Time Aggregation Operators for Multi-label Audio Event Detection

Pankaj Joshi, Digvijaysingh Gautam, Ganesh Ramakrishnan, Preethi Jyothi

Dept. of Computer Science and Engineering, Indian Institute of Technology Bombay
{panjoshi,digvijay,ganesh,pjyothi}@cse.iitb.ac.in

Abstract

In this paper, we present a state-of-the-art system for audio event detection. The labels on the training (and evaluation) data specify the set of events occurring in each audio clip, but neither the time spans nor the order in which they occur. Specifically, our task of weakly supervised learning is the “Detection and Classification of Acoustic Scenes and Events (DCASE) 2017” challenge [5]. We use the winning entry in this challenge given by Xu et al. [10] as our starting point and identify several important modifications that allow us to improve on their results significantly. Our techniques pertain to aggregation and consolidation over time and frequency signals over a (temporal) sequence before decoding the labels. In general, our work is also relevant to other tasks involving learning from weak labeling of sequential data.

1. Introduction

Multi-label audio event detection is a task that comprises of detecting various audio classes present in the audio file. Machine learning (ML) models can be trained using some labeled data to do this audio class detection. However, in practice, the available labeling may be weak, that is, the class labels might not be available with associated timestamps of the audio events. A task of learning to predicting these labels under weak supervision was presented as the Audio tagging task in the Detection and Classification of Acoustic Scenes and Events (DCASE) 2017 challenge [5]. The challenge was to evaluate systems for large-scale detection of sound events using weakly labeled training data. The best performing system in this challenge by Xu et al. [10] models the temporal structure of each sound clip using a recurrent neural network. It also benefits in robustness and numerical stability by making use of gated convolutional units in the earlier ReLU activations.

1.1. Related Work

Adavanne et al. [1] and Parascandolo et al. [7] propose the first uses of convolutional architectures for sound event detection using weakly labeled data. They benefited from the use of 2-d convolutions over audio spectrograms presumably because the audio class is influenced by inter-operations on both the time and frequency domains. As mentioned above, the winner [10] of the challenge replaced ReLU activations with a gated unit comprising of a linear and sigmoid. This is done to introduce attention to each layer of the neural network. They get additional benefits by employing a recurrent neural network later to model the temporal structure of each sound clip.

While most approaches have implicitly associated the clip-level labels with every segment in it, some like Yu et al. [11], Feng et al. [3] and Tseng et al. [9] have viewed a clip as a set of instances, where each instance is a fixed image/audio segment and approached the problem as a multi-instance, multi-labeled (MIML) problem. However, this treatment did not yield the best reported results.

Xu et al. [10] also predicted the time span for the audio activity using a soft argmax operator (which they refer to as softmax based attention).

1.2. Our Contributions

- We demonstrate significant benefits of the proposed operators using extensive experimentation and also present ablation tests to draw further insights.
- We believe that our proposed operators can complement existing best practices for many audio and video classification tasks.

2. Model architecture

Our architecture is inspired by the publicly available code of Xu et al. [10] which is the winning entry of the challenge. This architecture used the Gated Linear Unit (GLU) of Dauphin et al. [2] in place of the Relu activation by Nair et al. [6]. GLUs use a sigmoid gate, the value of which determines the flow of information. A GLU can be defined as:

\[ Y = (W * X + b) \odot \sigma(V * X + c) \]

Here \( \sigma \) is the sigmoid operator whereas \( \odot \) is the element wise multiplication and \( X \) is the input. We use 4 Gated Convolution blocks, each block containing two gated convolutions followed by an operator that consolidates signals from both time and frequency units and is therefore referred to as a t-f operator. Each convolution block contains two convolution layers with 64 filters of size 3 × 3 with gated linear units as activation. Our suite of t-f operators includes max pool over frequency, max pool over time, mean over time as well as some of their compositions such as max across frequency with mean across time, max pool across both frequency and time, etc.

