EMI-RNN: Multiple Instance Learning for Efficient Sequential Data Classification on Resource Constrained Devices

Don Kurian Dennis, Chirag Pabbaraju, Harsha Vardhan Simhadri, Proteek Jain

Recurrent Neural Networks (RNN)

- RNNs are state-of-the-art for time series modelling.
- Data is divided into overlapping windows and all RNNs are run over each window.
- Each RNN run is a sequence of updates to its internal state.
- The state update rule, complicated, non-linear and expensive.

$$h_t = \sigma(wx_t + uh_{t-1} + b)$$

- Prohibitively expensive for edge devices.
- Key-word spotting: Feature computation + prediction every 30ms for real-time response! Vanilla LSTM takes 84ms!

Typical class signature length $k << T$

- Example: Keyword spotting — keyword ‘Up’ usually just lasts a 100-200ms while the RNN window T is proportional to 1 second.
- Learning on these signatures can help learn a $k \ll T$ step RNN.
- In the class signature, common prefixes are small making it possible to predict early, without consuming expensive data.
- Class signature can lie anywhere!

Conclusions

- EMI-RNN: exploits temporal structure and abstract observations.
- USP: a) higher accuracy than baseline RNN architectures
  b) reduces inference time by as much as 10x
  c) Allows deployment on tiny devices like Raspberry Pi, M4 MCU
- Techniques: multi-instance learning (MIL) + early prediction
- Analysis: recovers provably optimal solution in non-homogeneous MIL settings — first such result for non-homogeneous MIL

Contributions

- EMI-RNN: Multiple Instance Learning for Efficient Sequential Data Classification on Resource Constrained Devices

MI-RNN: Multiple-Instance RNN

- Divide into bag of overlapping length windows (instances).
- Isolate the instances with signature. Relabel these instances and train.
- NP-Hard in general.

Exploit temporal locality and approximate signature length with MIL/Robust learning techniques in the optimization problem.

Formulation learns model as well as setting index $s_i$ of the class signature in each data point

$$\min_{f, T} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{s_i \in \mathbb{S}_t} k \delta_{s_i}(y_{s_i(t)}, y_{s_i(t)})$$

Algorithm

Step 1: Assign label $(z_{i,s_i}, t_{i,s_i}, i_{s_i}, s_i, y_{s_i, T})$
Step 2: Train classifier on this miss-labelled data $f_t$
Step 3: Score $(s_i, \tilde{z}_{i,s_i})$ and pick Type equation here.
Step 4: Update labels.

Theorem: In $O(T \log n)$ iterations, the true positive set will be recovered exactly, with high prob.

Setting:

- Two classes: $z_{i,s_i}^+$ — negative class instances sampled from a Gaussian with mean $\mu$.
- $z_{i,s_i}^-$ — positive class instances, lie in a small ball around $\mu^+$
- $\|\mu^+ - \mu^-\| \geq C \log T$
- Let $n \geq \frac{\log T \delta^2}{\delta^2 C^2}$

EMI-RNN: Early Multi-Instance RNN

- Naive early prediction inefficient due to common prefixes
- MI-RNN removes common prefixes making early prediction effective

$$L_{e}(X, h) = \sum_{i=1}^{T} (h_{i} - y)^2$$

Accuracy graph

Results

<table>
<thead>
<tr>
<th>Device</th>
<th>Hidden LSTM</th>
<th>MI-RNN</th>
<th>EMI-RNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>GesturePod-6</td>
<td>16</td>
<td>32.2</td>
<td>36.0</td>
</tr>
<tr>
<td>WakeWord-2</td>
<td>16</td>
<td>32.2</td>
<td>36.0</td>
</tr>
<tr>
<td>Google-13 Audio</td>
<td>32</td>
<td>32.2</td>
<td>36.0</td>
</tr>
<tr>
<td>Google-13 Gesture</td>
<td>64</td>
<td>32.2</td>
<td>36.0</td>
</tr>
</tbody>
</table>

Prediction time on Raspberry Pi 3 and Pi Zero

<table>
<thead>
<tr>
<th>Device</th>
<th>Hidden LSTM</th>
<th>MI-RNN</th>
<th>EMI-RNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>GesturePod-6</td>
<td>16</td>
<td>32.2</td>
<td>36.0</td>
</tr>
<tr>
<td>WakeWord-2</td>
<td>16</td>
<td>32.2</td>
<td>36.0</td>
</tr>
<tr>
<td>Google-13 Audio</td>
<td>32</td>
<td>32.2</td>
<td>36.0</td>
</tr>
<tr>
<td>Google-13 Gesture</td>
<td>64</td>
<td>32.2</td>
<td>36.0</td>
</tr>
</tbody>
</table>