Visual-textual Capsule Routing for Text-based Video Segmentation

Bruce McIntosh  
bwmcint@gmail.com

Kevin Duarte  
kevin95duarte@gmail.com

Yogesh S Rawat  
yogesh@crcv.ucf.edu

Mubarak Shah  
shah@crcv.ucf.edu

Center for Research in Computer Vision  
University of Central Florida  
Orlando, FL, 32816

Abstract

Joint understanding of vision and natural language is a challenging problem with a wide range of applications in artificial intelligence. In this work, we focus on integration of video and text for the task of actor and action video segmentation from a sentence. We propose a capsule-based approach which performs pixel-level localization based on a natural language query describing the actor of interest. We encode both the video and textual input in the form of capsules, which provide a more effective representation in comparison with standard convolution based features. Our novel visual-textual routing mechanism allows for the fusion of video and text capsules to successfully localize the actor and action. The existing works on actor-action localization are mainly focused on localization in a single frame instead of the full video. Different from existing works, we propose to perform the localization on all frames of the video. To validate the potential of the proposed network for actor and action video localization, we extend an existing actor-action dataset (A2D) with annotations for all the frames. The experimental evaluation demonstrates the effectiveness of our capsule network for text selective actor and action localization in videos. The proposed method also improves upon the performance of the existing state-of-the-art works on single frame-based localization.

1. Introduction

Deep learning and artificial neural networks have led to outstanding advancements in the fields of computer vision and natural language processing (NLP). In recent years, the vision and NLP communities have proposed several tasks which require methods to understand both visual and textual inputs. These include visual question answering [1], image and video captioning [29, 30], visual text correction [21], and video generation from text inputs [18]. In this work, we focus on detection of actors and actions in a video through natural language queries.

Actor and action detection in a video is an important task in computer vision and it has many applications, such as video retrieval, human-machine interaction, and surveillance. Most of the existing methods focus on detection of actor/action which are from a fixed set of categories. Instead of having these fixed categories, one can leverage natural language to describe the actors and actions which needs to be localized. This describes the task of actor and action video segmentation from a sentence [8]: given a video and a natural language sentence input, the goal is to output a pixel-level localization of the actor described by the sentence. For a method to perform this task, it must effectively merge the visual and textual inputs to generate a segmentation mask for the actor of interest.

The existing methods for video encoding are mainly based on 3D convolutions. The availability of large-scale datasets allow us to train effective 3D convolution based models, however, this encoded representation has some limitations as it fails to capture the relationship between different features. Capsule-based networks address some of these limitations and are effective in modeling visual entities and capturing their relationships [24]. Capsule net-
works are composed of groups of neurons called capsules which model objects or object-parts. These capsules undergo a routing-by-agreement procedure which allows it to learn relationships between these entities. Capsule networks are shown to be effective in both video [5] as well as textual domain [31]. In this work, we explore the use of capsules to jointly encode and merge visual and textual information for the task of actor and action detection in videos.

We propose an end-to-end capsule-based network for actor-action segmentation using a natural language query. The video and the textual query, both are encoded as capsules for learning an effective representation. We demonstrate that capsules and routing-by-agreement can be utilized for the integration of both visual and textual information. Our novel routing algorithm finds agreement between the visual and textual entities to produce a unified representation in the form of visual-textual capsules.

Our main contributions are summarized as follows:

- We propose an end-to-end capsule network for the task of selective actor and action localization in videos, which encodes both the video and the textual query in the form of capsules.
- We introduce a novel visual-textual capsule routing algorithm which fuses both modalities to create a unified capsule representation.
- To demonstrate the potential of the proposed text-selective actor and action localization in videos, we extend the annotations in A2D dataset to full video clips.

Our experiments demonstrate the effectiveness of the proposed method, and we show its advantage over existing state-of-the-art works both qualitatively and quantitatively.