Following the four gated convolutional blocks described earlier is another convolution layer of 256 filters of size 3 × 3 followed by another max pool across frequency. The output from this convolutional network is fed into a BiRNN block with 128 GRU units and GLUs in place of activation. The output from the previous layer is fed into fully connected layers with softmax and sigmoid activations. The weights in the softmax layer are used to compute a weighted mean of the sigmoid layer.
to finally get the label predictions. The softmax can also be used to predict time stamps for sound events in the audio clip. For this, we take cue from the work of Xu et al. [10], and check if jointly training a model to classify the audio segment while also learning to predict the time-stamp using a soft-argmax operator mutually help each other. While we find marginal gains using this joint learning, we translate this somewhat insightful experiment into another alternative (the mean over time).

In the first variant of the model, following the BiRNN layer, we use an operator that consolidates signals on the time unit, hereafter referred to as a \( t \)-operator. The suite of \( t \)-operators include mean over time [8], a similar max over time and their compositions such as max over mean over time. In this case, we use max over frequency as \( t-f \) operator. The output from the \( t \)-operator is fed into a fully connected layer with sigmoid unit that is trained to predict labels.

In another variant of this architecture, instead the Gated Convolution blocks, we use 5 convolutional blocks, wherein each block contains two convolutional layers with same number of filters followed by a max over frequency. The number of filters is gradually increased in each block from 16 to 256. In this architecture, we use ReLU as our activation function for the convolutional blocks. This layer is followed by BiRNN layer and the rest of the architecture is same as the architecture for the \( t \)-operator. We call this model ReLU convolutions model.

We suggest one more variant for the ReLU convolutions model where we replace the Gated Linear units in the BiRNN block with ReLU we call it ReLU CNRNN model.

In Figure 1, we present our network architecture highlighting the place of \( t-f \) operators. Whereas, in Figure 2, we remove the fully connected block at the end of the previous network and instead feed output from the BiRNN block to one of our proposed \( t \)-operators.

3. Experiments

3.1. Dataset

For our experiments, we used the training, development and evaluation sets predefined by DCASE 2017 for the audio event detection challenge [5]. These datasets are fixed subsets of the Google AudioSet corpus [4] and consist of sound clips from 17 different audio events relating to vehicle sounds (‘Car’, ‘Train’, etc.) and warning sounds (‘Fire engine’, ‘Ambulance siren’, etc.). The training, development and evaluation sets contain a total of 51172, 488 and 1103 audio clips, respectively. These datasets are weakly labeled in that only the types of the audio events occurring in the clip are provided, but the timestamp of the events are not specified. In each of these datasets, the label distribution across the 17 audio event classes is highly unbalanced. For example, the ‘Car’ class is most frequent with a total of 25,744 clips whereas the ‘Car alarm’ class is least frequent with a total of 273 clips.

3.2. Implementation Details

Log-mel filterbank features extracted from the raw audio waveforms were fed as input to our architecture. Each training clip has 240 time steps with 64 mel frequency bands in each time step. Our model implementation largely borrows from the publicly available code of the winning system [10]. We adopt three training strategies that were initially proposed for the winning system: 1) Batches were created such that even the least frequent classes were included in each batch. 2) Instead of setting a fixed threshold across all classes, class-specific thresholds were determined by tuning on the development set. 3) Predictions across epochs for the same system were averaged in order to improve its stability.
### 3.3. Results

We first evaluate the utility of our \( t-f \) operators which appear before the BiRNN block in our architecture. Tables 1 and 2 show the F-1 score/precision/recall breakup for different \( t-f \) operators. The first two rows correspond to the current state-of-the-art scores for this task, as reported in Xu et al. [10], from their best-performing individual system and fusion system, respectively. The third row corresponds to our extension of the best individual system from Xu et al. [10]. Our reimplementation leads to slightly higher numbers on both the development and evaluation sets; this system is henceforth referred to as “Benchmark”. Each of the \( t-f \) operators that we have suggested, with the exception of mean over frequency, outperforms the “benchmark” on the development set but not always on the evaluation set. We believe this could be due to differences in the distribution of sound events across the development and evaluation sets. Since we tune threshold values on the development set, this difference in class distribution could likely affect performance on the evaluation set.

