MULTI-DECODER DPRNN: SOURCE SEPARATION FOR VARIABLE NUMBER OF SPEAKERS

Junzhe Zhu, Raymond A. Yeh, Mark Hasegawa-Johnson
Separation with variable number of speakers

- Source separation: Given an audio mixture $X = \sum S_1, S_2, S_3, \ldots$, find $S_1, S_2, S_3$
- Dataset: a possible range of number of speakers (we set it to 2-5)
- Challenge: How do we use a single neural network to output a variable-sized tensor containing the estimated source signals?
Challenges of Existing methods

- OR-PIT: Separate one source at a time, combine with permeation-invariant training (PIT)
- Nachmani et al. proposed to train a model for each dataset with a fixed number of speakers.
- Luo et al. proposed to use a model with a high number of output channels, and discard extra channels
- The first two methods perform better, but runtime scale up with the number of sources. The third method has low SNR when number of sources goes up
Problem formulation

- We divide the problem into two parts - 1. Source counting (determining number of speakers) 2. Given the source count, estimate the source signals
- We assume that if we use a sequence neural network (e.g. LSTM) to process a mixture signal, the output distribution will change depending on the number of ground-truth sources.
- Question: how do we make a single neural network adapt to a variable number of sources
Instead of having one decoder, we used a number of decoders, each with a different num_spks.
Our solution

• Use the same encoder & backbone network, but a different decoder for each number of sources!
Our solution

• We always use the same encoder & backbone, shared by all decoders
• Each decoder takes the same input, but has a different number of output channels (achieved by a different projection layer)
• During training, we select the decoder based on the ground-truth
• We also train a classifier that selects which decoder to use during inference
Implementation

- The number of output channels(sources) scales up with output size of projection layer
- The count-head(which selects decoder) is trained, but not used during training

\[
[256, 3999] -\rightarrow [512, 3999] \\
[256, 3999] -\rightarrow [768, 3999] \\
\]
Training

- Two components: decoder(signal loss) and count-head(classification loss)
- Training steps:
  1. Input mixture signal M
  2. Run encoder & LSTM backbone, compute intermediate output Z
  3. Run count-head with Z as input, compute probability P
  4. Run a decoder(selected based on ground truth) with Z as input, compute estimated sources S
  5. Loss is weighted sum of cross entropy loss for P and reconstruction loss for S

\[
\mathcal{L}_{\text{count}}(\mathbf{x}, \mathcal{Y}) = - \sum_{k}^{K} \mathbf{1}_{|\mathcal{Y}| = k} \cdot \log \hat{p}(|\mathcal{Y}| = k | \mathbf{x}), \quad \mathcal{L}_{\text{decoders}}(\mathbf{x}, \mathcal{Y}) = \sum_{k}^{K} \mathbf{1}_{|\mathcal{Y}| = k} \cdot \text{uPIT}(\mathcal{Y}, \hat{\mathcal{Y}}_k),
\]

Final Loss: \[ \min_{\theta} \sum_{(\mathbf{x}, \mathcal{Y}) \in \mathcal{D}} \alpha \cdot \mathcal{L}_{\text{count}}(\mathbf{x}, \mathcal{Y}) + (1 - \alpha) \cdot \mathcal{L}_{\text{decoders}}(\mathbf{x}, \mathcal{Y}). \]
Inference

- Two components: decoder (signal loss) and count-head (classification loss)
- Inference steps:
  1. Input mixture signal M
  2. Run encoder & LSTM backbone, compute intermediate output Z
  3. Run count-head with Z as input, compute probability P
  4. Run the decoder selected based on P to get estimated signals $S = \{S_1, S_2, \ldots\}$
Performance Metric

- We define penalized-SNR, which has two components:
  1. Si-SNR for the matched part between ground truth (GT) and estimated sources
  2. Penalty term = \{Constant P\_ref\} \times \{number of mismatched channels\}

- We choose P\_ref to either be -30 or the average SNR the system would achieve when provided with oracle number of speakers

\[
P\text{-Si-SNR} = \frac{1}{|\mathcal{D}|} \sum_{(x, y) \in \mathcal{D}} \frac{1}{\max(|y|, |\hat{y}|)} (\mathcal{L}_{\text{match}} + \mathcal{L}_{\text{pad}}),
\]

\[
\mathcal{L}_{\text{match}} = \max_{\pi} \sum_{n=1}^{\min(|y|, |\hat{y}|)} \text{SI-SNR}(y^{\pi(n)}, \hat{y}^n) \text{ Signal-to-noise ratio}
\]

where

\[
\mathcal{L}_{\text{pad}} = P_{\text{ref}} \cdot \left| |y| - |\hat{y}| \right|. \text{ (penalty term)}
\]
The count-head performs with high accuracy (98.5%)
Result: Source counting accuracy & oracle SNR

