EUROGRAPHICS 2022 / R. Chaine and M. H. Kim
(Guest Editors)

Volume 41 (2022), Number 2

Supplemental materials for CAST: Character labeling in Animation using Self-supervision by Tracking

Oron Nir[1,2], Gal Rapoport[1], and Ariel Shamir[1]
[1]Reichman University , Israel, [2]Microsoft Corporation

1. Introduction

This document contains supplemental materials for the *CAST* paper. Visualizing the challenge to generalize across animation styles using semantic learning representations on Figure1 showing characters from our CAST EVALUATION dataset with both dinosaurs, aliens, and tractors produced in CG, Cutout, and other animation technologies.

2. Proposal Clustering

We have analyzed the different alternatives for semantic embedding networks by evaluating their gain to cluster purity. As stated in the paper, the main goal of clustering is to reduce the load of user interaction in naming characters as much as possible. Hence, we prefer cluster purity over other measures. In Figure 4 we compare the performance of SEResNeXt, ResNet18, and ArtMiner for cluster purity. We show that our network based on a SEResNeXt backbone with 40 epochs of refinement converge to the top performance. This analysis does not include the CAST refinement.

3. Unsupervised discovery of character dictionaries

To complete the visual picture of dictionary discovery by our method, on both CAST and SAIL datasets, the 'Southpark' dictionary could be seen in Figure 3. Our method discovers 13 additional characters and considered relevant exemplars. Another example for a dictionary discovery could be seen in Figure 3

4. Character Identification

We include all of the confusion matrices of the seven test-episodes' characters from CAST EVALUATION data-set in Figure 12. To visually illustrate the generalization of our method the character recognition is presented per style on Figures 5, 6, 7, 8, 9, 10, 11.

4.1. Negative examples sampling algorithm analysis

The suggested method $LER()$ in Algorithm 13 is claimed to have a time complexity of $O(n^2)$. The proof is straight forward using the Master method. As illustrated in the pseudo code, the recursion has four splits into the top, bottom, left, and right parts of the frame where each is bounded by approximately one half of the frame size. On each call, the recursion handles the remaining boxes while filtering those who are outside the sub-frame area and cropping those that intersect with it. This operation is done in $O(n)$ time. Thus, the recursive work is bounded by the following function: $T(n) = 4T(n/2) + O(n)$ which fits the first case of the Master method: $n^{\log_b a}$ where $a = 4; b = 2; d = 1$; which puts us in a complexity of $O(n^2)$. QED. See a visualization of the resulting output on Figure 13.

Algorithm 1 LER($frame, boxes, min\_size$)

Input: Characters bboxes in a given frame character boxes
Output: Large enough empty rectangles empty boxes
Initialisation : boxes are consolidated and validated

```
if ¬frame.is_valid(min_size) then
    return {}
end if
if |boxes| == 0 then
    return {frame}
end if
Θ ← {}
center ← get_centered_bbox(boxes)
subframes ← split_by_box(frame, center)
for σ ∈ subframes do
    Θ ← Θ ∪ LER(σ, boxes, min_size)
end for
return Θ
```

4.2. Error analysis

We further tested our detector by annotating *all* proposal bounding boxes in the test-episodes of the series as ground truth. In this case, many proposals do not contain one of the characters and are marked as 'unknown'. Error analysis on the false-negative cases revealed that many of them contained character parts such as hands, legs etc. These proposals were identified and annotated as characters by the human annotator, but are still a challenge for automatic classifiers (see Figure 14). This may be an avenue of future research on identifying partial and occluded animated characters and the human factors in a setup of consistent data annotation in scale.

Figure 1: *All characters of our* CAST EVALUATION *data-set from the series 'Bob the Builder', 'Fairly Odd Parents', 'Fireman Sam', 'Floogals', 'Garfield', 'Southpark', and 'The Land Before Time'.*

Figure 2: *Our method's dictionary output on the* SAIL AMCD *Evaluation video - 'Cars 2'.*

Figure 3: *Our method's dictionary extraction over the* CAST EVALUATION *video - 'Southpark'.*