Tables 3 and 4 show results from using our \( t \)-operators following the BiRNN block in our architecture. For easy comparison, we show the state-of-the-art scores from Xu et al. [10] again in both the tables. On both the development and evaluation sets, we observe consistent improvements for every choice of our \( t \)-operator. The “max over time” \( t \)-operator improves even over the “state of the art (Fusion)” system shown in Tables 1 and 2. The “max over time” \( t \)-operator improves only marginally compared to the “mean over time” \( t \)-operator on the development set (i.e. 0.599 vs. 0.591), but provides a larger improvement on the evaluation set (0.575 vs. 0.561). The “max over mean” \( t \)-operator was designed to combine the best properties of both the preceding operators: Average a fixed set of time-indexed vectors in a sliding window across time and use a coordinate-wise maximum operation to reduce it to a single vector. This is the best-performing \( t \)-operator both on the development and evaluation sets.

In Tables 5 and 6, we experiment with the “ReLU convolution” and “ReLU CNRNN” variants of the convolutional net-work that we described earlier in Section 2. We used the “max over mean” \( t \)-operator (which performed the best, as shown in Tables 3 and 4) with both these variants. We find additional benefits in performance from using these two variants with the \( t \)-operator. To assess the importance of the \( t \)-operator, we also experimented with using the “ReLU CNRNN” networks without adding the \( t \)-operator after the BiRNN layers: this did not perform as well, the results are reported in Table 5 and 6.

### 3.4. Class-wise Analysis

In Tables 7 and 8, we summarize the class-wise F1 scores of all the different models on the development set and the evaluation set, respectively. For each class, the best F1 score is highlighted in bold. The column “Net wins” lists the net number of classes on which each model outperforms the benchmark model.

From the tables, we observe the following:

- Firstly, we observe that all our systems have a positive

### Table 1: Comparison of models on development Set for different \( t-f \) operators

<table>
<thead>
<tr>
<th>Model</th>
<th>F1 Score</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>State of the art [10]</td>
<td>0.567</td>
<td>0.538</td>
<td>0.601</td>
</tr>
<tr>
<td>State of the art (Fusion) [10]</td>
<td>0.577</td>
<td>0.565</td>
<td>0.589</td>
</tr>
<tr>
<td>Max across both (Benchmark)</td>
<td>0.570</td>
<td>0.531</td>
<td>0.614</td>
</tr>
<tr>
<td>Max Frequency</td>
<td>0.575</td>
<td>0.540</td>
<td>0.614</td>
</tr>
<tr>
<td>Avg Time Max Freq</td>
<td>0.582</td>
<td>0.545</td>
<td>0.625</td>
</tr>
<tr>
<td>Avg Freq</td>
<td>0.527</td>
<td>0.487</td>
<td>0.574</td>
</tr>
</tbody>
</table>

### Table 2: Comparison of models on evaluation Set for different \( t-f \) operators

<table>
<thead>
<tr>
<th>Model</th>
<th>F1 Score</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>State of the art [10]</td>
<td>0.542</td>
<td>0.589</td>
<td>0.502</td>
</tr>
<tr>
<td>State of the art (Fusion) [10]</td>
<td>0.556</td>
<td>0.614</td>
<td>0.508</td>
</tr>
<tr>
<td>Max across both (Benchmark)</td>
<td>0.570</td>
<td>0.534</td>
<td>0.611</td>
</tr>
<tr>
<td>Max Frequency</td>
<td>0.546</td>
<td>0.506</td>
<td>0.592</td>
</tr>
<tr>
<td>Avg Time Max Freq</td>
<td>0.557</td>
<td>0.516</td>
<td>0.605</td>
</tr>
<tr>
<td>Avg Freq</td>
<td>0.519</td>
<td>0.473</td>
<td>0.574</td>
</tr>
</tbody>
</table>

