Supplementary Material:
Actor and Observer: Joint Modeling of First and Third-Person Videos

Gunnar A. Sigurdsson [1*] Abhinav Gupta [1] Cordelia Schmid [2] Ali Farhadi [3] Karteek Alahari [2]
[1]Carnegie Mellon University [2]Inria[†] [3]Allen Institute for Artificial Intelligence
github.com/gsig/actor-observer

Supplementary Material

This supplementary material contains the following.

1. Details of the implementations of the new layers
2. Details of ActorObserverNet
3. Full derivation of the loss with respect to the selector

1. Implementation of the new layers

In this section we derive the equations that are used to update the VideoSoftmax layer (Eq. 3 from the paper) and final loss layer (Eq. 2 and Eq. 4 from the paper) with SGD. This is needed since the equations require computations across samples in different batches (frames in the same video across batches) and we need to do the computation in an online fashion.

VideoSoftmax We want to implement the VideoSoftmax layer to fit the SGD framework. We start with the VideoSoftmax objective (Eq. 3 from the paper):

$$p_\theta(x) = \frac{e^{f_\theta(x)}}{\sum\limits_{x\in\mathcal{V}} e^{f_\theta(x)}}, \tag{1}$$

$$= \frac{e^{f_\theta(x)}}{e^{f_\theta(x)} + \sum\limits_{\tilde{x}\in\mathcal{V}\setminus x} e^{f_\theta(\tilde{x})}}. \tag{2}$$

We now make the normalization explicit:

$$p_\theta(x) = \frac{1}{N}\frac{e^{f_\theta(x)}}{\frac{1}{N}e^{f_\theta(x)} + \frac{N-1}{N}\frac{1}{N-1}\sum\limits_{\tilde{x}\in\mathcal{V}\setminus x} e^{f_\theta(\tilde{x})}}, \tag{3}$$

$$p_\theta(x) = k\frac{e^{f_\theta(x)}}{ke^{f_\theta(x)} + (1-k)\Sigma^{\mathcal{V}}_{N-1}}, \tag{4}$$

where N is the number of terms in the sum. We replace this with a constant k that is defined to be $k = 0.1$. Here $\Sigma^{\mathcal{V}}_N = \frac{1}{N}\sum\limits_{\tilde{x}\in\mathcal{V}\setminus x} e^{f_\theta(\tilde{x})}$. To avoid having p_θ of different ranges for different hyperparameters, we work with $p_\theta(x)/k$ which has the expected value of 1. Our final online update equation is as follows:

$$\Sigma^{\mathcal{V}}_N = ke^{f_\theta(x)} + (1-k)\Sigma^{\mathcal{V}}_{N-1}, \tag{5}$$

$$\frac{p_\theta(x)}{k} = \frac{e^{f_\theta(x)}}{\Sigma^{\mathcal{V}}_N}, \tag{6}$$

where $\Sigma^{\mathcal{V}}_N$ is our online update of the denominator (the normalization constant) for video $\mathcal{V}$.

Loss layer The details of the online equation for the loss are slightly more involved than the previous VideoSoftmax because of the importance sampling, but results in a similarly simple equation. For clarity we use a shorthand notation for the triplet $\tau{=}(x,z,z')$. We start with the loss from Eq. 2 from the paper and write out the normalization constant explicitly:

$$L = \sum_{\tau\sim Q} \frac{p_\theta(\tau)}{q(\tau)} l_\theta(\tau), \tag{7}$$

$$= N\frac{|Q|}{N} \sum_{\tau\sim Q} p_\theta(\tau) l_\theta(\tau), \tag{8}$$

$$\approx N\frac{1}{\sum_{\tau\sim Q} p_\theta(\tau)} \sum_{\tau\sim Q} p_\theta(\tau) l_\theta(\tau), \tag{9}$$

where $\tau{\sim}Q$ indicates that τ is sampled from Q, N is the number of samples we draw from Q, and $|Q|$ is the size of Q. We can write $q(\tau) = \frac{1}{|Q|}$ because Q is uniform. Finally, we use $\frac{1}{|Q|}{\approx}\frac{\sum_{\tau\sim Q} p_\theta(\tau)}{N}$ (normalized importance sampling). We then break the sum into parts for the current sample $\hat{\tau}$ and other samples τ and work with $\frac{L}{N}$:

