Multi-Moments in Time: Learning and Interpreting Models for Multi-Action Video Understanding

Mathew Monfort[13], Kandan Ramakrishnan[3], Alex Andonian[1], Barry A McNamara [1], Alex Lascelles [1], Bowen Pan[1], Dan Gutfreund[23], Rogerio Feris[23], Aude Oliva[13]
[1] MIT CSAIL, [2] IBM Research, [3] MIT-IBM Watson AI Lab

arXiv:1911.00232v3 [cs.CV] 10 Jan 2020

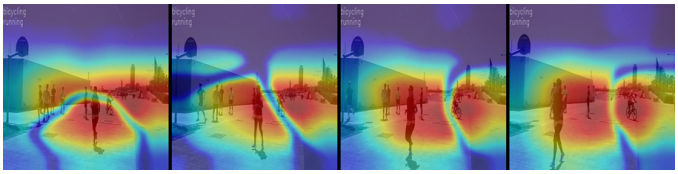
Abstract—An event happening in the world is often made of different activities and actions that can unfold simultaneously or sequentially within a few seconds. However, most large-scale datasets built to train models for action recognition provide a single label per video clip. Consequently, models can be incorrectly penalized for classifying actions that exist in the videos but are not explicitly labeled and do not learn the full spectrum of information that would be mandatory to more completely comprehend different events and eventually learn causality between them. Towards this goal, we augmented the existing video dataset, Moments in Time (MiT), to include over two million action labels for over one million three second videos. This multi-label dataset introduces novel challenges on how to train and analyze models for multi-action detection. Here, we present baseline results for multi-action recognition using loss functions adapted for long tail multi-label learning and provide improved methods for visualizing and interpreting models trained for multi-label action detection.

1 Introduction

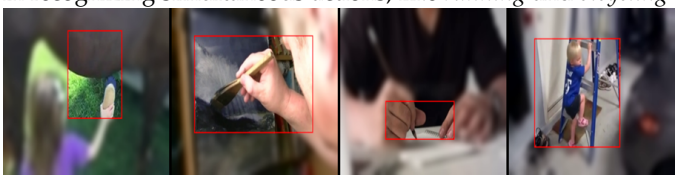
Events depicted in videos are often made of several actions that may occur simultaneously or sequentially, like the giant panda climbing, trying to hang out on a branch and then falling off the tree (Fig. 1.a). To capture the natural richness of the dynamic world, models must be able to learn actions that occur in a sequence, as well as be able to spot different unrelated events co-occurring in space (Fig. 1.b). Describing these events with a single labeled activity would be incomplete and would miss a large amount of useful information for video comprehension.

Several large-scale video datasets provide a large diversity and coverage in terms of the categories of activities and exemplars they capture [19], [12], [26]. However, these labeled datasets only provide a single annotated label for each video and this label may not cover the rich spectrum of events occurring in the video. For example a video of an audience *applauding* may also include a person on a stage *performing*, *playing music*, *singing*, *dancing*, etc. While the visual, audio and semantic complexity of each video is challenging to fully annotate, we took a step toward this goal by extending the Moments in Time dataset [26], to contain multiple action labels describing one or more visual or audio events occurring in each clip.

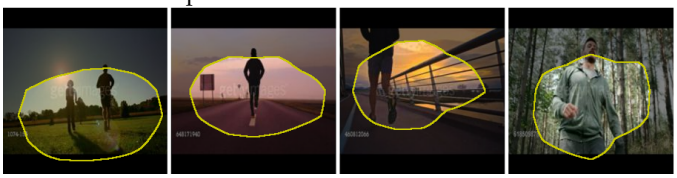
(a) **Multi-Moments in Time** 2 million action labels for 1 million 3 second videos



(b) **Multi-Regions** Localizing multiple visual regions involved in recognizing simultaneous actions, like *running* and *bicycling*



(c) **Action Regions** Spatial localization of actions in single frames for network interpretation