2. Related Work

Vision and Language Both vision and language have been used in several challenging problems. Several works have dealt with image captioning [7, 27] and video captioning [6] where a natural language description is generated for a given image or video. Zero-shot object detection from a textual input is explored by [20], which can localize novel object instances when given a textual description. In the video domain, a popular problem is that of temporal localization using natural language [9, 3, 4], where a method must localize the temporal boundary of the action described by a text query. The task of actor and action video segmentation given a sentence is similar, but a pixel-level segmentation of the described actor is output. The only work dealing with this is [8], however only a single frame is segmented. We believe that video segmentation should produce a segmentation for all frames in a video, so we extend the A2D dataset with annotations for all frames.

Merging Visual and Textual Inputs Hu et al. [12] introduced the problem of segmenting images based on a natural language expression; their method for merging images and text in a convolutional neural network (CNN) was by concatenating features extracted from both modalities and performing a convolution to obtain a unified representation. [19] propose a different approach to merge these modalities for the task of tracking a target in a video; they use an element-wise multiplication between the image features and the sentence features in a process called dynamic filtering. These are the two most commonly used approaches for merging both vision and language in a neural network. We present the first capsule-based approach which uses routing-by-agreement to merge both visual and textual inputs.

Capsule Networks Hinton et al. first introduced the idea of capsules in [10], and subsequently capsules were popularized in [24], where dynamic routing for capsules was proposed. This was further extended in [11], where a more effective EM routing algorithm was introduced. Recently, capsule networks have shown state-of-the-art results for human action localization in video [5], object segmentation in medical images [17], and text classification [31]. [32] proposed a capsule-based attention mechanism for the task of visual question answering. To our knowledge, our work is the first to use capsules and routing to combine both video and natural language inputs.

3. Visual-Textual Capsule Routing

Brief Introduction to Capsule Networks A capsule is a group of neurons that models objects, or parts of objects. In this work, we use the matrix capsule formulation proposed by [11], where a capsule, \( C \), is composed of a \( 4 \times 4 \) pose matrix \( M \), and an activation \( a \in [0, 1] \). The pose matrix contains the instantiation parameters, or properties, of the object modeled by the capsule and the activation is the existence probability of the object. Capsules from one layer pass information to capsules through a routing-by-agreement operation. This begins when the lower level capsules produce votes for the capsules in the higher level; these votes, \( V_{ij} = M, T_{ij} \), are the result of a matrix multiplication between learned transformation matrices, \( T_{ij} \), and the lower level pose matrices, where \( i \) and \( j \) are the indices of the lower and higher level capsules respectively. Once these votes are obtained, they are used in the EM-routing algorithm to obtain the higher level capsules \( C_j \), with pose matrices \( M_j \) and activations \( a_j \).

Our Routing Method Capsules represent entities and routing uses high-dimensional coincidence filtering [11] to learn part-to-whole relationships between these entities. We argue that this allows capsule networks to effectively merge
visual and textual information. There are several possible ways to implement this using capsule networks. One simple approach would be to apply a convolutional method (concatenation followed by a 1x1 convolution [12] or multiplication/dynamic filtering [19]) to create a unified representation in the form of feature maps, and extract a set of capsules from these feature maps. This, however, would not perform much better than the fully convolutional networks, since the same representation is obtained from the merging of the visual and textual modalities, and the only difference is how they are transformed into segmentation maps.

Another method would be to first extract a set of capsules from the video, and then apply the dynamic filtering on these capsules. This can be done by (1) applying a dynamic filter to the pose matrices of the capsules, or (2) applying a dynamic filter to the activations of the capsules. The first is not much different than the simple approach described above, since the same feature map representation would be present in the capsule pose matrices, as opposed to the layer prior to the capsules. The second approach would just discount importance of the votes corresponding to entities not present in the sentence; this is not ideal, since it does not take advantage of routing’s ability to find agreement between entities in both modalities.

Instead, we propose an approach that leverages the fact that the same entities exist in both the video and sentence inputs and that routing can find similarities between these entities. Our method allows the network to learn a set of entities (capsules) from both the visual and sentence inputs. With these entities, the capsule routing finds the similarity between the objects in the video and sentence inputs to generate a unified visual-textual capsule representation.