$$\frac{L}{N} = \frac{1}{\sum\limits_{\tau\sim Q} p_\theta(\tau)} \left(p_\theta(\tau) l_\theta(\tau) + \sum_{\tilde{\tau}\sim Q} p_\theta(\tilde{\tau}) l_\theta(\tilde{\tau}) \right), \tag{10}$$

$$= \frac{1}{p_\theta(\tau) + \sum\limits_{\tilde{\tau}\sim Q} p_\theta(\tilde{\tau})} \left(p_\theta(\tau) l_\theta(\tau) + \sum_{\tilde{\tau}\sim Q} p_\theta(\tilde{\tau}) l_\theta(\tilde{\tau}) \right), \tag{11}$$

$$= \frac{1}{\frac{p_\theta(\tau) + \sum\limits_{\tilde{\tau}\sim Q} p_\theta(\tilde{\tau})}{N}} \left(\frac{1}{N} p_\theta(\tau) l_\theta(\tau) + \frac{N{-}1}{N} \frac{1}{N{-}1} \sum_{\tilde{\tau}\sim Q} p_\theta(\tilde{\tau}) l_\theta(\tilde{\tau}) \right), \tag{12}$$

$$= \frac{1}{k p_\theta(\tau) + (1{-}k)\Sigma_{N-1}} \left(k p_\theta(\tau) l_\theta(\tau) + (1{-}k) \frac{\sum\limits_{\tilde{\tau}\sim Q} p_\theta(\tilde{\tau}) l_\theta(\tilde{\tau})}{(N{-}1) \sum\limits_{\tilde{\tau}\sim Q} p_\theta(\tilde{\tau})} \right), \tag{13}$$

where we scaled the by $\frac{N}{N}$ and $\frac{N-1}{N-1}$ to write the recursive equation. We can see that the right hand side contains $\frac{L}{N-1}$ except with one less samples. We defined $k{=}\frac{1}{N}$ and fix $k{=}0.1$. We define $\Sigma_N = \frac{\sum\limits_{\tilde{\tau}\sim Q} p_\theta(\tilde{\tau})}{N}$. Now we can write this as:

$$L_N = \frac{k p_\theta(\tau) l_\theta(\tau) + (1{-}k)\Sigma_{N-1} L_{N-1}}{\Sigma_N}, \tag{14}$$

$$\Sigma_N = k p_\theta(\tau) + (1{-}k)\Sigma_{N-1}, \tag{15}$$

where L_N is $\frac{L}{N}$ with N samples drawn from Q. This is used to estimate L in order to compute the update in Eq. 4 from the paper.

2. Details of the final triplet network

In this section we describe additional implementation details of the model. The third-person classification loss attaches to the third-person stream when it is used. When in use, the losses are toggled on or off depending if they have training data in the given triplet. In the classification mixed setup the triplet contains either $(x, \emptyset, \emptyset)$ along with a classification label, otherwise we have (x, z, z') and no classification label. The batchsize was set to 15 to accomodate the 3 ResNet-152 streams. We used a $3e^{-5}$ learning rate and reduced the learning rate by 10 every 3 epochs. We used momentum of 0.95. Since our implementation has two very different loss updates (triplet loss, selector update, and classification loss) we found it initially difficult balance the losses. This was solved by rescaling all gradients to have norm equal to the triplet loss gradient.

To balance the ability of the model to choose what samples to learn from and overfitting we introduced a Tanh layer to bound the possible $f(x)$ (Eq. 3 from the paper) values in the network. We allowed the model to learn this weight for each

p starting from Gaussian noise of σ=5. While sharing of the FC layers and scaling parameters did not affect the results in Section 4.2, we found the network to have better performance in Section 4.3 and Section 4.4 when sharing parameters between the two first-person FC streams, and constraining the TanH scale to be positive. This is likely because it adds additional constraints on the selector, and discourages it from overfitting.

To ensure consistent training and testing setup. The triplets from Q consist of every third-person frame, paired with the best corresponding frame (alignment error is estimated to be approximately a second, which implies Δ=1 sec), as well as a randomly sample noncorresponding frame that it at least 10 seconds away.

