Appendix

Appendix provides details on:

1. Relative Position Encoding (Section A)
2. ActivityNet-SRL construction and statistics (Section B)
3. Evaluation Metrics (Section C)
4. Implementation Details (Section D)
5. Additional Experiments (Section E)
6. Visualizations (Section F)

A. Relative Position Encoding Discussion

In this section, we describe the challenges of using relative position encoding, followed by an overview of the method used in [54] and finally show how we adapt their formulation to our setting. For an overview of the technical details of the Transformer [61], we refer to the following well-written blogs "The Annotated Transformer"[2], "The Illustrated Transformer"[3], "Transformers From Scratch"[4].

In general, Transformer performs self-attention with multiple heads and multiple layers. For a particular head, to compute self-attention, it derives the query Q, key K and value V from the input X itself as follows:

$$Q = W_q X \quad K = W_k X \quad V = W_v X \tag{A.1}$$

Using the derived Q, K, V triplet, it assigns new values to each input X using attention A given by

$$A(Q, K, V) = \text{SoftMax}\left(\frac{QK^T}{\sqrt{d_z}}\right) V \tag{A.2}$$

Here Q, K, V are each of shape $B \times T \times d_z$ where B is the batch size, T is the sequence length, and d_z is the dimension of each vector. The attention A can be computed efficiently using batch matrix multiplication since the multiplication QK^T and the subsequent multiplication with V have the common $B \times T$. For instance, when computing QK^T we perform batch matrix multiplication with $B \times T \times d_z$ and $B \times d_z \times T$ resulting in B matrix multiplications to give $B \times T \times T$ matrix.

Since the attention mechanism itself doesn't encode the positions of the individual T vectors, it is insensitive to the order of the T inputs. To address this, a position encoding is added to each of the T inputs to make the transformer dependent on the order of inputs. [54] follows up by using an additional relative position encoding. They define two new matrices $a^K_{i,j}$and $a^V_{i,j}$ (both of shape $B \times T \times T \times d_z$) and change the attention equation as follows:

$$A(Q, K, V) = \text{SoftMax}\left(\frac{Q(K^T + a^K_{i,j})}{\sqrt{d_z}}\right)(V + a^V_{i,j}) \tag{A.3}$$

As [54] notes, this removes the computation efficiency in the original transformer due to computation of $a^K_{i,j}$ for all pairs, and more importantly, the efficient batch matrix multiplication cannot be used due to addition of $a^K_{i,j}$ to K making it of shape $B \times T \times T \times d_z$. To resolve this, they propose the following equivalent formulation for computing QK^T (similarly for multiplying V):

$$Q(K^T + a^K_{i,j}) = QK^T + Qa^K_{i,j} \tag{A.4}$$

Such formulation removes the additional time to compute $K + a^K_{i,j}$ which would otherwise be a major bottleneck.

There are two related challenges in adopting it to the visual domain: (i) the positions are continuous rather than discrete (ii) both $a^K_{i,j}$ and $a^V_{i,j}$ have d_z dimension vector which is highly over-parameterized version of the $5d$ position vector ($d_z \gg 5$). To address (i) we use a $\mathcal{M}_p$ (MLP) to encode the $5d$ position which is a reasonable way to encode continuous parameters. For (ii) we change Eq. A.3 as

$$A(Q, K, V) = \text{SoftMax}\left(\frac{QK^T + \Delta}{\sqrt{d_z}}\right) V \tag{A.5}$$

Here Δ is of shape $B \times T \times T$ same as QK^T and Δ is computed from the relative positions of two object proposals p_i, p_j as $\Delta_{i,j} = \mathcal{M}_p(p_i - p_j)$ is a scalar. For added flexibility, we have that $\Delta_{i,j} \in \mathbb{R}^{n_h}$ where n_h is the number of heads allowing us to use different Δ for different heads.

As mentioned in Section 3.3 (of the main paper), the computation of $p_i - p_j$ for every pair remains the major bottleneck of our proposed relative position encoding.

B. Dataset Construction

We derive ActivityNet-SRL from ActivityNet-Entities (AE) [75] and ActivityNet-Captions (AC) [29] (Section B.1), provide the train, valid, and test split construction and statistics (Section B.2), show the distribution of the dataset (Section B.3) and finally compare ActivityNet against other datasets with object annotations (Section B.4).

B.1. Constructing ASRL

We first use a state-of-the-art BERT [10] based semantic role labeling system (SRL) [55] to predict the semantic roles of the video descriptions provided in AC. For SRL system, we use the implementation provided in AllenNLP [15] [5]. It is trained on OntoNotes 5 [46] which uses PropBank annotations [42]. PropBank annotations are better suited for `Verb` oriented descriptions. The system achieves 86.4% on OntoNotes5. To ensure the quality, we randomly picked 100 samples and looked at the various labeled roles. We found a majority of these to be unambiguous and satisfactory. The few that were not found were removed by the following heuristics: (i) in a sentence like "Man is seen throwing a ball", we remove the "seen" verb even though it is detected because "seen" verb doesn't provide any extra information (ii) similarly we also remove single verbs like "is", "was" which are already considered when some other verb is chosen (iii) finally, in a small number of cases, no semantic-roles could be found, and such cases were discarded. In general, each description can contain multiple verbs, in such cases, we treat each verb separately. Table 1 shows this with an example.