### Table 3: Comparison of models on the development set for different \( t \)-operators

<table>
<thead>
<tr>
<th>Model</th>
<th>F1 Score</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>State of the art [10]</td>
<td>0.542</td>
<td>0.589</td>
<td>0.502</td>
</tr>
<tr>
<td>State of the art (Fusion) [10]</td>
<td>0.556</td>
<td>0.614</td>
<td>0.508</td>
</tr>
<tr>
<td>Mean over time</td>
<td>0.561</td>
<td>0.517</td>
<td>0.615</td>
</tr>
<tr>
<td>Max over time</td>
<td>0.575</td>
<td>0.541</td>
<td>0.613</td>
</tr>
<tr>
<td>Max over mean over time</td>
<td>0.590</td>
<td>0.567</td>
<td>0.616</td>
</tr>
</tbody>
</table>

### Table 4: Comparison of models on the evaluation set for different \( t \)-operators

<table>
<thead>
<tr>
<th>Model</th>
<th>F1 Score</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>State of the art [10]</td>
<td>0.567</td>
<td>0.538</td>
<td>0.601</td>
</tr>
<tr>
<td>State of the art (Fusion) [10]</td>
<td>0.577</td>
<td>0.565</td>
<td>0.589</td>
</tr>
<tr>
<td>Mean over time</td>
<td>0.591</td>
<td>0.556</td>
<td>0.630</td>
</tr>
<tr>
<td>Max over time</td>
<td>0.599</td>
<td>0.574</td>
<td>0.623</td>
</tr>
<tr>
<td>Max over mean over time</td>
<td>0.601</td>
<td>0.590</td>
<td>0.614</td>
</tr>
</tbody>
</table>

### Table 5: Performance of the ReLU based variants on the development set

<table>
<thead>
<tr>
<th>Model</th>
<th>F1 Score</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReLU Convolution</td>
<td>0.610</td>
<td>0.573</td>
<td>0.652</td>
</tr>
<tr>
<td>ReLU CNRNN (no ( t )-operator)</td>
<td>0.601</td>
<td>0.565</td>
<td>0.642</td>
</tr>
<tr>
<td>ReLU CNRNN ((2,2) pool)</td>
<td>0.589</td>
<td>0.583</td>
<td>0.594</td>
</tr>
<tr>
<td>ReLU CNRNN (with ( t )-operator)</td>
<td>0.601</td>
<td>0.576</td>
<td>0.629</td>
</tr>
</tbody>
</table>

### Table 6: Performance of the ReLU based variants on the evaluation set

<table>
<thead>
<tr>
<th>Model</th>
<th>F1 Score</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReLU Convolution</td>
<td>0.576</td>
<td>0.535</td>
<td>0.623</td>
</tr>
<tr>
<td>ReLU CNRNN (no ( t )-operator)</td>
<td>0.582</td>
<td>0.538</td>
<td>0.634</td>
</tr>
<tr>
<td>ReLU CNRNN ((2,2) pool)</td>
<td>0.574</td>
<td>0.561</td>
<td>0.589</td>
</tr>
<tr>
<td>ReLU CNRNN (with ( t )-operator)</td>
<td>0.591</td>
<td>0.562</td>
<td>0.624</td>
</tr>
</tbody>
</table>
Table 7: Class-wise comparison of models on development set

