Crowdsourced Video Annotation

AshokVydyanathan
ECE 285 (WI 2013)
Efficiently Scaling Up Crowdsourced Video Annotation

Carl Vondrick  Donald Patterson  Deva Ramanan

The result of a three year study into crowdsourcing video annotation tasks, including framework and interface development and user studies, and their results.

End result was VATIC, a framework for crowdsourcing the annotation of video data through Amazon’s Mechanical Turk system.

This paper focuses on interface development linked to user studies.
What is annotation? Why crowdsourcing it?

In this context, annotation is the creation of labeled datasets that can be used as ground-truth data for testing different vision-based systems.

Annotation of large datasets can be tedious, taking as much (or perhaps even more) time to generate a ground-truth dataset as to actually develop the system to be tested.

If annotation is done by a small group of researchers on a large dataset, many man-hours can be spent on this task.

If instead, a framework is developed for allowing users from all over the world to contribute towards the generation of a labeled dataset, time would be saved.
Prior work

Static image dataset annotation has already been demonstrably crowdsourced on a large scale (ImageNet, LabelMe) on platforms such as Amazon’s Mechanical Turk.

Significant progress has been made into the development of specialized tools for video annotation (LabelMe, ViPER, FlowBoost). This brings a new set of challenges, due to the introduction of time as a dimension in the dataset. Frame-to-frame similarity increases the tedium of the process, but can be exploited by interpolation/tracking.

However, these tools are geared towards the creation of high quality labels, but are not necessarily economical. These tools are complex to use, and require expert use, again limiting the user base to researchers or trained annotators.
VATIC development

Since annotation will be crowdsourced, high quality tools aimed at experts are too complex, take too long.

Studies were carried out to develop a user interface that was easy to use and quick to deliver labeled datasets.
User interface

Very important aspect of VATIC
  If it’s too complex, most users will not complete task
  If it’s too simple, might lose some functionality

Basic issues to tackle
  Key-frame schedule
  Multiple object annotation
  Maintaining track identity
  Object attributes
VATIC’s user interface
Datasets used for testing

3 datasets

**Scripted**: A group of people quickly walking around in a complex manner

**Basketball**: Clip from a basketball game, with ambiguous motion, non-linear paths

**VIRAT**: Parking lot with a few cars driving in linear paths
**Key-frame schedule**

Manually labeling each frame in a video is time-consuming and inefficient. Usually (but not always), Temporal and spatial dependency of object positions can be used.

**Key-frames** can be used at intervals in the video to define object position, and intermediate positions can be interpolated/tracked using these positions.

Two ways to define key-frame positions:

- **Fixed key-frames**: The user labels every $T$ frames. For a particular video clip, $T$ is fixed. For complex videos, $T$ must be small in order to fully capture motion.

- **User-defined key-frames**: The user can pause and annotate any frame, which will then be treated as a key-frame. This is flexible and allows skipping of long, relatively invariant parts of video, while focusing on complex, quickly changing sequences.
Key-frame schedule: User-study results

Intuitively, expectation is that user-defined key-frames will provide better results in terms of time taken. In fact, most tools use this key-frame schedule.

Surprisingly, results of study showed that most users are not accurate at estimating optimal key-frame locations, since in-between interpolation can be non-intuitive. Therefore, time is spent deciding where to place key-frames, in addition to annotation.

Researchers noted that in fixed-rate approach, users only had to watch video once. In user-defined approach, users watched the video multiple times to correct interpolation.

*Fixed-rate key-frame approach is faster than user-defined key-frame approach*
Key-frame schedule: User-study results

<table>
<thead>
<tr>
<th>Subject</th>
<th>Scripted</th>
<th></th>
<th></th>
<th></th>
<th>Basketball</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>VIRAT</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>User</td>
<td>Fixed</td>
<td>Ratio</td>
<td>Saved</td>
<td>User</td>
<td>Fixed</td>
<td>Ratio</td>
<td>Saved</td>
<td>User</td>
<td>Fixed</td>
<td>Ratio</td>
<td>Saved</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>599</td>
<td>463</td>
<td>0.77</td>
<td>136</td>
<td>1,457</td>
<td>1,323</td>
<td>0.91</td>
<td>134</td>
<td>220</td>
<td>244</td>
<td>1.11</td>
<td>-24</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>653</td>
<td>247</td>
<td>0.38</td>
<td>406</td>
<td>4,555</td>
<td>2,275</td>
<td>0.50</td>
<td>2,280</td>
<td>176</td>
<td>178</td>
<td>1.03</td>
<td>-2</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>476</td>
<td>275</td>
<td>0.58</td>
<td>201</td>
<td>1,216</td>
<td>830</td>
<td>0.68</td>
<td>386</td>
<td>338</td>
<td>215</td>
<td>0.64</td>
<td>123</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>772</td>
<td>432</td>
<td>0.56</td>
<td>340</td>
<td>1,505</td>
<td>1,497</td>
<td>0.99</td>
<td>8</td>
<td>489</td>
<td>302</td>
<td>0.62</td>
<td>187</td>
<td></td>
</tr>
<tr>
<td>*E</td>
<td>605</td>
<td>371</td>
<td>0.61</td>
<td>234</td>
<td>935</td>
<td>1,501</td>
<td>1.61</td>
<td>-566</td>
<td>269</td>
<td>231</td>
<td>0.85</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>*F</td>
<td>654</td>
<td>472</td>
<td>0.72</td>
<td>182</td>
<td>1,672</td>
<td>1,858</td>
<td>1.11</td>
<td>-186</td>
<td>372</td>
<td>326</td>
<td>0.87</td>
<td>46</td>
<td></td>
</tr>
<tr>
<td>*G</td>
<td>235</td>
<td>193</td>
<td>0.82</td>
<td>42</td>
<td>591</td>
<td>696</td>
<td>1.18</td>
<td>-105</td>
<td>165</td>
<td>120</td>
<td>0.73</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>*H</td>
<td>312</td>
<td>331</td>
<td>1.06</td>
<td>-19</td>
<td>656</td>
<td>748</td>
<td>1.14</td>
<td>-92</td>
<td>172</td>
<td>164</td>
<td>0.95</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>538</td>
<td>348</td>
<td>0.66</td>
<td>190</td>
<td>1,573</td>
<td>1,341</td>
<td>0.96</td>
<td>232</td>
<td>275</td>
<td>223</td>
<td>0.83</td>
<td>53</td>
<td></td>
</tr>
</tbody>
</table>
Multiple-object annotation