[2] https://nlp.seas.harvard.edu/2018/04/03/attention.html

[3] http://jalammar.github.io/illustrated-transformer/

[4] http://www.peterbloem.nl/blog/transformers

[5] see https://demo.allennlp.org/semantic-role-labeling for a demo

Sentence: A woman is seen speaking to the camera while holding up various objects and begins brushing her hair.		
Verb	Semantic-Role Labeling	Considered Inputs
is	A woman [V: is] seen speaking to the camera while holding up various objects and begins brushing her hair.	x
seen	A woman is [V: seen] [ARG1: speaking to the camera while holding up various objects and begins brushing her hair]	x
speaking	[ARG0: A woman] is seen [V: speaking] [ARG2: to the camera] [ARGM-TMP: while holding up various objects] and begins brushing her hair .	A woman speaking to the camera while holding up various objects
holding	[ARG0: A woman] is seen speaking to the camera while [V: holding] [ARGM-DIR: up] [ARG1: various objects] and begins brushing her hair .	A woman holding up various objects
begins	[ARG0: A woman] is seen speaking to the camera while holding up various objects and [V: begins] [ARG1: brushing her hair]	x
brushing	[ARG0: A woman] is seen speaking to the camera while holding up various objects and begins [V: brushing] [ARG1: her hair]	A woman brushing her hair

Table 1. An example of applying semantic role labeling to the video description. Each verb is treated independent of each other and the verbs "is", "seen", "begins" are not considered. For all other verbs, the last column shows the considered input to the system

Once we have all the SRL annotated, we align them with the annotations of AE. This is non-trivial due to mis-match between the tokenization used by AE (which is based on Stanford Parser [38]) compared to the tokenization used in AllenNLP [15]. Thus, we utilize the Alignment function provided in spacy v2 [21] to align the tokens from the two systems. To provide bounding box information to each role, we look at the tokens within the boundaries of the semantic role, and if any of them has been assigned a bounding box, we mark the semantic-role groundable, and assign it the corresponding bounding box. Figure 2 shows the most common considered roles followed by Figure 3 depicting the most common roles which have a bounding box annotations (groundable roles). Note that a particular role could be considered multiple times, *e.g.* in Table 1 "A woman" is considered for each of the verbs "speaking", "holding", "begins", "brushing" *i.e.* some of the roles (in particular Arg0) are counted more than once. While some roles like ArgM-TMP and ArgM-DIR appear more often than ArgM-LOC (see Figure 2), the number of groundable instances for the latter is much higher as locations are generally easier to localize. Further, note that Verb doesn't refer to an object and hence doesn't have any corresponding bounding boxes.

After having matched the annotated semantic roles with the bounding box annotations from AE, we lemmatize the arguments and create a dictionary for efficient contrastive sampling (as described in Section 4.2 in the main paper). To obtain the lemmatized words, we use the object-name annotations given in AE which are themselves derived from stanford parser [38]. To lemmatize the verbs, we use the inbuilt lemmatizer in spacy [21].

B.2. Training, Validation, Test Splits

Once the roles and lemmatized words have been assigned, we need to create a train, validation and test splits.

Train: We keep the same train split as AC, AE, ActivityNet. This allows using activity classification networks like TSN [62] trained on ActivityNet.

Validation and Test: Creating the validation and test splits is non-trivial. Since the test split of AC is kept private, AE uses half of validation split of AC as its test split which is again kept private. Thus, we divide the existing validation set into two to create the validation and test set for ASRL (see Figure 1 for an illustration of deriving the splits, and Table 2 for the exact numbers).

Dividing the original validation set implies high miss-rate (*i.e.* not enough examples to sample contrastive examples). To address this, we allow contrastive sampling from the test set during validation and vice-versa during testing for more robust evaluation.

B.3. Dataset Distribution

Figure 4 highlights the distributions of the semantic-role-structures (*i.e.* the order of the semantic role labels) found in the sentences. It is interesting to note Arg0-Verb-Arg1 far outnumbers all competing structures. This also motivates the choice of considering k=4 videos at a time (if structure contains 3 roles, we can sample 3 more videos).

We look at the total number of lemmatized words in Table 3 and the most frequent (top-20) lemmatized words for each role with their frequencies: (i) Verb Figure 5 (ii) Arg0 Figure 6 (iii) Arg1 Figure 7 (iv) Arg2 Figure 8 .