<table>
<thead>
<tr>
<th>Model</th>
<th>Train horn</th>
<th>Air horn</th>
<th>Truck horn</th>
<th>Car alarm</th>
<th>Reversing beeps</th>
<th>Ambulance</th>
<th>Fire engine</th>
<th>Civil defense siren</th>
<th>Police car</th>
<th>Screaming</th>
<th>Car passing by</th>
<th>Bus</th>
<th>Truck</th>
<th>Motor- cycle</th>
<th>Train</th>
<th>Net wins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max across both (Benchmark)</td>
<td>0.79</td>
<td>0.53</td>
<td>0.62</td>
<td>0.67</td>
<td>0.51</td>
<td>0.76</td>
<td>0.55</td>
<td>0.61</td>
<td>0.82</td>
<td>0.63</td>
<td>0.64</td>
<td>0.40</td>
<td>0.36</td>
<td>0.51</td>
<td>0.44</td>
<td>0.52</td>
</tr>
<tr>
<td>-operator: mean over time</td>
<td>0.84</td>
<td>0.63</td>
<td>0.65</td>
<td>0.76</td>
<td>0.54</td>
<td>0.70</td>
<td>0.56</td>
<td>0.59</td>
<td>0.84</td>
<td>0.61</td>
<td>0.69</td>
<td>0.48</td>
<td>0.42</td>
<td>0.42</td>
<td>0.45</td>
<td>0.50</td>
</tr>
<tr>
<td>-operator: max over time</td>
<td>0.87</td>
<td>0.55</td>
<td>0.64</td>
<td>0.77</td>
<td>0.52</td>
<td>0.76</td>
<td>0.53</td>
<td>0.56</td>
<td>0.81</td>
<td>0.60</td>
<td>0.72</td>
<td>0.48</td>
<td>0.50</td>
<td>0.49</td>
<td>0.51</td>
<td>0.59</td>
</tr>
<tr>
<td>-operator: max after mean</td>
<td>0.81</td>
<td>0.64</td>
<td>0.71</td>
<td>0.71</td>
<td>0.51</td>
<td>0.72</td>
<td>0.60</td>
<td>0.60</td>
<td>0.79</td>
<td>0.55</td>
<td>0.77</td>
<td>0.52</td>
<td>0.42</td>
<td>0.43</td>
<td>0.48</td>
<td>0.56</td>
</tr>
<tr>
<td>ReLU CNRNN (no -operator)</td>
<td>0.84</td>
<td>0.56</td>
<td>0.79</td>
<td>0.73</td>
<td>0.56</td>
<td>0.76</td>
<td>0.50</td>
<td>0.53</td>
<td>0.79</td>
<td>0.63</td>
<td>0.71</td>
<td>0.50</td>
<td>0.36</td>
<td>0.48</td>
<td>0.52</td>
<td>0.59</td>
</tr>
<tr>
<td>ReLU CNRNN (with -operator)</td>
<td>0.83</td>
<td>0.53</td>
<td>0.69</td>
<td>0.69</td>
<td>0.60</td>
<td>0.78</td>
<td>0.48</td>
<td>0.65</td>
<td>0.83</td>
<td>0.63</td>
<td>0.71</td>
<td>0.49</td>
<td>0.38</td>
<td>0.44</td>
<td>0.53</td>
<td>0.50</td>
</tr>
<tr>
<td>ReLU CNRNN (2:2) pool</td>
<td>0.77</td>
<td>0.57</td>
<td>0.57</td>
<td>0.71</td>
<td>0.53</td>
<td>0.72</td>
<td>0.55</td>
<td>0.53</td>
<td>0.80</td>
<td>0.60</td>
<td>0.78</td>
<td>0.44</td>
<td>0.36</td>
<td>0.55</td>
<td>0.55</td>
<td>0.56</td>
</tr>
<tr>
<td>ReLU Convolution (with -operator)</td>
<td>0.86</td>
<td>0.56</td>
<td>0.73</td>
<td>0.76</td>
<td>0.56</td>
<td>0.74</td>
<td>0.62</td>
<td>0.58</td>
<td>0.85</td>
<td>0.62</td>
<td>0.78</td>
<td>0.51</td>
<td>0.39</td>
<td>0.50</td>
<td>0.50</td>
<td>0.52</td>
</tr>
</tbody>
</table>

