{$*cat* , *dog* , *breathing* , *eating* , *sneezing* , *violin* , *drums* , *piano* , *beep* , *boing* , *train* , *siren*$\}$ and $C_2 = \{$*nature* , *human* , *music* , *effects* , *urban*$\}$. Here every element in C_1 is related to one element in C_2; e.g., *cat* belongs to *nature*, or *drums* belongs to *music*.

Furthermore, for a given representation $\mathbf{x} \in \mathcal{X}$, if we know the corresponding label y_1 in C_1, we can infer its label in C_2. This intuition can be formalized using a probabilistic formulation, where it is straight forward to see that, assuming $p(y_2|y_1, \mathbf{x}) = p(y_2|y_1)$, the following is satisfied:

$$p(y_2|\mathbf{x}) = \sum_{y_1} p(y_2|y_1, \mathbf{x}) \cdot p(y_1|\mathbf{x}) \quad (1)$$

$$= \sum_{y_1 \in \text{children}(y_2)} p(y_1|\mathbf{x}) \quad (2)$$

Therefore, if we want to estimate $p(y_2|\mathbf{x})$ using a model, we just need to compute the estimation of $p(y_1|\mathbf{x})$ and sum the values corresponding to the children of y_2. This case is valid for inference time, however, it is not clear that using the representation and label $(\mathbf{x}, y_1)$ should be enough to train the model. If at training time we can make use of knowledge to relate the different classes in y_1, it should improve the performance of the model, specially at making predictions for classes y_2. In the following sections we take our proposed framework and use it to design ontology-based neural network architectures.

[⋆]Authors contributed equally.

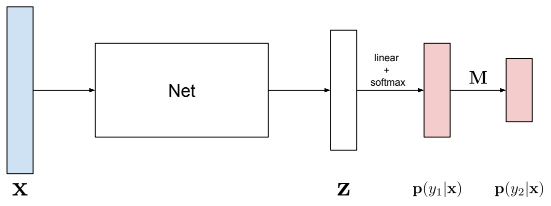

Fig. 1. Architecture of the Feed-forward Network with Ontological Layer. The blue column represents the acoustic feature vector, the red columns are the output probabilities for both levels.

1.2. Feed-forward Network with Ontological Layer

A Feed-forward Network (FFN) with Ontological Layer consists of a base network (Net), an intermediate vector $\mathbf{z}$, and two outputs, one for each ontology level. The base network weights are learned at every parameter update and utilizes an input vector of audio features $\mathbf{x}$ and generates a vector $\mathbf{z}$. This vector is used to generate two outputs, $\mathbf{p}(y_1|\mathbf{x})$ a probability vector for C_1 and $\mathbf{p}(y_2|\mathbf{x})$ a probability vector for C_2. First, the vector $\mathbf{z}$ is passed to a softmax layer of the size of C_1. Second, the same vector $\mathbf{z}$ is multiplied by the *ontological layer* $\mathbf{M}$ and generates a layer of size of C_2. Once the FFN is trained, it can be used to predict any class C_1 and C_2 for any input $\mathbf{x}$.

The *ontological layer* reflects the relations between classes and sub classes given by the ontology. To describe how we used this layer, we refer to Equation 2, where $p(y_2|\mathbf{x})$ is the sum of all the values of $p(y_1|\mathbf{x})$ corresponding to the children of y_2. If we consider this equation as a directed graph where $\mathbf{M}$ is the $|C_2| \times |C_1|$ incidence matrix, then, it is clear that Equation 2 can be rewritten as,

$$\mathbf{p}(y_2|\mathbf{x}) = \mathbf{M} \cdot \mathbf{p}(y_1|\mathbf{x}) \quad (3)$$

Note that the *ontological layer* $\mathbf{M}$ defines the weights of a standard layer connection. Although we do not consider that these weights are trainable, they are part of our training data.