The higher number of verbs (Table 3) shows the diversity in the caption and their distribution is reasonably balanced (Figure 5) which we attribute to the curation process of ActivityNet [4]. In comparison, Arg0 is highly unbalanced as agents are mostly restricted to "people". We also observe that "man" appears much more often than "woman"/"she". This indicates gender bias in video curation or video description. Another interesting observa-

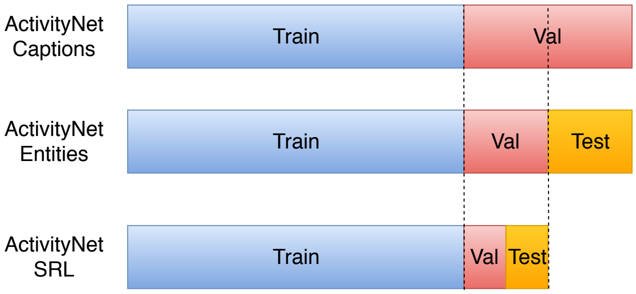


Figure 1. Train, val and test splits for AC, AE, ASRL.

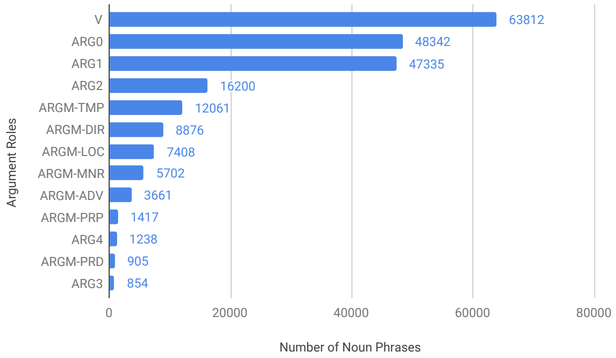


Figure 2. Number of Noun-Phrases for each role

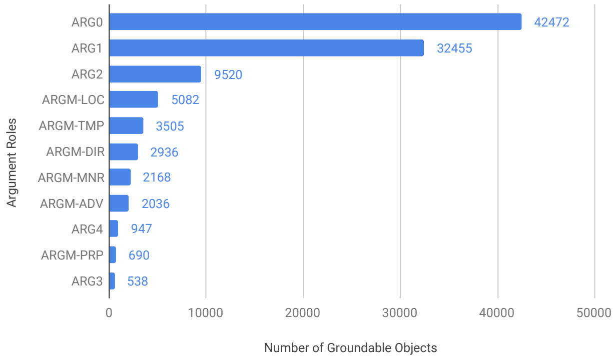


Figure 3. Number of groundable objects for each role

	Training	Validation	Testing
AC	37421	17505	
AE	37421	8774	8731
ASRL	31718	3891	3914

Table 2. Number of Videos in train, validation, and test splits. Some instances are removed from training if they don't contain meaningful SRLs. Our test split is derived from AE validation set.

tion is that "person" class dominates in each of argument roles which suggest "person-person" interactions are more commonly described than "person-object" interactions.

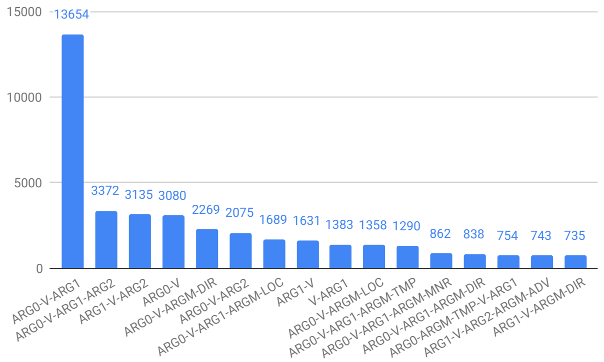


Figure 4. Frequently appearing SRL-Structures

V	Arg0	Arg1	Arg2	ArgM-LOC
338	93	281	114	59

Table 3. Total number of lemmatized words (with at least 20 occurrence) in the train set of ASRL.

B.4. Dataset Choice

Existing Datasets: (As of Nov 2019) Other than ActivityNet, there are three video datasets which have visual and language annotations in frames namely EPIC-Kitchens [9], TVQA+ [32] and Flintstones [18]. We consider the pros and cons of each dataset.

EPIC-Kitchens contains ego-centric videos related to kitchen activity. It provides object level annotations, with narrative descriptions. While the annotations are rich, the narrative descriptions are too short in length (like "open the fridge" or "cut the vegetable") and the actors `Arg0` are not visible (ego-centric).

TVQA+ is a question-answering dataset subsampled from TVQA [31] with additional object annotations. While the videos are themselves rich in human activities, the questions are heavily dependent on the sub-titles which diminishes the role of actions.

Flintstones is a richly annotated dataset containing clips from the cartoon Flintstones. The frames are 2-4 seconds long with 1-4 sentence description of the scene. With the objects in cartoons easier to identify it also serves as a diagnostic dataset for video understanding. However, the provided descriptions are less verb oriented and more image/scene-oriented due to shorter clips.

In contrast, ActivityNet contains longer videos, as a result the corresponding descriptions in ActivityNet Captions capture verbs over an extended period of time. While the object annotations are richer in EPIC-Kitchens, TVQA+ and Flintstones, the richer verb-oriented language descriptions make it more suitable for our task.