3. Full derivation of the loss with respect to the selector

The loss in Eq. 2 includes contribution from p_θ, where each output of p_θ (VideoSoftmax) is normalized across all frames in that video. We assume that the last layer before the loss layer is a softmax layer (i.e. VideoSoftmax):

$$p_\theta(\tau) = \frac{e^{f(\tau)}}{\sum_{\tilde{\tau}} e^{f(\tilde{\tau})}}, \tag{16}$$

$$= \frac{e^{f(\tau)}}{e^{f(\tau)} + \sum_{\tilde{\tau}} e^{f(\tilde{\tau})}}. \tag{17}$$

This allows us to account for the contribution of $e^{f(\tau)}$ to other samples. That is, note that the denominator includes the values over other frames (in the same video), so those terms have to be included in the derivative, the second equation clarifies this relationship by separating the triplet of interest from the sum. For clarity we again use a shorthand notation for the triplet τ=(x,z,z').

We start with Eq. 2 from the paper, and insert the assumption for p_θ:

$$L = \sum_{\tau} p_\theta(\tau) l_\theta(\tau), \tag{18}$$

$$= \sum_{\tau} \frac{e^{f(\tau)}}{\sum_{\tilde{\tau}} e^{f(\tilde{\tau})}} l_\theta(\tau), \tag{19}$$

where we use $\tilde{\tau}$ to emphasize different triplets. We take the derivative of this with respect to $f(\hat{\tau})$, a particular input to the Softmax that occurs once in the numerator and many times in the denominator:

$$\frac{\partial L}{\partial f(\hat{\tau})} = \frac{\partial}{\partial f(\hat{\tau})} \sum_{\tau} \frac{e^{f(\tau)}}{\sum_{\tilde{\tau}} e^{f(\tilde{\tau})}} l_\theta(\tau), \tag{20}$$

$$= \frac{\partial}{\partial f(\hat{\tau})} \frac{e^{f(\hat{\tau})}}{\sum_{\tilde{\tau}} e^{f(\tilde{\tau})}} l_\theta(\hat{\tau}) + \sum_{\tau \setminus \hat{\tau}} \frac{e^{f(\tau)}}{\sum_{\tilde{\tau}} e^{f(\tilde{\tau})}} l_\theta(\tau), \tag{21}$$

$$= \frac{\partial}{\partial f(\hat{\tau})} \frac{e^{f(\hat{\tau})}}{e^{f(\hat{\tau})} + \sum_{\tau \setminus \hat{\tau}} e^{f(\tau)}} l_\theta(\hat{\tau}) + \frac{\partial}{\partial f(\hat{\tau})} \sum_{\tau \setminus \hat{\tau}} \frac{e^{f(\tau)}}{e^{f(\hat{\tau})} + \sum_{\tau \setminus \hat{\tau}} e^{f(\tau)}} l_\theta(\tau), \tag{22}$$

where we have expanded the softmax to make clear where $\hat{\tau}$ occurs in the numerator and denominator. That is, $\tau \setminus \hat{\tau}$ is the set of all τ that excludes $\hat{\tau}$. We then find the derivative similarly to the derivative of softmax, where we write it in terms of

$p_\theta(\hat{\tau})$ for clarity:

$$\frac{\partial L}{\partial f(\hat{\tau})} = p_\theta(\hat{\tau})(1-p_\theta(\hat{\tau}))l_\theta(\hat{\tau}) + \sum_{\tau \diagdown \hat{\tau}} p_\theta(\hat{\tau})(-p_\theta(\tau))l_\theta(\tau), \tag{23}$$

$$= p_\theta(\hat{\tau})l_\theta(\hat{\tau}) - p_\theta(\hat{\tau})p_\theta(\hat{\tau})l_\theta(\hat{\tau}) + \sum_{\tau \diagdown \hat{\tau}} p_\theta(\hat{\tau})(-p_\theta(\tau))l_\theta(\tau), \tag{24}$$

$$= p_\theta(\hat{\tau})l_\theta(\hat{\tau}) - p_\theta(\hat{\tau}) \left(p_\theta(\hat{\tau})l_\theta(\hat{\tau}) + \sum_{\tau \diagdown \hat{\tau}} p_\theta(\hat{\tau})l_\theta(\tau) \right), \tag{25}$$

$$= p_\theta(\hat{\tau})l_\theta(\hat{\tau}) - p_\theta(\hat{\tau})L, \tag{26}$$

$$= p_\theta(\hat{\tau})(l_\theta(\hat{\tau}) - L). \tag{27}$$

Here we factorize and recombine the terms that constitute the definition of L, allowing us to write this in a compact format. These terms are then implemented in an online fashion as previously described. The final update is Eq. 4 in the paper:

$$\frac{\partial L}{\partial f(x,z,z')} = p_\theta(x,z,z')(l_\theta(x,z,z') - L). \tag{28}$$