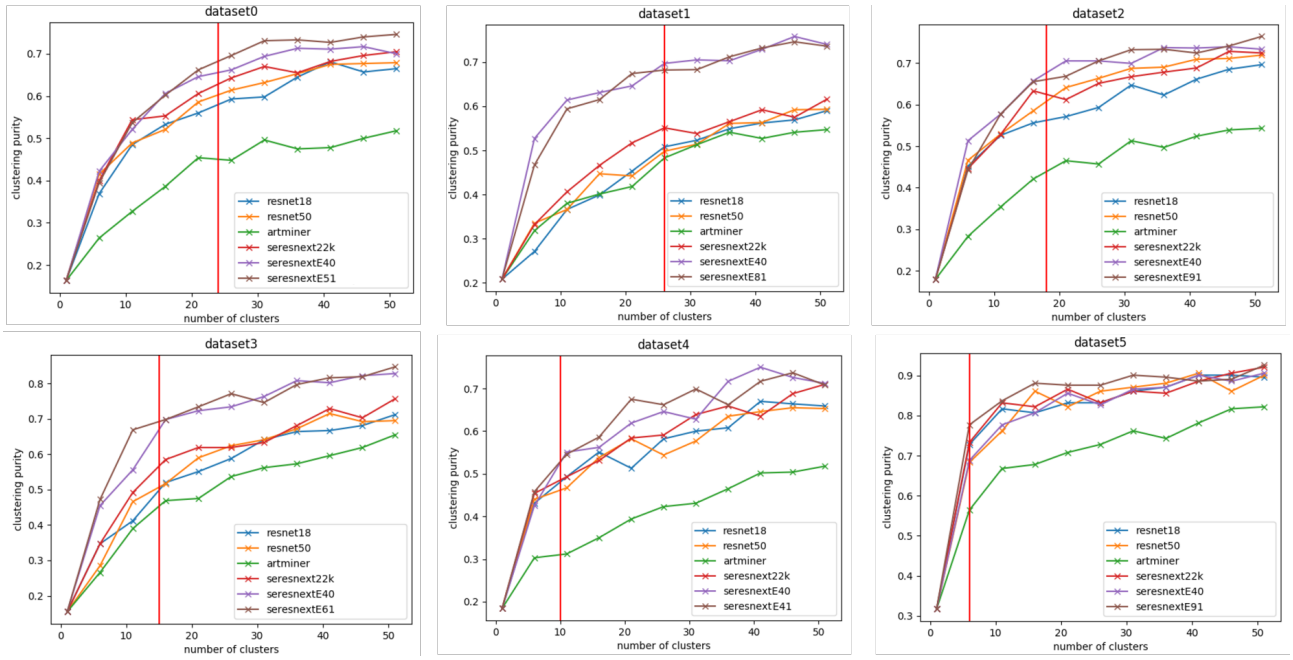

Figure 4: *Comparison of various network architectures for cluster purity on six different character subsets or as named in the figure - datasets. These character subsets were selected at random from the* CAST TEST *data-set. Each graph shows the clustering purity measure as a function of the number of clusters for the different embedding network configurations. The red vertical line indicates the true number of characters (i.e. the number of clusters needed if they were all pure). Our network with a SEResNeXt architecture yields the best results across all data-sets, and this network seems to converge after 40 epochs.*