Using Natural Videos for evaluation: Our test data is generated "synthetically" by contrastive sampling followed by `SPAT` and `TEMP` strategies. An alternative evaluation protocol would be to test on naturally occurring videos. We discuss the challenges in obtaining such a dataset.

Recall that in our formulation of VOG a model needs to understand the relations among various objects prior to localizing them. For instance, to evaluate if a model understands "man petting a

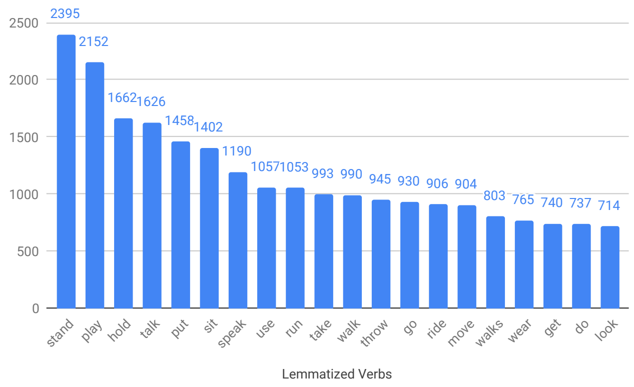


Figure 5. Top-20 Lemmatized `Verb`

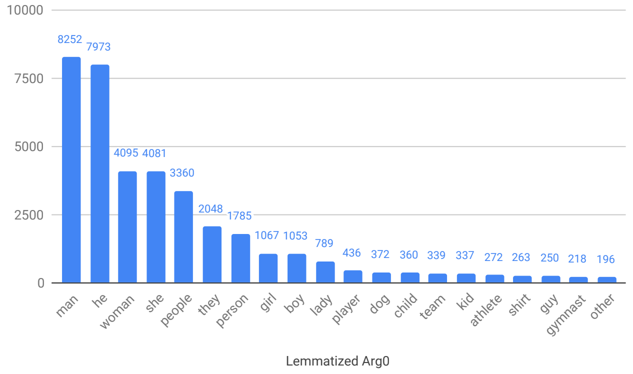


Figure 6. Top-20 Lemmatized `Arg0`

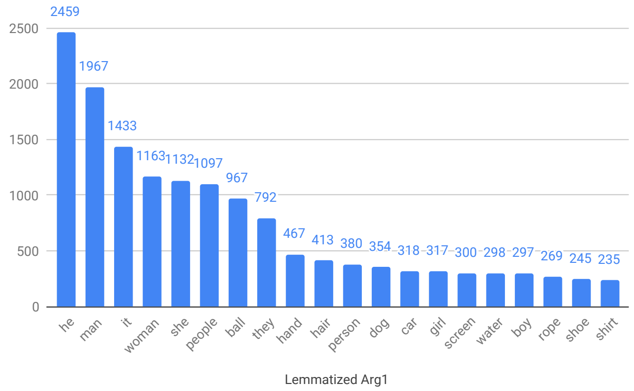


Figure 7. Top-20 Lemmatized `Arg1`

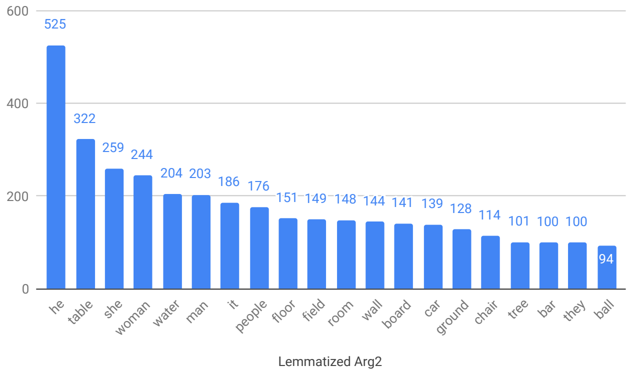


Figure 8. Top-20 Lemmatized `Arg2`

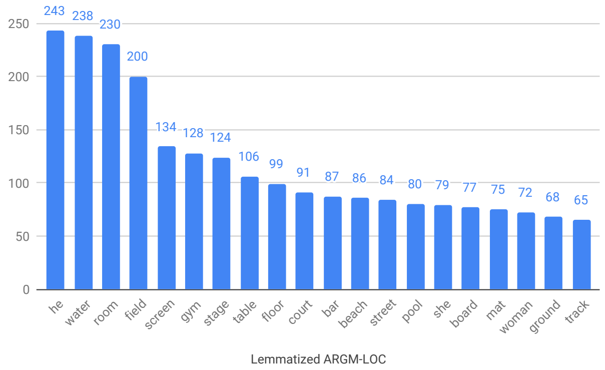


Figure 9. Top-20 Lemmatized `ArgM-LOC`

dog” (example from Fig 2a Q1,), we need contrastive examples Q2: “X petting a dog”,Q3: “man X a dog”,Q4: “man petting X” in the same video. In the absence of any of these examples, it is hard to verify that the model indeed understands to query. (*e.g.* without Q3, “man” and “dog” could be localized without understanding “petting”). Creating such a test set from web sources is impractical at present because there is no large-scale densely annotated video dataset to isolate such contrastive videos.