More formally, we extract a grid of capsules describing the visual entities, $C_v$, with pose matrices $M_v$ and activations $a_v$ from the video. Similarly, we generate sentence capsules, $C_s$, with pose matrices $M_s$ and activations $a_s$ for the sentence. Each set of capsules has learned transformation matrices $T_v$ and $T_s$, for video and text respectively, which are used to cast votes for the capsules in the following layer. Video capsules at different spatial locations share the same transformation matrices. Using the procedure described in Algorithm 1, we obtain a grid of higher-level capsules, $C_j$. This algorithm allows the network to find similarity, or agreement, between the votes of the video and sentence capsules at every location on the grid. If there is agreement, then the same entity exists in both the sentence and the given location in the video, leading to a high activation of the capsule corresponding to that entity. Conversely, if the sentence does not describe the entity present at the given spatial location, then the activation of the higher-level capsules will be low since the votes would disagree.

Algorithm 1 Visual-Textual Capsule Routing. The inputs to this procedure are the video capsules’ poses and activations $(M_v, a_v)$ and the sentence capsules’ poses and activations $(M_s, a_s)$. The $(\cdot, \cdot)$ operation is concatenation, such that the activations and votes the video and sentence capsules are inputs to the EM ROUTING procedure described in [11].

\begin{algorithm}
\caption{Visual-Textual Capsule Routing}
\begin{algorithmic}[1]
\State $V_{ij} \leftarrow M_v \cdot T_{vj}$
\For{$x = 1$ to $W$}
\For{$y = 1$ to $H$}
\State $V_{ij} \leftarrow M_v [x, y] \cdot T_{vj}$
\State $a \leftarrow \{ a_s; a_v [x, y] \}$
\State $V \leftarrow \{ V_{ij}; V_{vj} \}$
\State $C_j [x, y] \leftarrow$ EM ROUTING $(a, V)$
\EndFor
\EndFor
\State return $C_j$
\end{algorithmic}
\end{algorithm}

4. Network Architecture

The overall network architecture is shown in Figure 2. In this section, we discuss the components of the architecture as well as the objective function used to train the network.

4.1. Video Capsules

The video input consists of $4 \times 224 \times 224$ frames. The process for generating video capsules begins with a 3D convolutional network known as I3D [2], which generates $832 \times 28 \times 28$ spatio-temporal feature maps taken from the max-pool3d.3a.3x3 layer. Capsule pose matrices and activations are generated by applying a $9 \times 9$ convolution operation to these feature maps, with linear and sigmoid activations respectively. Since there is no padding for this operation, the result is a $20 \times 20$ capsule layer with 8 capsule types.

4.2. Sentence Capsules

A series of convolutional and fully connected layers is used to generate the sentence capsules. First, each word from the sentence is converted into a size 300 vector using a word2vec model pre-trained on the Google News Corpus [22]. The sentence representation is then passed through 3 parallel stages of 1D convolution with kernel sizes of 2, 3 and 4 with a ReLU activation. We then apply max-pooling to obtain 3 vectors, which are concatenated and passed through a max-pooling layer to obtain a single length 300 vector to describe the entire sentence. A fully connected layer then generates the 8 pose matrices and 8 activations for the capsules which represent the entire sentence. We found that this method of generating sentence capsules performed best in our network: various other methods are explored in the Supplementary Material.

4.3. Merging and Masking

Once the video and sentence capsules are obtained, we merge them using the proposed routing algorithm. The re-
result of the routing operation is a $20 \times 20$ grid with 8 capsule types - one for each actor class in the A2D dataset and one for a “background” class, which is used to route unnecessary information. The activations of these capsules correspond to the existence of the corresponding actor at the given location, so averaging the activations over all locations gives us a classification prediction over the video clip. We find that this class to capsule correspondence improves the network’s segmentations overall.

We perform the capsule masking as described in [24]. When training the network, we mask (multiply by 0) all pose matrices not corresponding to the ground truth class. At test time, we mask the pose matrices not corresponding to the predicted class. These masked poses are then fed into an upsampling network to generate a foreground/background actor segmentation mask. Our network outperforms contemporary methods without classification and masking, but this extra supervision signal improves the performance. We explore this further in our ablations.

