Self-Supervised Learning of Face Representations for Video Face Clustering

Vivek Sharma, Makarand Tapaswi, Saquib Sarfraz, and Rainer Stiefelhagen

https://github.com/vivoutlaw/SSIAM
Motivation

• To learn discriminative face representation via self-supervision
  • Small intra-person-distance and large inter-person-distance.

• This will benefit potential applications in
  • Video understanding, video summarization, content-based indexing & retrieval
  • Automatic reasoning about multimedia content.
Introduction

- Video face clustering is hard.
  - Discriminative features help.

- Most prior works utilize: must-link and cannot-link information.
• Difficult to train from scratch (require lots of training data), typically handled by net surgery:
  • Fine-tuning
  • Use of additional embedding's on the features from the last layer
  • Both

• We propose two self-supervised discriminative methods.
  • Self-supervised Siamese network (SSiam)
  • Track-supervised Siamese network (TSiam)

• We evaluate on three video face clustering datasets.
Temporal Constraints

- Video constraints: must-link and cannot-not link.

Related Work

- Link-constrained based improved triplet loss

Related Work

- Based on loss function or MRF modeling.

Related Work: Pseudo-RF

- This is especially in light of CNN face representations that are very similar even across different identities.
- We see a large overlap between the cosine similarity distributions of positive (same id) and negative (across id) track pairs.
Self-supervised Siamese network (SSiam)

• Does not need tracks or temporal information.
• Mechanism for mining positive and negative examples automatically.
• Compute a distance matrix (i.e. ranking) over random subset per iteration
  • Use the farthest positives and closest negatives pairs sets as labels.
### Distance Matrix

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0.1</td>
<td>0.5</td>
<td>0.7</td>
</tr>
<tr>
<td>2</td>
<td>0.1</td>
<td>0</td>
<td>0.9</td>
<td>0.4</td>
</tr>
<tr>
<td>3</td>
<td>0.5</td>
<td>0.9</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>4</td>
<td>0.7</td>
<td>0.4</td>
<td>0.3</td>
<td>0</td>
</tr>
</tbody>
</table>

### Pairs

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-1</td>
<td>1-2</td>
<td>1-3</td>
<td>1-4</td>
</tr>
<tr>
<td>2</td>
<td>2-1</td>
<td>2-2</td>
<td>2-3</td>
<td>2-4</td>
</tr>
<tr>
<td>3</td>
<td>3-1</td>
<td>3-2</td>
<td>3-3</td>
<td>3-4</td>
</tr>
<tr>
<td>4</td>
<td>4-1</td>
<td>4-2</td>
<td>4-3</td>
<td>4-4</td>
</tr>
</tbody>
</table>

**Sort distance row-wise**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0.1</td>
<td>0.5</td>
<td>0.7</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0.1</td>
<td>0.4</td>
<td>0.9</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0.3</td>
<td>0.5</td>
<td>0.9</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0.3</td>
<td>0.4</td>
<td>0.7</td>
</tr>
</tbody>
</table>

- Choose positive pairs from the second column with the largest distance, and negative pairs from the last column with the smallest distance.
- These pairs are semi-hard.
- Example:
  - 1 positive (3-4) and 1 negative pair (1-4)
SSiam selects hard pairs: farthest positives and closest negatives using a ranked list based on distance matrix. $B$ corresponds to batch.
Track-supervised Siamese networks (TSiam)

- Use temporal information (must-link/cannot-link).
- Also include negative pairs for singleton tracks
  - based on track-level distances (computed on base features)
  - randomly sample frames from the farthest $F = 25$ tracks.

Face track with $M$ frames

- Pos. Pair $y = 0$
- Neg. Pair $y = 1$

$B$

$\begin{align*}
x_1 \\ x_+ \\ x_-
\end{align*}$

Contrastive Loss

same

$\begin{array}{c}
\text{CNN Feature Maps} \\
\text{MLP}
\end{array}$

$\begin{array}{c}
\text{same} \\
0/1
\end{array}$
Evaluation

• We present our evaluation on three challenging datasets.
  • Buffy the Vampire Slayer (BF) (season 5, episodes 1 to 6)
  • Big Bang Theory (BBT) (season 1, episodes 1 to 6)
  • Harry Potter 1 Movie (ACCIO)

<table>
<thead>
<tr>
<th>Datasets</th>
<th>#Cast</th>
<th>This work</th>
<th>Previous work</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>#TR (#FR)</td>
<td>#TR (#FR)</td>
</tr>
<tr>
<td>BBT0101</td>
<td>5</td>
<td>644 (41220)</td>
<td>182 (11525)</td>
</tr>
<tr>
<td>BF0502</td>
<td>6</td>
<td>568 (39263)</td>
<td>229 (17337)</td>
</tr>
<tr>
<td>ACCIO</td>
<td>36</td>
<td>3243 (166885)</td>
<td>3243 (166885)</td>
</tr>
</tbody>
</table>

• Metrics
  • Clustering acc. for BBT, BF
  • BCubed, P, R, F1 for ACCIO
Implementation details

• We extract VGGFace2 features. The features are of 2048 Dimensions.

• Siamese network. Fully-connected neural network (2048 $\rightarrow$ 512 $\rightarrow$ 2). We extract the feature representations of 512D for clustering.
SSiam and TSiam labels mining

• For SSiam,
  • We use a random subset of size $B = 3000$
  • Choose $2K$: positive and negative pairs, $K = 64$.
  • Higher values of $B$ did not improve.