A different (and quite expensive) method would be crowdsourcing the video creation process by handing out detailed scripts to be enacted [56]. Here we would need to perform an additional “domain adaptation” step since we would be training and testing on different sources of videos (“YouTube” → “Crowd-Sourced”). This makes it challenging to attribute the source of error *i.e.* whether the reduced performance is due to poor generalization of object interactions or due to domain shift in the data.

In practice, `SPAT` and `TEMP` strategies when applied to contrastive videos from ActivityNet are effective proxies to obtaining naturally occurring contrastive examples from the web. This is validated by the drop from `SVSQ` to `SPAT` and `TEMP` (Table 3).

C. Evaluation

We use the following evaluation metrics:

1. Accuracy: correct box is predicted for the given phrase in a sentence (a sentence has multiple phrases)
2. Strict Accuracy: correct box is predicted for all the phrases in the sentence
3. Consistency: predicted boxes (for all the phrases) belong to the same video, even if they are incorrect
4. Video Accuracy: the predicted boxes are consistent, and the chosen video is also correct.

Since there is only one video in `SVSQ`, both consistency and

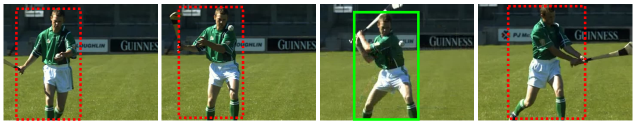


Figure 10. SVSQ: Illustration of the ground-truth annotations for the "man" (green) obtained from AE. The red boxes show equally correct boxes for "man" but are not annotated. As a result, we only consider the third frame to compute accuracy of the predicted box.

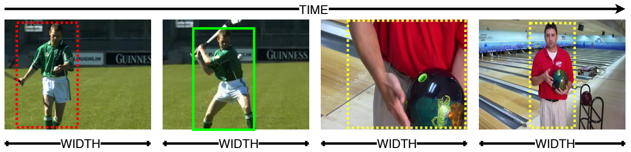


Figure 11. TEMP: Two videos are concatenated along the time dimension (we show 2 frames from each video) and with the description "man throwing a ball" and we are considering the object "man". If the predicted box is within the same video as ground-truth but the frame doesn't have any annotation (red box) we ignore it. However, if the predicted box belongs to another video (yellow boxes), we say the prediction is incorrect.

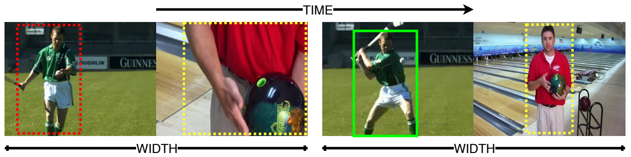


Figure 12. SPAT: Similar to previous case, we have the same description of "man throwing a ball" and we consider the object "man" but the videos are concatenated along the width dimension (we show 2 frames in the concatenated video). Again, if the predicted box lies in the same video as ground-truth (red box), we ignore it. If the predicted box is in another video (yellow boxes), the predictions are deemed incorrect.

video accuracy are not meaningful. Similarly, we first choose a video in SEP, it is trivially consistent.

As mentioned earlier, the bounding box annotations in AE is sparse, the object has a bounding box in only one frame in the video where it is most clearly visible. Since such sparse annotations complicate the computation of the above metrics, we describe their computation for each case.

C.1. Concatenation Strategies with Examples

SVSQ: We have a video with F frames, however, for each object, the bounding boxes are available in exactly one frame. Moreover, this annotated frame could be different for every object (the guideline provided in AE [75] is to annotate in the frame where it is most clearly visible). As a result, we cannot be sure if the same object appears in a frame where it is not annotated.

To address this, we require the model to predict exactly one bounding box in every frame. During evaluation, we consider only the annotated frame for a given object. If in this annotated frame, there is a predicted bounding box with $IoU \geq 0.5$, we consider the object correctly predicted as illustrated in Figure 10. This gives us Accuracy for SVSQ. Strict Accuracy can be easily computed by considering all the phrases in the query *i.e.* the predicted boxes for each phrase should have $IoU \geq 0.5$ with the ground-truth boxes.

SEP: We have k videos and we choose one of these k videos which gives us the Video Accuracy. If the chosen video is correct, we then apply scoring based on SVSQ otherwise mark it incorrect. Accuracy and Strict Accuracy computation is same as SVSQ.

TEMP: We have k videos concatenated temporally. In other words, we have kF frames in total of which we know $(k-1)F$ frames don't contain the queried object. Thus, if among the $(k-1)F$ frames not containing the queried object, if a predicted box has a score greater than a certain threshold, we mark it incorrect. For the F frames belonging to the queried video, we use the same method as for evaluating SVSQ. This is illustrated in Figure 11.