4.4. Upsampling Network

The upsampling network consists of 5 convolutional transpose layers. The first of these increases the feature map dimension from $20 \times 20$ to $28 \times 28$ with a $9 \times 9$ kernel, which corresponds to the $9 \times 9$ kernel used to create the video capsules from the I3D feature maps. The following 3 layers have $3 \times 3 \times 3$ kernels and are strided in both time and space, so that the output dimensions are equal to the input video dimensions ($4 \times 224 \times 224$). The final segmentation is produced by a final layer which has a $3 \times 3 \times 3$ kernel. Note that a unique feature of our method compared to previous method is it outputs segmentations for all input frames, rather than a single frame segmentation per video clip input. We use parameterized skip connections from the I3D encoder to obtain more fine-grained segmentations.

4.5. Objective Function

The network is trained end-to-end using an objective function based on classification and segmentation. For classification, we use a spread loss which is computed as:

$$L_c = \sum_{i \neq t} \max (0, m - (a_t - a_i))^2,$$

where $m \in (0, 1)$ is a margin, $a_t$ is the activation of the capsule corresponding to class $i$, and $a_i$ is the activation of the capsule corresponding to the ground-truth class. During training, $m$ is linearly increased between 0.2 and 0.9, following the standard set by [11, 5].

The segmentation loss is computed using sigmoid cross entropy. When averaged over all $N$ pixels in the segmentation map, we get the following loss:

$$L_s = -\frac{1}{N} \sum_{j=1}^{N} p_j \log (\hat{p}_j) - (1 - p_j) \log (1 - \hat{p}_j),$$

where $p_j \in \{0, 1\}$ is the ground-truth segmentation map and $\hat{p}_j \in [0, 1]$ is the network’s output segmentation map.

The final loss is a weighted sum between the classification and segmentation losses:

$$L = \lambda L_c + (1 - \lambda) L_s,$$

where $\lambda$ is set to 0.5 when training begins. Since the network quickly learns to classify the actor when given a sentence input, we set $\lambda$ to 0 when the classification accuracy saturates (over 95% on the validation set). We find that this reduces over-fitting and results in better segmentations.

5. Experiments

Implementation Details The network was implemented using PyTorch [23]. The I3D used weights pretrained on Kinetics [14] and fine tuned on Charades [26]. The network was trained using the Adam optimizer [16] with a learning rate of .001. As video resolutions vary within different
| Hu et al. [12] | 34.8 23.6 13.3 3.3 0.1 13.2 47.4 35.0 |
| Li et al. [19] | 38.7 29.0 17.5 6.6 0.1 16.3 51.5 35.4 |
| Gavrilyuk et al. [8] | 50.0 37.6 23.1 9.4 0.4 21.5 55.1 42.6 |
| **Our Network** | **52.6** **45.0** **34.5** **20.7** **3.6** **30.3** **56.8** **46.0** |

Table 1. Results on A2D dataset with sentences. Baselines [12, 19] take only single image/frame inputs. Gavrilyuk et al. [8] uses multi-frame RGB and Flow inputs. Our model uses only multi-frame RGB inputs and outperforms other state-of-art-methods in all metrics without the use of optical flow.

| Hu et al. [12] | 63.3 35.0 8.5 0.2 0.0 17.8 54.6 52.8 |
| Li et al. [19] | 57.8 33.5 10.3 0.6 0.0 17.3 52.9 49.1 |
| Gavrilyuk et al. [8] | **69.9** **46.0** **17.3** **1.4** **0.0** **23.3** **54.1** **54.2** |
| **Our Network** | **67.7** **51.3** **28.3** **5.1** **0.0** **26.1** **53.5** **55.0** |

Table 2. Results on JHMDB dataset with sentences. Our model outperforms other state-of-the-art methods at higher IoU thresholds and in the mean average precision metric.