(d) **Action Concepts** Interpretable action features learned by a trained model (i.e. jogging)

Building a multi-label dataset introduces new challenges in how to train and analyze models for multi-label action detection, including loss function to optimize model learning such as to take advantage of distinct labels for the same visual inputs. For model's representation analysis, recently

introduced methods such as Class Activation Mapping (CAM) [46] and Network Dissection (NetDissect) [3] focused on single label interpretation (CAM) or did not provide methods for analyzing learned action concepts (NetDissect). Here, we address these limitations by extending both of these approaches to multi-action models (see examples in Fig.1.b-d).

In Section 5.1 we present a multi-label extension to Class Activation Mapping that identifies the important image regions for predicting multiple simultaneous actions in a given scene. In Section 5.2 we outline our approach for adding action concepts to the NetDissect framework for interpreting internal units of a deep network. We additionally examine different multi-label loss functions applied to our dataset in Section 7. This includes a modification to the recently introduced LSEP loss function [23] to support weighted learning for unbalanced datasets to handle the natural long tail distribution of actions in video. In the next section, we describe related work in the area followed by a description of the Multi-Moments in Time dataset and our annotation procedure. Overall the key contributions of this paper include:

- **Multi-Moments in Time (M-MiT)**: A large-scale multi-label action dataset for video understanding with over two million action labels [1].
- **wLSEP**: A novel multi-label loss function that supports learning from an unbalanced class distribution where some classes have more examples than others.
- **mCAM**: Multi-Label Class Activation Mapping for identifying multiple important visual features for model predictions.
- **Action Network Dissection**: We present a single frame dataset (Action Boxes) with bounding boxes on visible actions that we use for incorporating **action concepts** into Network Dissection [3] to identify key interpretable features learned by action recognition models.

2 Related work

2.1 Video Datasets and Models

Many video datasets are available to test models of action recognition or detection, including Hollywood2 [22], LabelMe video [40], UCF101 [31], HMDB51 [21], THUMOS [18], AVA [13], "something something" [12] and Charades [29]. Training deep neural networks for these tasks requires available large video datasets, like ActivityNet [6], Kinetics [19], Moments in Time [26], or YouTube-8M [1].

To perform various tasks of video understanding, deep convolutional neural networks [20], [32], [37], [14], [17] have been combined with optical flow [16] to capture temporal dynamics [30], [7], [43]. A popular architecture is the two-stream CNNs [30] which separately process optical flow and RGB frames. Spatial-temporal information can also be directly modeled using 3D convolutions [33] which can be "inflated" from 2D filters (I3D) [7] in order to take advantage of the strong features learned by pre-training a network on ImageNet [9]. More recent 3D networks incorporate non-local modules [35] in order to capture long-range dependencies. Temporal Relation Networks [44] take a different approach by learning the relevant state transitions in sparsely sampled frames from different temporal segments.

1. The data is available on our site, http://moments.csail.mit.edu.

2.2 Multi-Label Optimization

Predicting multiple labels for the same input has not been attempted for action recognition, but the task is increasingly popular for predicting multiple objects present in the scene (e.g. predicting the presence of a person, a TV and a table in the same image). Given the variety of appearances of objects within a scene, it is challenging for global CNN features to correctly predict multiple labels. Approaches instead use object proposals [11], [38] or learn spatial co-occurrences of labels using LSTMs [34], [41]. Convolutional neural networks (CNN) have also been applied on raw images of multiple objects to learn image-level deep visual representations for multi-label classification [34].

To learn to optimize training for multiple labels, different loss functions have been proposed. A common approach is binary cross entropy which optimizes each class label individually. However, when treating each class individually, it is also difficult to learn the correlations between different classes [10], [42], [23]. Indeed, this approach may incorrectly penalize some examples which do not have full label coverage as it assumes the absence of a label is a negative label.