SPAT: In SPAT, we have k videos concatenated along the width axis. That is, we have F frames and each of width $kW \times H$ (here W, H are the width and height of a single video). In each of the F frames, there should not be a predicted box outside the boundaries of the correct video with a score greater than some threshold and for the annotated frame the predicted box should have $IoU \geq 0.5$. This is illustrated in Figure 12.

For TEMP and SPAT strategies, Consistency is computed by how often the various objects are grounded in the same video. Video Accuracy is derived from Consistency and is marked correct only when the correct video is considered. Finally, Strict Accuracy measures when all the phrases in the query are correctly grounded.

Selecting Threshold for Evaluation: As noted earlier, we pose the proposal prediction as a binary classification problem, if a proposal has a score higher than a threshold (hyper-parameter tuned on validation set), it is considered as a predicted box. For evaluation, we consider only the boxes which have the highest score in each frame. But in both SVSQ and SEP cases there is no incentive to set a threshold (>0), as the false positives cannot be identified in the same video. On the other hand, in both TEMP and SPAT cases, false positives can be identified since we are sure boxes in a different video are negatives.

D. Implementation Details

ImgGrnd is an image grounding system that considers each frame separately. It concatenates the language features to the visual features of each object which is then used to predict whether the given object is correct. More specifically, given $\tilde{q}_j$ (Eqn 1) and the visual features $\hat{v}_{i,j}$ we concatenate them to get the multi-modal features m_{IG} where $m_{IG}[l,i,j] = [\hat{v}_{i,j} || \tilde{q}_l]$. These are passed through a two-layered MLP classifier and trained using BCE Loss. In essence, ImgGrnd can be derived from VOGNet by removing the object transformer and the multi-modal transformer.

VidGrnd is a video grounding system which builds upon ImgGrnd. Specifically, it has an object transformer to encode the language-independent relations among the objects. More formally, given $\hat{v}_{i,j}$ we apply object transformer to get $\hat{v}^{sa}_{i,j}$. The remaining steps are the same as ImgGrnd. We concatenate the language features $\tilde{q}_j$ with the self-attended object features $\hat{v}^{sa}_{i,j}$ to get the multi-modal features m_{VG} where $m_{VG}[l,i,j] = [\hat{v}^{sa}_{i,j} || \tilde{q}_j]$. After passing through a 2 layer MLP classifier, it is trained using BCE Loss. In essence, VidGrnd can be derived from VOGNet by removing the multi-modal transformer altogether and the relative position encoding from object transformer.

Model	Train	SVSQ		SEP			TEMP				SPAT			
		Acc	SAcc	Acc	VAcc	SAcc	Acc	VAcc	Cons	SAcc	Acc	VAcc	Cons	SAcc
ImgGrnd	GT5	46.31	24.83	20.55	47.49	9.92	8.06	2.68	25.35	2.68	4.64	2.47	34.17	1.31
	P100	55.22	32.7	26.29	46.9	15.4	9.71	3.59	22.97	3.49	7.39	4.02	37.15	2.72
VidGrnd	GT5	43.37	22.64	22.67	49.6	11.67	9.35	3.37	28.47	3.29	5.1	2.66	33.6	1.74
	P100	53.30	30.90	25.99	47.07	14.79	10.56	4.04	29.47	3.98	8.54	4.33	36.26	3.09
VOGNet	GT5	46.25	24.61	24.05	51.07	12.51	9.72	3.41	26.34	3.35	6.21	3.40	39.81	2.18
	P100	53.77	31.9	29.32	51.2	17.17	12.68	5.37	25.03	5.17	9.91	5.08	34.93	3.59

Table 4. Comparing models trained with GT5 and P100. All models are tested in P100 setting.

VOGNet: Our models are implemented in Pytorch [43]. VOGNet SPAT using GT5 takes nearly 25-30 mins per epoch (batch size 4), compared to 3 hours per epoch for P100 (batch size 2). All models are trained for 10 epochs (usually enough for convergence). All experiments can be run on a single 2080Ti GPU.

Language Feature Encoding: We use a Bi-LSTM [20, 52] (fairseq [41] implementation). The words are embedded in $\mathbb{R}^{512}$ and the Bi-LSTM contains 2 layers with hidden size of 1024, max sequence length of 20, and $\mathcal{M}_q$ with input/output size of 256.

Visual Feature Encoding: The object features are obtained from a FasterRCNN [47] with ResNext [63] pre-trained on Visual Genome [30]. Each object feature is 2048d vector. The image level features (2048d) and optical flow (1024d) are extracted using resnet-200 [19] and TVL1 [69] respectively and are encoded using temporal segment networks [62]. They are concatenated to give segment features for each frame which are 3072d vector. We project both object and segment features into 512d vectors and then concatenate them to get 1024d vector for each object.

Object Transformer uses 3 heads and 1 layer with each query, key, value of 1024d (full feature dimension which is divided by number of heads for multi-headed attention).

Multi-Modal Transformer also uses 3 heads and 1 layer but the query, key, value are 1280d vectors (additional 256 due to concatenating with the language features).