Datasets: We conduct our experiments on two datasets: A2D [28] and J-HMDB [13]. The A2D dataset contains 3782 videos (3036 for training and 746 for testing) consisting of 7 actor classes, 8 action classes, and an extra action label *none*, which accounts for actors in the background or actions different from the 8 action classes. Since actors cannot perform all labeled actions, there are a total of 43 valid actor-action pairs. Each video in A2D has 3 to 5 frames which are annotated with pixel-level actor-action segmentations. The J-HMDB dataset contains 928 short videos with 21 different action classes. All frames in the J-HMDB dataset are annotated with pixel-level segmentation masks. Gavrilyuk et al. [8] extended both of these datasets with human generated sentences that describe the actors of interest for each video. These sentences use the actor and action as part of the description, but many do not include the action and rely on other descriptors such as location or color.

Evaluation: We evaluate our results using all metrics used in [8]. The overall IoU is the intersection-over-union (IoU) over all samples, which tends to favor larger actors and objects. The mean IoU is the IoU averaged over all samples, which treats samples of different sizes equally. We also measure the precision at 5 IoU thresholds and the mean average precision over .50 : .05 : .95.

Results: We compare our results on A2D with previous approaches in Table 1. Our network outperforms previous state-of-the-art methods in all metrics, and has a notable 8.8% improvement in the mAP metric, even though we do not employ optical-flow, which requires extra computation. We also find that our network achieves much stronger results at higher IoU thresholds, which signifies that the segmentations produced by the network are more fine-grained and adhere to the contours of the queried objects. Qualitative results on A2D can be found in Figure 3.

Following the testing procedure in [8], we test on all the videos of J-HMDB using our model trained on A2D without fine-tuning. The results on J-HMDB are found in Table 2; our network outperforms other methods at the higher IoU thresholds (0.6, 0.7, and 0.8), the mAP metric, and in mean IoU. We perform slightly worse at the lower threshold and in overall IoU. We find that our network performs poorly on J-HMDB actions which have little motion like “brush-hair”, “stand”, and “sit” (which have an IoU > 0.5 for less than 20% of the videos). On the other hand, our network
performs well on actions with larger amounts of motion like “pullup”, “swing baseball”, and “shoot ball”. Since A2D videos tend to have large amounts of motion, we believe that training on A2D forced our network to focus on motion cues which are not present in J-HMDB.

5.2. Full Video Segmentation from a Sentence

In this set of experiments, we train the network using the bounding box annotations for all the frames. Since previous baselines only output single frame segmentations, we test our method against our single-frame segmentation network as a baseline which can generate segmentations for an entire video, by processing the video frame-by-frame.

Importance of full video segmentation  Previous methods for actor and action video segmentation from a sentence [8] process multiple frames but only segments a single frame at a time. We find this to be a weakness for two reasons: 1) it negatively impacts the temporal consistency of the generated segmentations and 2) it increases the computational time for generating segmentations for an entire video. Therefore, we propose a method which generates segmentation masks for the entire video at a time.

A2D dataset extension  To successfully train and evaluate such a model, one would need a video dataset which contained localization annotations for all video frames. To this end, we extend the A2D dataset by adding bounding box localizations for the actors of interest in every frame of the dataset. This allows us to train and test our method using the entire video, not just the 3 to 5 key frames which were previously annotated. The extended A2D dataset contains annotations for 6046 actors, with an average of 136 bounding boxes per actor. These annotations will be made publicly available.

Datasets  For the full video segmentation experiments we use the extended A2D dataset. We use the same train and test video splits defined in [8], but the new annotations allow for training and evaluation on all video frames. The J-HMDB dataset has annotations on all frames, so we can evaluate the method on this dataset as well.

Evaluation  To evaluate the segmentation results for entire videos, we consider each video as a single sample. Thus, the IoU computed is the intersection-over-union between the ground-truth tube and the generated segmentation tube. Using this metric, we can calculate the video overall IoU and the video mean IoU; the former will favor both larger objects and objects in longer videos, while the latter will treat all videos equally. We also measure the precision at 5 different IoU thresholds and the video mean average precision over .50 : .05 : .95.