• For TSiam, we mine 2 positive and 4 negative pairs for each frame.
Testing Setup

- Extract features from base network and trained MLP: SSiam or TSiam.
- Perform clustering via HAC
TSiam, impact of singleton tracks

- Ignoring singleton tracks leads to significant performance drop.
- Approx. 50-70% tracks are singleton and ignoring them lowers accuracy by 4%.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>TSiam w/o Single (FG_Best)</th>
<th>TSiam Ours</th>
<th># Tracks</th>
<th># Single</th>
<th># Co-oc</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBT-0101</td>
<td>0.936</td>
<td>0.964</td>
<td>644</td>
<td>331</td>
<td>313</td>
</tr>
<tr>
<td>BF-0502</td>
<td>0.849</td>
<td>0.893</td>
<td>568</td>
<td>395</td>
<td>173</td>
</tr>
</tbody>
</table>

FG_Best: Datta et al.: Unsupervised learning of face representations. In FG. IEEE, 2018
SSiam, comparison to pseudo-RF

• In Pseudo-RF, all samples are treated independent of each other.
• A pair of samples closest in distance are chosen as positive, and farthest as negative.
• SSiam that involves sorting a batch of queries is much more efficient over pseudo-RF

<table>
<thead>
<tr>
<th>Method</th>
<th>BBT-0101</th>
<th>BF-0502</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pseudo-RF</td>
<td>0.930</td>
<td>0.814</td>
</tr>
<tr>
<td>SSiam</td>
<td><strong>0.962</strong></td>
<td><strong>0.909</strong></td>
</tr>
</tbody>
</table>
Performance on training videos.

- Training is done at frame-level information.
- Testing is done at track-level i.e. mean representation.

<table>
<thead>
<tr>
<th>Train/Test</th>
<th>Base</th>
<th>TSiam</th>
<th>SSiam</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBT-0101</td>
<td>0.932</td>
<td><strong>0.964</strong></td>
<td>0.962</td>
</tr>
<tr>
<td>BF-0502</td>
<td>0.836</td>
<td>0.893</td>
<td><strong>0.909</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methods</th>
<th>P</th>
<th>R</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>JFAC (ECCV ’16)</td>
<td>0.690</td>
<td>0.350</td>
<td>0.460</td>
</tr>
<tr>
<td>Ours (with HAC)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSiam</td>
<td>0.749</td>
<td>0.382</td>
<td>0.506</td>
</tr>
<tr>
<td>SSiam</td>
<td>0.766</td>
<td>0.386</td>
<td>0.514</td>
</tr>
</tbody>
</table>

#cluster=36
Comparison with the SOTA at Frame-Level

<table>
<thead>
<tr>
<th>Method</th>
<th>BBT-0101</th>
<th>BF-0502</th>
</tr>
</thead>
<tbody>
<tr>
<td>ULDML (ICCV ’11)</td>
<td>57.00</td>
<td>41.62</td>
</tr>
<tr>
<td>HMRF (CVPR ’13)</td>
<td>59.61</td>
<td>50.30</td>
</tr>
<tr>
<td>HMRF2 (ICCV ’13)</td>
<td>66.77</td>
<td>-</td>
</tr>
<tr>
<td>WBSLRR (ECCV ’14)</td>
<td>72.00</td>
<td>62.76</td>
</tr>
<tr>
<td>VDF (CVPR ’17)</td>
<td>89.62</td>
<td>87.46</td>
</tr>
<tr>
<td>Imp-Triplet (PacRim ’16)</td>
<td>96.00</td>
<td>-</td>
</tr>
<tr>
<td>JFAC (ECCV ’16)</td>
<td>-</td>
<td>92.13</td>
</tr>
<tr>
<td>Ours (with HAC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSiam</td>
<td>98.58</td>
<td>92.46</td>
</tr>
<tr>
<td>SSiam</td>
<td>99.04</td>
<td>90.87</td>
</tr>
</tbody>
</table>

- Training the SSiam for about 15 epochs on BBT-0101 requires less than 25 minutes.

## Comparison with SOTA on ACCIO

<table>
<thead>
<tr>
<th>Methods</th>
<th># clusters=40</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
</tr>
<tr>
<td>K-means-DeepID2&lt;sup&gt;+&lt;/sup&gt; (ECCV ’16)</td>
<td>0.543</td>
</tr>
<tr>
<td>DIFFRAC-DeepID2&lt;sup&gt;+&lt;/sup&gt; (ICCV ’11)</td>
<td>0.557</td>
</tr>
<tr>
<td>WBSLRR-DeepID2&lt;sup&gt;+&lt;/sup&gt; (ECCV ’14)</td>
<td>0.502</td>
</tr>
<tr>
<td>HMRF-DeepID2&lt;sup&gt;+&lt;/sup&gt; (CVPR ’13)</td>
<td>0.599</td>
</tr>
<tr>
<td>DeepID2&lt;sup&gt;+&lt;/sup&gt;.C0-Intra (ECCV ’16)</td>
<td>0.657</td>
</tr>
<tr>
<td>JFAC (ECCV ’16)</td>
<td>0.711</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ours (with HAC)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TSiam</td>
<td>0.763</td>
</tr>
<tr>
<td>SSiam</td>
<td>0.777</td>
</tr>
</tbody>
</table>
Conclusion

• Presented two variants of discriminative methods to learn strong face representations
  • Self-supervised Siamese network (SSiam)
  • Track-supervised Siamese network (TSiam)
• State-of-the-art representation learning approach on BBT, BF and ACCIO.
Thank you!

https://vivoutlaw.github.io/  sharma.vivek@live.in