Figure 5: *'Bob the builder' identification example.*

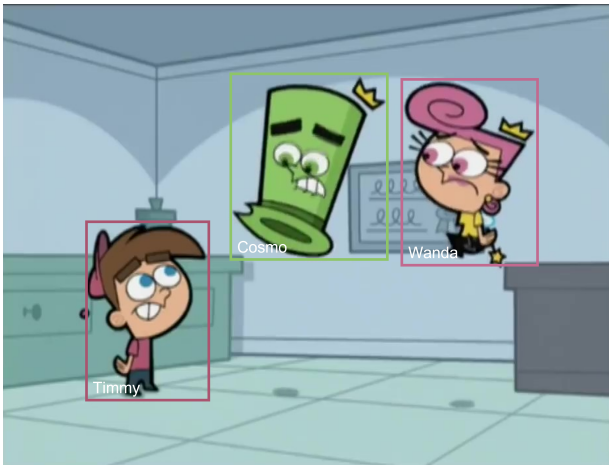

Figure 6: *'Fairly odd parents' identification example.*

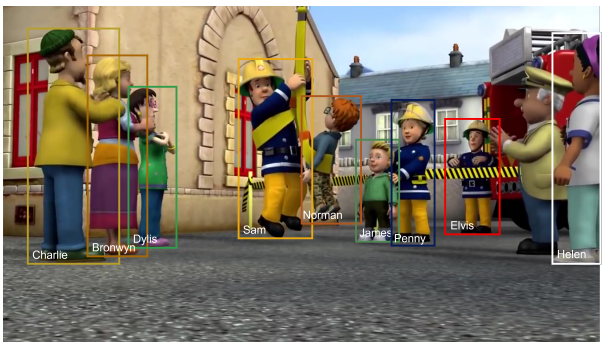

Figure 7: *'Fireman Sam' identification example.*

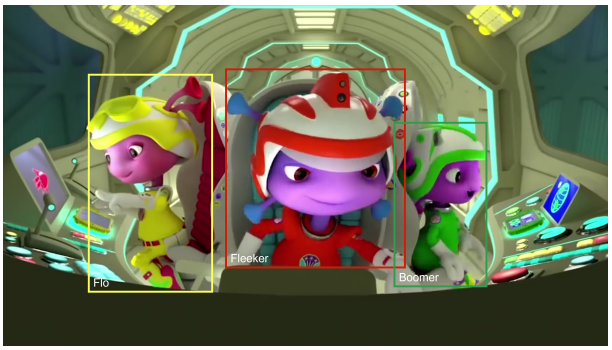

Figure 8: *'Floogals' identification example.*

Figure 9: *'Garfield' identification example.*

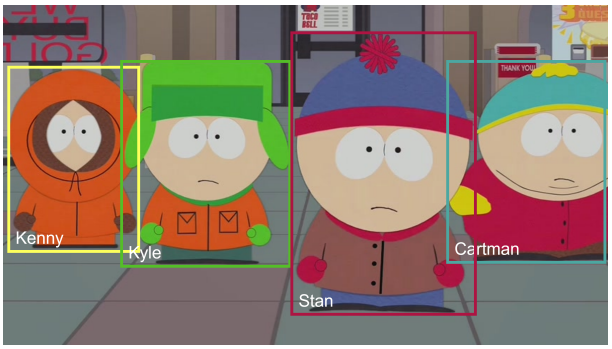

Figure 10: *'Southpark' identification example.*

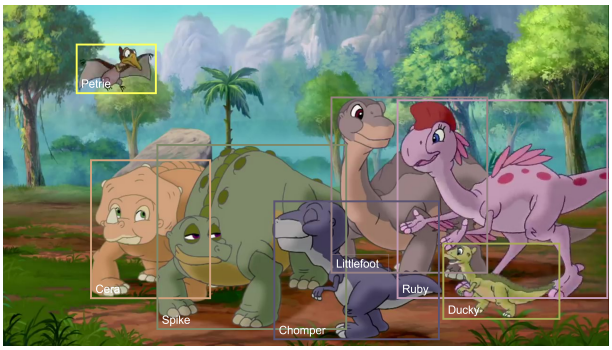

Figure 11: *'The land before time' identification example.*

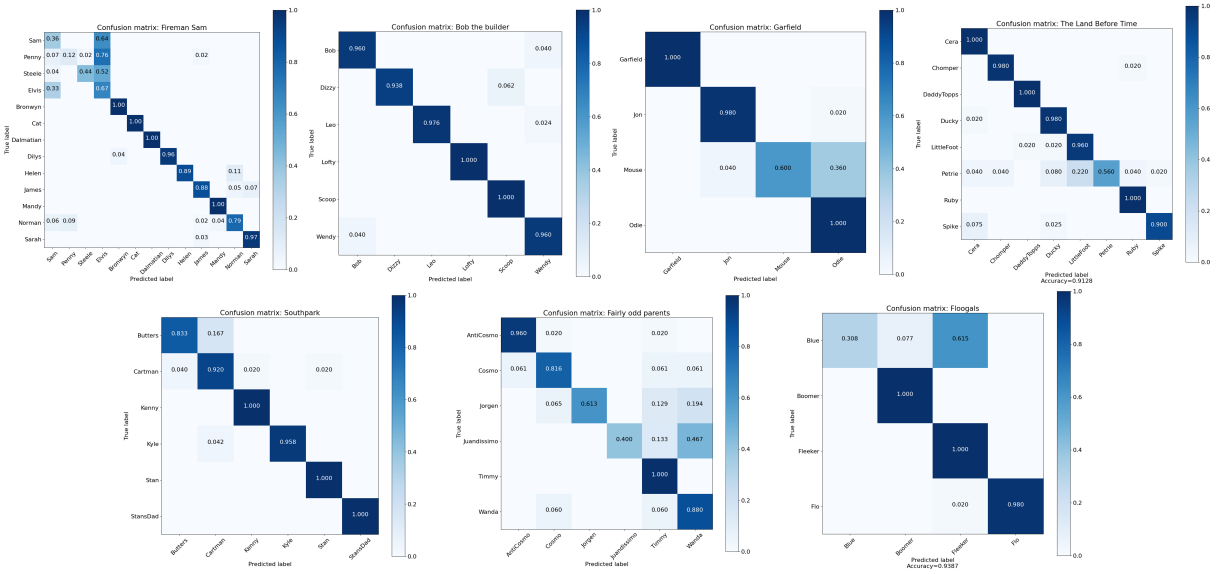

Figure 12: *The* EVALUATION *dataset full Confusion Matrices*

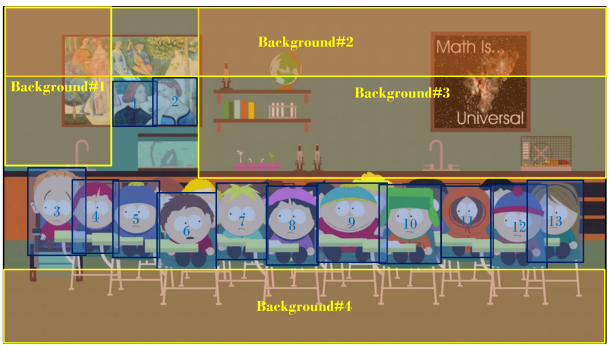

Figure 13: *Negative examples sampling from a video frame using our algorithm to find largest rectangles not including character proposals ('Southpark').*

Figure 14: *False-negative examples include many character subparts (Floogals).*