E. Additional Experiments

We perform two additional experiments: (i) if the representations learned in GT5 transfer to the more general case of P100 (ii) the effect of adding more heads and layers to the object transformer (OTx) and multi-modal transformer (MTx).

GT5 models in P100 setting: In Table 4 we compare the models ImgGrnd, VidGrnd, and VOGNet trained in GT5 and P100 and tested in P100 setting to calculate the transfer-ability of GT5 setting. While testing in P100, for TEMP and SPAT, we set the threshold for models trained in GT5 as 0.5 which is higher than the threshold used when testing in GT5 (0.2). This is expected as a lower threshold would imply a higher chance of a false positive.

In general, the drop from P100 to GT5 is significant (a $15-25\%$ drop) for almost all models suggesting training with just ground-truth boxes is insufficient. Nonetheless, since the relative drops are same across models, GT5 remains a valuable proxy for carrying out larger number of experiments.

SPAT	Acc	VAcc	Cons	SAcc
ImgGrnd	17.03	9.71	50.41	7.14
+OTx (1L, 3H)	19.8	10.91	48.34	8.45
+OTx (2L, 3H)	20.8	11.38	49.45	9.17
+OTx (2L, 6H)	**21.16**	**12.2**	48.86	**9.58**
+OTx (3L, 3H)	20.68	11.34	48.66	9.19
+OTx (3L, 6H)	21.14	12.1	**49.66**	9.52
VOGNet	23.53	14.22	56.5	11.58
+MTx (2L,3H)	23.38	14.78	55.5	11.9
+MTx (2L,6H)	23.96	14.44	55.5	11.59
+MTx (3L,3H)	24.53	14.84	56.19	12.37
+MTx (3L,6H)	24.24	15.36	57.37	12.52
+OTx(3L,6H)	**24.99**	**17.33**	**66.29**	**14.47**

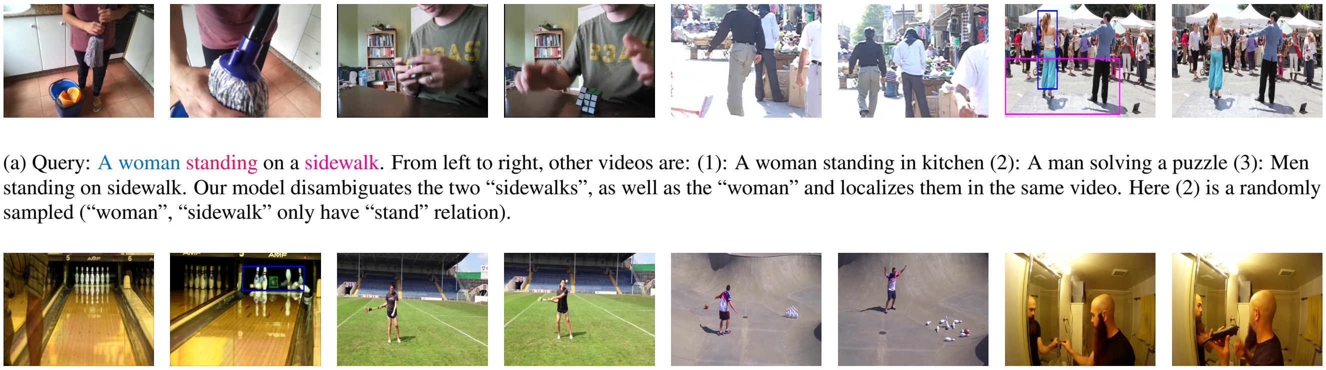
Table 5. Ablative study layers and heads of Transformers.

Transformer Ablation: In Table 5 we ablate the object transformer and the multi-modal transformer with number of layers and heads. It is interesting to note adding more heads better than adding more layers for object transformer, while in the case of multi-modal transformer both number of heads and number of layers help. Finally, we find that simply adding more layers and heads to the object transformer is insufficient, as a multi-modal transformer with 1 layer and 3 heads performs significantly better than the object transformer with 3 layers and 6 heads.

F. Visualization

In general, contrastive examples differ in exactly one part of the phrase. However, we observed that some contrastive examples were very difficult to distinguish. We identify two reasons: (i) Considering only one verb in the query becomes restrictive. For instance, in Figure 13-(b) video (3), the complete description has “the bowling ball that goes around the ring and then hits the pins” and the initial part of it going around the ring is lost. (ii) Language ambiguity of the form “person playing guitar” vs “person practicing guitar”, while “playing” and “practicing” have distinct meanings, in some situations they can be used interchangeably.

