Feature Re-Learning with Data Augmentation for Content-based Video Recommendation

Jianfeng Dong¹, Xirong Li², Chaoxi Xu², Gang Yang², Xun Wang¹

1. Zhejiang Gongshang University
²AI & Media Computing Lab, Renmin University of China

Grand Challenge Session @ ACM Multimedia 2018
Videos are important

Video-sharing websites are very popular.

On YouTube:

• 300 hours of video are uploaded every minute
• 5 billion videos are watched per day
• 30 million user visited YouTube per day
• 2.1 hours consumed by visitors per day per person
Video recommendation

In a rich context

• User interaction: browsing, commenting and rating
• Meta-data: title, filename
• ...

YOU MAY ALSO LIKE

- Harlots
- National Treasure
- Designated Survivor
- The Path
- Chance
- This Is Us
Cold-start video recommendation

- No contextual information
- Video content only
**Hulu task**

Content-based Video Relevance Prediction Challenge

Given a video, participants are asked to rank a list of pre-specified videos in terms of their relevance.

![Diagram showing the Hulu task process](image-url)

- **Given video**: BROOKLYN NINE-NINE
- **Candidate videos**: SOUTH PARK, ... (multiple candidates)
- **Recommend videos**: High relevance (example videos), Low relevance (example videos)

The diagram illustrates the process of ranking candidate videos based on their relevance to the given video.
Task setup

What we have

- Two tracks:
  - Movies Track
  - TV-shows Track
- Video relevance list
- Visual features
  - frame-level feature: Inception-v3
  - video-level feature: C3D

What we do not have

- Videos
- Frames
- Contextual information
  - user interaction
  - meta-data
  - ...

Impossible to visually examine recommendation results
**Challenge one**

Limited training data.

<table>
<thead>
<tr>
<th></th>
<th>train</th>
<th>validation</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movies Track</td>
<td>4500</td>
<td>1188</td>
<td>4500</td>
</tr>
<tr>
<td>TV-shows Track</td>
<td>3000</td>
<td>864</td>
<td>3000</td>
</tr>
</tbody>
</table>
Challenge two

Off-the-shelf CNN features are not optimal.

inception-v3

C3D
Our solution

**Challenge one**

Limited training data.

**Challenge two**

CNN features are not optimal.

**Late fusion**

Data augmentation

Feature relearning

Our solution
Augmentation for frame-level features

Inspired by the fact that humans could grasp the video topic after watching only several sampled video frames in order, we augment data by skip sampling.

![Diagram showing skip sampling and mean pooling to augment data for frame-level features]
Augmentation for video-level features

As adding tiny perturbations to image pixels are imperceptible to humans, we introduce perturbation-based data augmentation.
Feature re-learning

Original feature space

Re-learned feature space

Triplet ranking loss:
\[ L(v, v^+, v^-; W, b) = \max(0, \alpha - cs_\phi(v, v^+) + cs_\phi(v, v^-)) \]
Augmentation and re-learning

Both data augmentation and feature re-learning is effective.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Augmentation</th>
<th>Re-Learning</th>
<th>Movies</th>
<th>TV-shows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception-v3</td>
<td>×</td>
<td>×</td>
<td>0.099</td>
<td>0.124</td>
</tr>
<tr>
<td></td>
<td>×</td>
<td>√</td>
<td>0.163</td>
<td>0.199</td>
</tr>
<tr>
<td></td>
<td>√</td>
<td>√</td>
<td>0.191</td>
<td>0.244</td>
</tr>
<tr>
<td>C3D</td>
<td>×</td>
<td>×</td>
<td>0.112</td>
<td>0.145</td>
</tr>
<tr>
<td></td>
<td>×</td>
<td>√</td>
<td>0.155</td>
<td>0.185</td>
</tr>
<tr>
<td></td>
<td>√</td>
<td>√</td>
<td>0.163</td>
<td>0.196</td>
</tr>
</tbody>
</table>
Choice of loss functions

Triplet ranking loss consistently outperforms the other two loss functions on both two tracks.

<table>
<thead>
<tr>
<th>Loss</th>
<th>Movies</th>
<th>TV-shows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triplet ranking loss</td>
<td>0.163</td>
<td>0.199</td>
</tr>
<tr>
<td>Improved Triplet ranking loss</td>
<td>0.125</td>
<td>0.181</td>
</tr>
<tr>
<td>Contrastive loss</td>
<td>0.160</td>
<td>0.194</td>
</tr>
</tbody>
</table>

Late fusion

Late fusion is employed by averaging the relevance given by multiple models, which further boosts the performance.