A significant burden in video annotation is that possibly hundreds of objects may be moving through a scene at any point, all with possibly independent motions. Since the aim is to get as dense labeling as possible, all these objects have to be labeled eventually.

There are three ways to annotate multiple objects in a scene:

- **All objects**: All objects in a frame are labeled before moving to the next frame
- **Single objects**: One object is annotated at a time till the end, and then user rewinds to the starting point for the next object
- **Groups of objects**: Groups of objects, linked semantically, are labeled together at a time
Multiple-object annotation
User-study results

Two datasets were used for this study, Human Joints and Basketball.

Study showed that initially most users adopt the all-object approach, which ended up being time-consuming. Some would resort to labeling objects in groups, which sped things up, but very few users would resort to single-object labeling.

The results of the study showed that annotating a single object at a time was more efficient, resulting in users spending less time annotating the entire clip.

Users were also requested to annotate clips using all three approaches:

- Users gave a strong preference for single-object approach
- Some users refused (or were unable) to complete the study if required to annotate in groups (or all objects at a time)

*Annotating one object at a time is more efficient, and preferred by users.*
Human Joints dataset

Based on **Scripted** dataset, but users now had to label different joints and body parts in the individuals.
Multiple-object annotation
User-study results

<table>
<thead>
<tr>
<th>Subject</th>
<th>Human Joints</th>
<th></th>
<th></th>
<th></th>
<th>Basketball</th>
<th></th>
<th></th>
<th>Preferred</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One</td>
<td>Group</td>
<td>All</td>
<td>One</td>
<td>Group</td>
<td>All</td>
<td>Preferred</td>
<td>Order</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>1639</td>
<td>1.00</td>
<td>1763</td>
<td>1.07</td>
<td>-</td>
<td>-</td>
<td>476</td>
<td>1.00</td>
<td>358</td>
</tr>
<tr>
<td>B</td>
<td>2071</td>
<td>1.00</td>
<td>2100</td>
<td>1.01</td>
<td>2440</td>
<td>1.17</td>
<td>1382</td>
<td>1.00</td>
<td>1618</td>
</tr>
<tr>
<td>C</td>
<td>1867</td>
<td>1.00</td>
<td>2106</td>
<td>1.12</td>
<td>-</td>
<td>-</td>
<td>681</td>
<td>1.00</td>
<td>813</td>
</tr>
<tr>
<td>D</td>
<td>2399</td>
<td>1.00</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>494</td>
<td>1.00</td>
<td>448</td>
</tr>
<tr>
<td>Mean</td>
<td>1994</td>
<td>1.00</td>
<td>1989</td>
<td>1.07</td>
<td>2440</td>
<td>1.17</td>
<td>759</td>
<td>1.00</td>
<td>809</td>
</tr>
</tbody>
</table>

ECE 285 WI 2013
Maintaining track identity across frames

A common mistake among annotators is confusion of object identity between frames. For example, in the Basketball dataset, due to its low resolution and similar objects (players), annotators can misidentify players, leading to incorrect labeling. Also, when annotating multiple objects at a time, tracks can become mixed up.

There are two ways to counter this:

- **Video playback**: Allowing the user to play the video (instead of just viewing key-frames) enables them to track objects correctly.

- **Spacetime tooltip**: To prevent repeated rewinds, VATIC now incorporates a small tool-tip that summarizes an object and its previous labels appears when clicking a bounding box. This allows users to correctly track objects without frequent rewinds.

Users responded enthusiastically to this addition and adopted its use immediately.
Spacetime tool-tip

(a) Which person is the red box tracking? (b) A tooltip reminds the worker. (c) The worker can update the box.
Attributes and visibility

Apart from labeling the position of objects, some binary attributes of the object may also be required, such as whether a person is running or walking, or whether an object is partially occluded. These properties are included as checkboxes when a user is annotating an object. Since defining these attributes in each frame is inefficient, these must be done on key-frames only as well, and interpolated.