Results  Since the network is trained using the bounding box annotations, the segmentations are more block-like, but
it still successfully segments the actors described in the
given queries. We compare the qualitative results between
the network trained only using fine-grained segmentations
and the network trained using bounding box annotations in
Figure 4. When tested on the A2D dataset, we find that
there is a significant improvement in all metrics when com-
pared to the network trained only on single frames with
pixel-wise segmentations. However, this is to be expected,
since the ground-truth tubes are bounding boxes and box-
like segmentations around the actor would produce higher
IoU scores. For a fairer comparison, we place a bound-
ing box around the fine-grained segmentations produced by
the network trained on the pixel-wise annotations; this pro-
duces better results since the new outputs more resemble
the ground-truth tubes. Even with this change, the network
trained on bounding box annotations has the strongest re-
sults since it learned from all frames in the training videos,
as opposed to a handful of frames per video (Table 3).

The J-HMDB dataset has pixel-level annotations for all
frames, so the box-like segmentations produced by the net-
work should be detrimental to results; we found that this
was the case: the network performed poorly when com-
pared to the network trained on fine-grained pixel-level
annotations. However, if evaluation is performed on bound-
ing boxes surrounding the ground-truth segmentations, then
considerable improvements are observed across all metrics.

5.3. Image Segmentation Conditioned on Sentences

To investigate the versatility of the visual-textual routing
algorithm, we also evaluate our method by segmenting im-
gages based on text queries. To make as few modifications
to the network as possible, the single images are repeated to
create a “boring” video input with 4 identical frames.

Dataset We use the ReferItGame dataset [15], which con-
tains 20000 images with 130525 natural language expres-
sions describing various objects in the images. We use the
same train/test splits as [12, 25], with 9000 training and
10000 testing images. Unlike A2D there are no predefined
set of actors, so no classification loss or masking is used.

Results We obtain similar results to other state-of-the-art
approaches, even though our network architecture is de-
signaled for actor/action video segmentation. At high IoU
thresholds, our network’s precision outperforms [12] and is
within 3% of [25]. This demonstrates that our proposed
method for merging visual and textual information is effec-
tive on multiple visual modalities - both videos and images.

5.4. Ablation Studies

The ablation experiments were trained and evaluated us-
ing the pixel-level segmentations from the A2D dataset. All
ablation results can be found in Table 4.

Classification and Masking We test the influence of the
classification loss for this segmentation task, by running
an experiment without back-propogating this loss. Without
classification, the masking procedure would fail at test time,
so masking is not used and all poses are passed forward to
the upsampling network. This performed slightly worse
than the baseline in all metrics, which shows that the clas-
sification loss and masking help the capsules learn meaning-
ful representations. The network, however, still performs
segmentation well without this extra supervision: this abla-
tion outperforms previous methods on the A2D dataset in
all metrics except Overlap P@0.5. To further investigate
the effects of masking, we perform an experiment with no
masking, but with the classification loss. Surprisingly, it
performs worse than the network without masking nor clas-
sification loss; this signifies that classification loss can be
detrimental to this segmentation task, if there is no masking
to guide the flow of the segmentation loss gradient.

Effectiveness of Visual-Textual Routing We run several
experiments to compare our visual-textual capsule routing
procedure with alternative methods for merging video and
text. We test the four other methods for fusing visual and
textual information described earlier: the two trivial ap-
proaches (concatenation and multiplication), and the two
methods which apply dynamic filtering to the video cap-
sules (filtering the pose matrices and filtering the activa-
tions). The two trivial, convolutional-based approaches lead
to a significant decrease in performance (a decrease of about
21% and 11% in mean IoU respectively) when compared
to our visual-textual routing approach. Moreover, apply-
ing dynamic filtering to the video capsules results in about

<table>
<thead>
<tr>
<th></th>
<th>Video Overlap</th>
<th>v-mAP 0.5:0.95</th>
<th>Video IoU</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P@0.5</td>
<td>P@0.6</td>
<td>P@0.7</td>
</tr>
<tr>
<td>Key frames (pixel)</td>
<td>9.6</td>
<td>1.6</td>
<td>0.4</td>
</tr>
<tr>
<td>Key frames (bbox)</td>
<td>41.9</td>
<td>33.3</td>
<td>22.2</td>
</tr>
<tr>
<td>All frames</td>
<td>45.6</td>
<td>37.4</td>
<td>25.3</td>
</tr>
</tbody>
</table>

Table 3. Results on A2D dataset with bounding box annotations. The first row is for the network trained with only pixel-level annotations on key frames of the video, and evaluated with its pixel-wise segmentation output. The second is the same network, but a bounding-box is placed around its segmentation output for evaluation. The final row, is the network trained with bounding box annotations on all frames.
Table 4. Ablations on the A2D dataset with sentences. The last row shows the results of our final network.