In order to train this model, we simply propose to apply gradient-based method to minimize the loss function $\mathcal{L}$, which is a convex combination between two categorical cross-entropy functions; $\mathcal{L}_1$ the categorical cross entropy corresponding to $\mathbf{p}(y_1|\mathbf{x})$ and $\mathcal{L}_2$ corresponding to $\mathbf{p}(y_2|\mathbf{x})$. Formally,

$$\mathcal{L} = \lambda \mathcal{L}_1 + (1 - \lambda)\, \mathcal{L}_2 \quad (4)$$

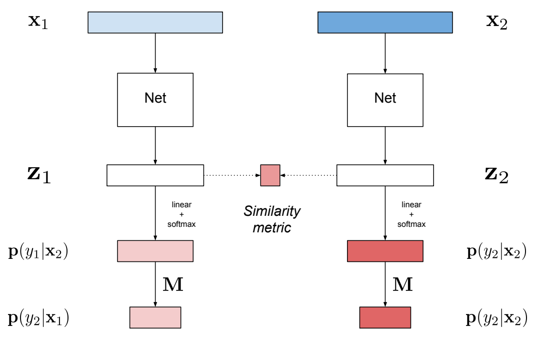
$\mathbf{x}_1$
$\mathbf{x}_2$
Net
Net
$\mathbf{z}_1$
$\mathbf{z}_2$
linear + softmax
Similarity metric
linear + softmax
$\mathbf{p}(y_1|\mathbf{x}_2)$
$\mathbf{p}(y_2|\mathbf{x}_2)$
M
M
$\mathbf{p}(y_2|\mathbf{x}_1)$
$\mathbf{p}(y_2|\mathbf{x}_2)$

Fig. 2. Architecture of the SNN that is used to learn ontological embeddings. The SNN is trained with three types of pairs depending on whether the inputs are from the same subclass, or different subclass, but same super class, or different super class.

1.3. Ontology-based embeddings

In this section, we describe how we learned the ontology-based embeddings. The embeddings are computed using a Siamese neural network (SNN), shown in Fig. 2, consisting of twin networks that have the same base architecture (Net) with shared weights. The weights are learned simultaneously at every parameter update. Each base network utilizes an input vector of audio features $\mathbf{x}$. Then, for the inputs $\mathbf{x}_1$ and $\mathbf{x}_2$, we obtained the outputs $\mathbf{p}(y_1|\mathbf{x}_1)$, $\mathbf{p}(y_1|\mathbf{x}_2)$, $\mathbf{p}(y_2|\mathbf{x}_1)$ and $\mathbf{p}(y_2|\mathbf{x}_2)$. In addition, we considered as output the similarity metric between the hidden vectors $\mathbf{z}_1$ and $\mathbf{z}_2$ as illustrated Figure 2.

For training, first, we needed to associate a loss function to every output. The innovation of this model is the loss for the similarity metric. Our attempt is that the similarity metric describes the ontology; the difference of the embeddings $\mathbf{z}_1$ and $\mathbf{z}_1$ should indicate how different $\mathbf{x}_1$ and $\mathbf{x}_2$ are with respect to the ontology. In a 2-level ontology there are 3 possible distances, for this work we chose 0 or 1 or 10 depending on whether the inputs are from the same subclass, or different subclass, but same super class, or different super class. These values can be tuned. Hence, the model attempts to approximate the distance within the ontology using the distance between embeddings.

In order to train the full model, we need to provide pairs of audio examples and apply a gradient-based method to minimize the loss function $\mathcal{L}$. We propose to use a linear combination between four categorical cross-entropy functions: $\mathcal{L}_1^1$ and $\mathcal{L}_1^2$ the categorical cross-entropy corresponding to $\mathbf{p}(y_1|\mathbf{x}_1)$ and $\mathbf{p}(y_1|\mathbf{x}_2)$ respectively, and $\mathcal{L}_2^1$ and $\mathcal{L}_2^2$ corresponding to $\mathbf{p}(y_2|\mathbf{x}_1)$ and $\mathbf{p}(y_2|\mathbf{x}_2)$, and finally the similarity metric D_w given by Euclidean Distance. Formally,

$$\mathcal{L} \;=\; \lambda_1(\mathcal{L}_1^1+\mathcal{L}_1^2)+\lambda_2(\mathcal{L}_2^1+\mathcal{L}_2^2)+\lambda_3 D_w \qquad (5)$$

2. EXPERIMENTS AND RESULTS

In this section, we evaluate the sound event classification performance of the ontological-based neural network architectures.