<table>
<thead>
<tr>
<th>Late fusion</th>
<th>Movies</th>
<th>TV-shows</th>
</tr>
</thead>
<tbody>
<tr>
<td>×</td>
<td>0.191</td>
<td>0.244</td>
</tr>
<tr>
<td>√</td>
<td>0.211</td>
<td>0.276</td>
</tr>
</tbody>
</table>
Official evaluation

Our runs are ranked first on Movies Track and second on TV-shows Track.
Take-home messages

Good practices

• data augmentation on features generating more training instances
• feature re-learning with the triplet ranking loss
• late fusion of multiple models

https://github.com/danieljf24/cbvr
## Our runs

### Track 1: TV-shows

<table>
<thead>
<tr>
<th></th>
<th>hit@k</th>
<th></th>
<th></th>
<th>recall@k</th>
<th></th>
<th></th>
<th>recall@k</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>k=5</td>
<td>k=10</td>
<td>k=20</td>
<td>k=30</td>
<td>k=50</td>
<td>k=100</td>
<td>k=200</td>
</tr>
<tr>
<td>Hulu</td>
<td>0.249</td>
<td>0.356</td>
<td>0.461</td>
<td>0.525</td>
<td>0.085</td>
<td>0.141</td>
<td>0.219</td>
</tr>
<tr>
<td>run 1</td>
<td>0.274</td>
<td>0.365</td>
<td>0.488</td>
<td>0.542</td>
<td>0.099</td>
<td>0.160</td>
<td>0.248</td>
</tr>
<tr>
<td>run 2</td>
<td>0.287</td>
<td>0.381</td>
<td>0.492</td>
<td>0.550</td>
<td>0.104</td>
<td>0.167</td>
<td>0.257</td>
</tr>
<tr>
<td>run 3</td>
<td>0.288</td>
<td>0.391</td>
<td>0.484</td>
<td>0.539</td>
<td>0.099</td>
<td>0.162</td>
<td>0.249</td>
</tr>
<tr>
<td>run 4</td>
<td>0.309</td>
<td>0.411</td>
<td>0.506</td>
<td>0.567</td>
<td>0.109</td>
<td>0.173</td>
<td>0.266</td>
</tr>
<tr>
<td>run 5</td>
<td>0.308</td>
<td>0.408</td>
<td>0.522</td>
<td>0.589</td>
<td>0.112</td>
<td>0.178</td>
<td>0.273</td>
</tr>
</tbody>
</table>

### Track 2: Movies

<table>
<thead>
<tr>
<th></th>
<th>hit@k</th>
<th></th>
<th></th>
<th>recall@k</th>
<th></th>
<th></th>
<th>recall@k</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>k=5</td>
<td>k=10</td>
<td>k=20</td>
<td>k=30</td>
<td>k=50</td>
<td>k=100</td>
<td>k=200</td>
</tr>
<tr>
<td>Hulu</td>
<td>0.190</td>
<td>0.242</td>
<td>0.320</td>
<td>0.373</td>
<td>0.081</td>
<td>0.116</td>
<td>0.168</td>
</tr>
<tr>
<td>run 1</td>
<td>0.210</td>
<td>0.272</td>
<td>0.355</td>
<td>0.412</td>
<td>0.092</td>
<td>0.133</td>
<td>0.192</td>
</tr>
<tr>
<td>run 2</td>
<td>0.211</td>
<td>0.278</td>
<td>0.368</td>
<td>0.427</td>
<td>0.095</td>
<td>0.139</td>
<td>0.201</td>
</tr>
<tr>
<td>run 3</td>
<td>0.215</td>
<td>0.278</td>
<td>0.359</td>
<td>0.422</td>
<td>0.096</td>
<td>0.138</td>
<td>0.198</td>
</tr>
<tr>
<td>run 4</td>
<td>0.234</td>
<td>0.298</td>
<td>0.390</td>
<td>0.448</td>
<td>0.104</td>
<td>0.148</td>
<td>0.210</td>
</tr>
<tr>
<td>run 5</td>
<td>0.232</td>
<td>0.302</td>
<td>0.389</td>
<td>0.441</td>
<td>0.105</td>
<td>0.151</td>
<td>0.215</td>
</tr>
</tbody>
</table>