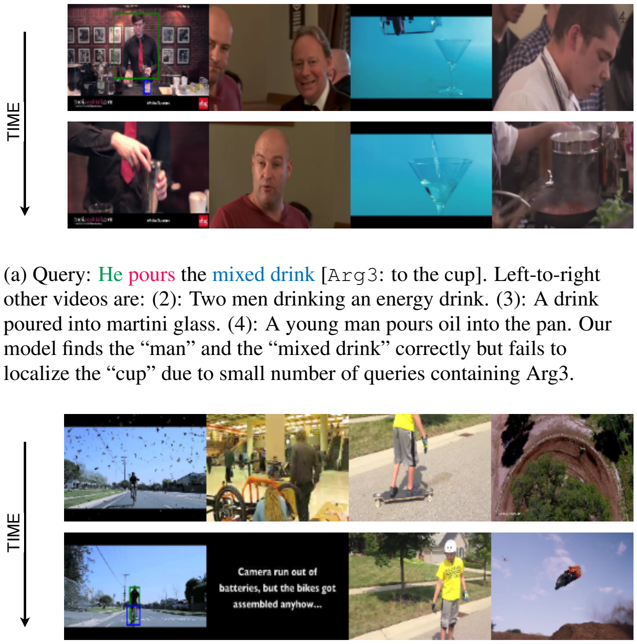
We now visualize a few examples for TEMP and SPAT in Figure 13, 14, 15. All visualizations are obtained using VOGNet trained in GT5 setting. For each case, we show 2 frames from each video and color-code the arguments in the given query (Arg0



(a) Query: A woman standing on a sidewalk. From left to right, other videos are: (1): A woman standing in kitchen (2): A man solving a puzzle (3): Men standing on sidewalk. Our model disambiguates the two "sidewalks", as well as the "woman" and localizes them in the same video. Here (2) is a randomly sampled ("woman", "sidewalk" only have "stand" relation).

(b) Query: The ball hits the pins creating a strike. From left to right, other videos are: (2): The girl with the ball hits it (3): A bowling ball hits the pins. (4): He uses razor to trim. While our model correctly chooses the correct frame, we note (3) is very close to (1) in terms of description. Here, our sampling method fails by providing "too" similar videos.

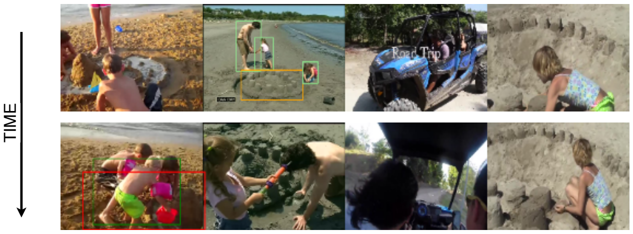
Figure 13. VOGNet predictions `TEMP` strategy in GT5 setting. We show two frames from each video, but the model looks at $F{=}40$ frames.


TIME
TIME
Camera run out of batteries, but the bikes got assembled anyhow...

(a) Query: He pours the mixed drink [`Arg3`: to the cup]. Left-to-right other videos are: (2): Two men drinking an energy drink. (3): A drink poured into martini glass. (4): A young man pours oil into the pan. Our model finds the "man" and the "mixed drink" correctly but fails to localize the "cup" due to small number of queries containing Arg3.

(b) Query: The man riding the bike. Left-to-right other videos are: (2): Men put the other bike down. (3): A boy rides his skateboard. (4): We see the boy riding his dirtbike. Here, our model correctly distinguishes among the bikes, and who is riding what.

Figure 14. VOGNet predictions `SPAT` strategy in GT5 setting. We show two frames from each video, and each frame contains 4 videos concatenated together.



Query: A man and two kids building a sand castle. Left-to-right other videos: (1): A group of kids trying to build a sand castle. (3): They are driving through the sand. (4): She is building a castle. In the first frame, the ground-truths are marked in light-green and orange and in the second frame is our model's incorrect prediction. It is unable to distinguish based on "man" due to influence of "kids" in the agent.

Figure 15. Incorrect prediction of VOGNet for `SPAT` strategy

is Green, Verb is Red, Arg1 is Blue, Arg2 is Magenta) Remaining arguments are mentioned in the query (like in Figure 14 (a)).

For `TEMP`, since objects are not being considered independent of each other, the model doesn't ground objects which are present in the query but not related. For instance in Figure 13-(a), even though "woman" and "sidewalk" are separately present in two other videos, these are given very low score. Similarly, in Figure 13-(b), "ball" in (2) is not grounded which is not related to the query. These suggest VOGNet is able to exploit the cues in the language query to ground the objects and their relations in the visual domain.

For `SPAT`, in Figure 14-(a) our model finds the correct video. It is able to differentiate among someone pouring drink into a glass (2), someone pouring oil (3), or someone drinking the drink (1). However, it is unable to find the "cup" which we attribute to the smaller number of examples containing `Arg3` which is limited to verbs like "pour". In Figure 14-(b) our model correctly finds both "man" and the "bike" that he is riding and distinguishes between "ride" and "put", "bike" and "skateboard" (3).

Finally, in Figure 15, we find the language ambiguity of "trying to build" and "building" which are synonymously used. While our model is able to distinguish (4) by its agent "she" compared to "man and two kids", it is unable to make the distinction between "a man and two kids" and "a group of kids" (1). We attribute this to the use of a single embedding for each role (here Arg0) and not differentiating among the various objects in that role.