<table>
<thead>
<tr>
<th>Method</th>
<th>P@0.5</th>
<th>mAP</th>
<th>Mean IoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>No $L_c$ nor Masking</td>
<td>49.4</td>
<td>28.8</td>
<td>43.6</td>
</tr>
<tr>
<td>No Masking (with $L_c$)</td>
<td>48.3</td>
<td>27.8</td>
<td>42.5</td>
</tr>
<tr>
<td>Concatenation</td>
<td>22.9</td>
<td>9.9</td>
<td>25.0</td>
</tr>
<tr>
<td>Multiplication</td>
<td>38.4</td>
<td>19.4</td>
<td>35.0</td>
</tr>
<tr>
<td>Filter Poses</td>
<td>49.1</td>
<td>29.1</td>
<td>42.7</td>
</tr>
<tr>
<td>Filter Activations</td>
<td>48.8</td>
<td>29.2</td>
<td>43.0</td>
</tr>
<tr>
<td>Our Network</td>
<td>52.6</td>
<td>30.3</td>
<td>46.0</td>
</tr>
</tbody>
</table>

a 3% decrease in mean IoU and a 4% decrease in Overlap P@0.5, showing that it is not a simple task to extend techniques developed for CNNs, like dynamic filtering, to capsule networks. Rather, new capsule and routing based approaches, like visual-textual routing, must be developed to fully leverage the capabilities of capsule networks.

6. Discussion and Analysis

Failure Cases  We find that the network has two main failure cases: (1) the network incorrectly selects an actor which is not described in the query, and (2) the network fails to segment anything in the video. Figure 6 contains examples of both cases. The first case occurs when the text query refers to an actor/action pair and multiple actors are doing this action or the video is cluttered with many possible actors from which to choose. This suggests that an improved video encoder which extracts better video feature representations and creates more meaningful video capsules could improve results. The second failure case tends to occur when the queried object is small, which is often the case with the “ball” class or when the actor of interest is far away.

How sentences are utilized We analyze the extent to which the model leverages the visual input and textual query. We present several cases where the network is given multiple queries for the same video in Figure 5. If the network is given a query which is invalid for a given video - this occurs when the actor described in the sentence is not present in the video - we find that our network correctly segments nothing; this behaviour is depicted in the first image of Figure 5. Moreover, if the network is given a sentence which describes multiple actors in the scene, it can segment all actors that are being described; this can be seen in the second image of Figure 5 where the sentence “Dogs running on the beach” is given to the network and both dogs are segmented. Our network can segment based on the action specified in the query; when given two similar sentences “The man walking to the right” and “The man standing on the right”, the network has learned the difference between the walking and standing actions and correctly segments the walking person only when the prior sentence is given. The A2D dataset is focused on actors and actions, so these tend to be the most powerful descriptors the network learns. The words “left” and “right” are frequently found in the training sentences, so the network seems to have a good grasp of these words as well. The network also understands other descriptors like color or size, but we find that these are less reliable since they occur less frequently in the training set.

7. Conclusion

In this work, we propose a capsule network for localization of actor and actions based on a textual query. The proposed framework makes use of capsules for both video as well as textual representation. By using visual-textual routing, our network successfully segments actors and actions in video, conditioned on a textual query. We extended the A2D dataset from single frame to all frame annotation to validate our performance. We demonstrate the effectiveness of visual-textual capsule routing and observe performance improvements over state-of-the-art approaches.

Acknowledgments

This research is based upon work supported in parts by the National Science Foundation under Grants No. 1741431 and the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA RD Contract No. D17PC00345. The views, findings, opinions, and conclusions or recommendations contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the NSF, ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.
References