2.1. Dataset and Audio Features

We used the Making Sense of Sounds Challenge daataset. For the audio we used state-of-the-art Walnet features [1] to represent audio recordings. For each audio, we computed a 128-dimensional logmel-spectrogram vector and transformed it *via* a convolutional neural network (CNN) that was trained separately on the balanced set of AudioSet. The network comprised 8 convolutional layers, resulting in an output feature vector of dimensionality 527. To this, we concatenated intermediate outputs from the 8^{th} layer of the CNN with 1024 dims.

System ID	Model	Level 1 Accuracy	Level 2 Accuracy
	Baseline (Challenge)	-	0.810
MLSP_ONTLAYER_1	FF + Ontology	0.740	0.913
MLSP_ONTEMB_1	Ontology Embeddings	0.736	0.886

Table 1. Both proposed methods outperformed the baseline

2.2. Base Network Architecture (Net)

The architecture of the base network (Net) considered in this experiment, shown in Fig. 1, is a feed-forward multi-layer perceptron network. It consists of 4 layers: the input layer of dimensionality 1024, which takes audio feature vectors, 2 dense layers of dimensionality 512 and 256, respectively, and the output layer of dimensionality 128, which is the dimensionality of the vector $\mathbf{z}$. The dense layers utilize Batch Normalization, a dropout rate of 0.5 and the ReLU activation function; $\max(0,x)$, where x is input to the function. We tuned the parameters in the 'Net' box as well as the parameters that transform $\mathbf{z}$ into $\mathbf{p}(y_1|\mathbf{x})$.

2.3. Performance of Feed-forward Model with Ontological layer

To validate the architecture presented in Section 1.2 and analyze the utility of the ontological layer, we trained models taking different values of λ. In general, we observe that considering values different from 0 and 1 helps to increase the performance. The best performance was obtained using $\lambda = 0.8$, getting 74.0% and 91.3% of accuracy in level 1 and 2 respectively. Thus, using the ontological structure we can get an absolute improvement of 5.4% and 6% respect baseline models.

2.4. Performance of Ontology-based embeddings

We tested the architecture described in Section 1.3 to evaluate the performance of the ontology-based embeddings for sound event classification.

We processed the Walnet audio features and chose different super and sub class pairs to train the Siamese neural network to produce the ontology-based embeddings. The architecture of the base network (Net) is the same as the one used in the previous section. We trained the SNN for 50 epochs using the Adam algorithm. We also tuned the hyper-parameters of the SNN to achieve good performance with the input features that are described in the next section. We also tried different number of pairs for the input training data, from 100 to 1,000,000 pairs and found that 100,000 yielded the best performance. For the loss function we used the considered the values computed in the previous experiment. We used the value of 0.8 for the lambda of the classifiers of level 2 and 0.2 for the classifiers in level 1, and 0.2 for the similarity metric. Modifying the lambdas in the loss function affected the overall performance.

The results in Table 1 show that the accuracy performance of MSoS as follows, in level 1 73.6% and in level 2 88.6%. Based on these results we make the following conclusions. The performance of this architecture is better than the baseline, but slightly under performs the original method of FF+Ontology. Nevertheless, the ontology-based embeddings have the added benefit of better grouping in contrast to the plain base network output vectors.

3. REFERENCES

[1] Ankit Shah, Anurag Kumar, Alexander G Hauptmann, and Bhiksha Raj, "A closer look at weak label learning for audio events," *arXiv preprint arXiv:1804.09288*, 2018.