<table>
<thead>
<tr>
<th>Model</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model-Select(DPRNN)[9]</td>
<td>81.3</td>
<td>64.4</td>
<td>46.2</td>
<td>85.6</td>
</tr>
<tr>
<td>Model-Select(Mulcat)[9]</td>
<td>84.6</td>
<td>69.0</td>
<td>47.5</td>
<td>92.3</td>
</tr>
<tr>
<td>Attractor Network[12]</td>
<td>95.7</td>
<td>97.6</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR-PIT[10]</td>
<td>95.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Ours</td>
<td>99.9</td>
<td>99.2</td>
<td>97.6</td>
<td>97.3</td>
</tr>
</tbody>
</table>

Table 1. Performance of source counting; Each column is recall for corresponding number of speakers. For OR-PIT, only overall accuracy is provided.

<table>
<thead>
<tr>
<th>Model</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv-Tasnet[6]</td>
<td>15.3</td>
<td>12.7</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DPRNN[7]</td>
<td>18.8</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Attractor Network[12]</td>
<td>15.3</td>
<td>14.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR-PIT[10]</td>
<td>14.8</td>
<td>12.6</td>
<td>10.2</td>
<td>-</td>
</tr>
<tr>
<td>Ours</td>
<td>19.1</td>
<td>14.1</td>
<td>9.3</td>
<td>5.9</td>
</tr>
</tbody>
</table>

Table 2. Oracle SNR; Each column shows results averaged from all mixtures with corresponding number of speakers. *models above double-line are models with fixed number of speakers.

2.4. Inference

- Demo can be found at [https://junzhejosephzhu.github.io/Multi-Decoder-DPRNN/](https://junzhejosephzhu.github.io/Multi-Decoder-DPRNN/)
- Open-sourced implementation and pre-trained lightweight model can be found in asteroid toolkit
- Full model can be found at [https://github.com/JunzheJosephZhu/MultiDecoder-DPRNN](https://github.com/JunzheJosephZhu/MultiDecoder-DPRNN)
Result: P-Si-SNR

- Model-Select indicates the method where a different model is trained for each number of speakers.
- As can be seen, our method performs similarly well, but requires much less training resources, and is faster during inference.

<table>
<thead>
<tr>
<th></th>
<th>$P_{\text{ref}} = -30dB$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model-Select(DPRNN)[10]*</td>
<td>15.2</td>
<td>10.7</td>
<td>6.0</td>
<td>7.7</td>
<td></td>
</tr>
<tr>
<td>Model-Select(Mulcat)[10]*</td>
<td>17.5</td>
<td>13.21</td>
<td>8.4</td>
<td>10.0</td>
<td></td>
</tr>
<tr>
<td>Attractor Network[14]</td>
<td>14.7</td>
<td>14.2</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>OR-PIT[11]</td>
<td>13.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td>19.1</td>
<td>14.0</td>
<td>9.2</td>
<td>5.8</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$P_{\text{ref}} = -SI\text{-SNR}_{\text{oracle}}$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model-Select(DPRNN)[10]*</td>
<td>15.9</td>
<td>12.1</td>
<td>8.1</td>
<td>8.2</td>
<td></td>
</tr>
<tr>
<td>Model-Select(Mulcat)[10]*</td>
<td>18.1</td>
<td>14.2</td>
<td>10.2</td>
<td>10.3</td>
<td></td>
</tr>
<tr>
<td>Attractor Network[14]</td>
<td>14.9</td>
<td>14.3</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>OR-PIT[11]</td>
<td>13.4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td>19.1</td>
<td>14.0</td>
<td>9.3</td>
<td>5.9</td>
<td></td>
</tr>
</tbody>
</table>

Table 4. P-SI-SNR of each model; For OR-PIT, result is computed by averaging the P-SI-SNR for both 2 and 3 speakers computed with 95.7% recall. Note that models with lower max speaker count generally have higher accuracy, since fewer classes implies a higher P-SI-SNR. * denotes models trained on fixed number of speakers.
Summary

• A method for source separation w/ unknown number of sources
• 2 Problems:
  1. how many sources? Solution: Use shared backbone & classifier(count-head)
  2. How to recover signal? Solution Use multiple decoder heads
• Result: High classification accuracy, no increase in complexity