Table 8: Class-wise comparison of models on evaluation set

<table>
<thead>
<tr>
<th>Model</th>
<th>Train horn</th>
<th>Air horn</th>
<th>Truck horn</th>
<th>Car alarm</th>
<th>Reversing beeps</th>
<th>Ambulance</th>
<th>Fire engine</th>
<th>Civil defense siren</th>
<th>Police car</th>
<th>Screaming</th>
<th>Car passing by</th>
<th>Bus</th>
<th>Truck</th>
<th>Motor- cycle</th>
<th>Train</th>
<th>Net wins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max across both (Benchmark)</td>
<td>0.63</td>
<td>0.52</td>
<td>0.51</td>
<td>0.37</td>
<td>0.37</td>
<td>0.57</td>
<td>0.67</td>
<td>0.59</td>
<td>0.85</td>
<td>0.57</td>
<td>0.77</td>
<td>0.62</td>
<td>0.31</td>
<td>0.34</td>
<td>0.50</td>
<td>0.60</td>
</tr>
<tr>
<td>-operator: mean over time</td>
<td>0.62</td>
<td>0.53</td>
<td>0.64</td>
<td>0.42</td>
<td>0.37</td>
<td>0.66</td>
<td>0.62</td>
<td>0.60</td>
<td>0.80</td>
<td>0.36</td>
<td>0.32</td>
<td>0.62</td>
<td>0.27</td>
<td>0.42</td>
<td>0.44</td>
<td>0.59</td>
</tr>
<tr>
<td>-operator: max over time</td>
<td>0.63</td>
<td>0.58</td>
<td>0.62</td>
<td>0.40</td>
<td>0.39</td>
<td>0.64</td>
<td>0.58</td>
<td>0.61</td>
<td>0.84</td>
<td>0.48</td>
<td>0.84</td>
<td>0.64</td>
<td>0.27</td>
<td>0.37</td>
<td>0.48</td>
<td>0.65</td>
</tr>
<tr>
<td>-operator: max after mean</td>
<td>0.65</td>
<td>0.58</td>
<td>0.65</td>
<td>0.40</td>
<td>0.36</td>
<td>0.64</td>
<td>0.57</td>
<td>0.59</td>
<td>0.80</td>
<td>0.52</td>
<td>0.81</td>
<td>0.64</td>
<td>0.27</td>
<td>0.39</td>
<td>0.48</td>
<td>0.65</td>
</tr>
<tr>
<td>ReLU CNRNN (no -operator)</td>
<td>0.60</td>
<td>0.57</td>
<td>0.59</td>
<td>0.47</td>
<td>0.46</td>
<td>0.62</td>
<td>0.56</td>
<td>0.48</td>
<td>0.77</td>
<td>0.64</td>
<td>0.79</td>
<td>0.65</td>
<td>0.26</td>
<td>0.43</td>
<td>0.49</td>
<td>0.58</td>
</tr>
<tr>
<td>ReLU CNRNN (with -operator)</td>
<td>0.63</td>
<td>0.60</td>
<td>0.59</td>
<td>0.48</td>
<td>0.48</td>
<td>0.60</td>
<td>0.57</td>
<td>0.60</td>
<td>0.85</td>
<td>0.53</td>
<td>0.80</td>
<td>0.65</td>
<td>0.25</td>
<td>0.41</td>
<td>0.47</td>
<td>0.64</td>
</tr>
<tr>
<td>ReLU CNRNN (2:2) pool</td>
<td>0.61</td>
<td>0.58</td>
<td>0.55</td>
<td>0.44</td>
<td>0.42</td>
<td>0.62</td>
<td>0.60</td>
<td>0.53</td>
<td>0.84</td>
<td>0.43</td>
<td>0.79</td>
<td>0.64</td>
<td>0.28</td>
<td>0.40</td>
<td>0.48</td>
<td>0.53</td>
</tr>
<tr>
<td>ReLU Convolution (with -operator)</td>
<td>0.63</td>
<td>0.57</td>
<td>0.60</td>
<td>0.44</td>
<td>0.43</td>
<td>0.59</td>
<td>0.52</td>
<td>0.59</td>
<td>0.84</td>
<td>0.55</td>
<td>0.81</td>
<td>0.64</td>
<td>0.30</td>
<td>0.45</td>
<td>0.49</td>
<td>0.57</td>
</tr>
</tbody>
</table>

In summary, we present a set of simple time-aggregation operators that provide significant improvements on an audio detection task where we surpass the previously best-performing system to set a new state-of-the-art. We provide a detailed analysis of the performance of our systems across different audio events, which gives us further insights about which events are easier to predict and when we can expect degradation in performance. For future work, we hope to scale our model so that it works efficiently with a larger set of classes from the Audioset corpus [4]. We also intend to explore better strategies to automatically learn good thresholds for each class rather than manually determine class-specific thresholds.

5. References