But how do you interpolate a binary value?

In the end, the approach was to assume the attribute label is the same as the label in the immediately preceding key-frame in time.
Amazon’s Mechanical Turk

An online labor marketplace that allows employers to easily hire workers to complete tasks. It’s ideal for tasks that are difficult for computers, but trivial for humans.

Employers create tasks (HITs) and set prices for each task. Workers accept tasks and complete them. On validation of completion, the workers are paid.

Two main challenges:
- How do you split up a large video annotation job among multiple workers?
- How do you ensure high-quality results?
Shot-based annotation

This approach involves breaking up a large video into multiple smaller overlapping segments, publish each segment to MTurk as a separate task.

Segment annotations then have to be “stitched” up to make continuous paths that span the entire video. Paths are matched using correspondence in overlapping segments, using a simple cost function:

$$\min_{f} \sum_{i \in S} C(i, f(i)) \quad \text{where} \quad f : S \rightarrow T$$

$$C(i, j) = \sum_{t=0}^{T} \left\{ \begin{array}{ll} 0 & \text{if both are visible and overlap} \\ 0 & \text{if both are not visible} \\ 1 & \text{otherwise} \end{array} \right.$$
Micro vs. Macro tasks

A common principle in crowdsourcing is to make individual tasks as small as possible (micro) so that workers can solve them quickly, improving throughput. In this context, workers would be asked to annotate only a single object in a segment.

The problem encountered with this approach was that sometimes workers would start annotation at a later frame, and other workers would begin this same annotation thinking that it had not been done (since previous annotations are normally visible to others), and then come across the older annotations. Instead of restarting, they would continue, and be influenced by older annotations.

Therefore, the system was changed to make workers label every object in a segment (macro task) in order to improve quality and distribute errors (since each segment will be labelled by only one worker).
Discovering good workers

One crucial aspect of the system is eliminating workers who give bad results. Since this data will be used as ground-truth for evaluation, accuracy is important.

Researchers created a gold-standard video for which ground-truth was already available, and new workers were required to label this first, to check their labeling abilities against the existing labels. Only workers who sufficiently match the ground-truth data are allowed to proceed. Random checking of results were also done to eliminate malicious workers.
Worker compensation

Sufficient compensation is required on MTurk system to attract workers. Since videos with more objects are harder to label, workers were paid per object and per frame, with higher pay for more complex video segments. Completion bonuses were also given to encourage workers to continue.

MTurk users are active on forums, and discuss employers amongst each other. Therefore, maintaining a good reputation is important to ensure that workers continue to do tasks. Therefore, no work was rejected, and all workers who completed tasks were paid regardless of actual output.

Workers were also given the option of their earning being donated to charity in a transparent system, in the hope that this would eliminate malicious workers.
Object interpolation

Since annotation from workers was only for key-frames, interpolation has to be done offline in order to get a complete annotated dataset.

There are various methods of interpolating data for non-key-frames:

- Linear interpolation
- Dynamic programming (tracking based on features within bounding boxes)
Comparison of interpolation methods

Datasets used in order of complexity

(a) Athletic Drills: Trivial due to easily distinguished foreground.

(b) VIRAT Cars: Intermediate due to stationary cameras and little motion

(c) VIRAT People: Difficult due to small size and frequent motion

(d) Basketball Players: Difficult due to frequent occlusions and similar looking objects

(e) Basketball Ball: Extremely difficult due to cluttered backgrounds
Comparison of interpolation methods

Results comparing linear interpolation and dynamic programming methods
Comparison of interpolation methods

CPU vs. Human cost
An interesting conclusion drawn was that different tracking algorithms could be compared simply based on the cost savings.
Comparison of tracking methods

One interesting conclusion that researchers arrived at was an alternative metric for comparing tracking algorithms.

Using Amazon’s EC2 cloud computing platform to measure processing costs, the cost benefit of implementing a tracking method in VATIC (due to balance between human MTurk work and EC2 work shifting) can be used as a metric to see the effectiveness of a tracking algorithm.
Future plans for VATIC

Researchers plan to extend VATIC in the future with some improvements:

Adaptive key-frame schedule that can adjust T automatically within a segment by analyzing complexity of the scene and its motion (requires active learning)

Distributing computing tasks for interpolation to user computers (at the moment, doing this through Javascript is inefficient)

Better incentive systems to attract more “good” workers
Questions

1. Why is annotation of image and video data so important?
2. What is the added complexity faced when annotating video (as compared to annotating images)?
3. What is the motivation behind crowdsourcing annotation work?
4. What did VATIC’s team focus on when developing the user interface for VATIC?
5. What are some of the binary attributes VATIC collects in addition to object locations?
6. What challenges did VATIC’s team face when deploying on MTurk?
7. What prevents the distribution of computing the tracking algorithm to user computers?
8. Why does dynamic programming not outperform linear interpolation for the Ball dataset?