Another approach is pair-wise ranking [36] which encourages the model to generally assign higher ranks to positive labels. This method has the added benefit of reducing the strength of a models' mistake as incorrect predictions tend to still include highly ranked positive labels. WARP [11], [36] expands on this by including a monotonically increasing weighting function that increases the error for positive labels that are poorly ranked, thus prioritizing them in learning. BP-MLL [42] is another approach that provides a smooth calculation of the ranking error but suffers from exceedingly large values when the positive classes are poorly ranked and the vocabulary size is large. LSEP [23] is a variation of the BP-MLL loss function which addresses it's numerical stability issues but can become dominated by poorly ranked positive classes.

3 A Multi-Action Dataset

Our goal with this project is to extend the existing Moments in Time dataset to include multiple action labels per video to take another step toward improving our understanding of the diverse and dynamic events that take place in short videos. With this multi-label dataset we will be able to better evaluate our existing models and train new models that can better leverage the amount of information in each video as well as the relationships between different actions.

3.1 Annotation

We began by annotating our added action labels using the same process as the Moments in Time dataset. The annotation phase used Amazon Mechanical Turk for crowd sourcing where each worker is presented with a video-verb pair and asked to respond Yes or No if the action is either seen or heard in the video (see [26] for more details).

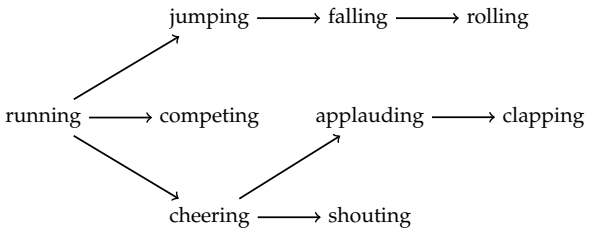


Fig. 2: An example of the path of generating new candidate verbs from previously annotated classes.

3.1.1 Generating Action Candidates

A difficulty of the annotation task is to choose candidate actions that are likely to return positive responses (i.e. actions that occur in the videos). We choose to generate candidate actions for each video using different techniques.

A first candidate generation method was based on WordNet [25] relationships where we iteratively picked actions for annotation that were closely linked in the WordNet semantic graph to the existing action labels for each video. We constrained ourselves to the original Moments in Time vocabulary of 339 actions in order to simplify this process and stopped our candidate selection when the harvest rate (positive worker response rate) dropped below 20%. For example, if we have a video with the action *running*, we find a strong semantic similarity to the action *sprinting* and begin annotating all videos that have the action *running* with the action *exercising*. We do this until the harvest drops below the threshold and then we no longer annotate videos that have *running* with *exercising*. It is important to note that the hierarchy in action relationships are less well defined than that of objects. For example, not every video of *eating* is defined by the action *dining*. Similarly there are videos in the dataset of animals *chasing* each other which do not relate to *exercising* in the same way a video of a person *jogging* does.

This method candidate generation allows to efficiently annotate multiple actions in each video, but it does result in videos where we do not annotate every present action. We consider this an acceptable trade-off for the ability to scale to such a large set of labels and we ensure that in training models we do not directly penalize predictions for actions that are not labeled (see Section 4). Similar to the WordNet approach, we use Word2Vec [24] similarity scores to select action candidates based on the existing action labels for a video. Word2Vec allows us to generate candidates for actions that describe commonly co-occurring events such as *stirring* and *boiling* or *running* and *jumping*.

For each of these methods we regenerate new candidates as new labels are verified through annotation. Figure 2 outlines the candidate generation path of a video that was originally only labeled with the action *running*. In this case we first generate and annotate the action candidates *jumping*, *competing* and *cheering* which then eventually lead to subsequent annotated actions such as *clapping* and *rolling* such that we are able to provide a much more thorough description of the actions taking place in the video than simply *running*. Also note that *cheering*, *applauding*, *clapping* and *shouting* are auditory actions that may not be visible in the video it self but instead heard.

The final approach we use for candidate selection is to run a trained model over the videos and select the top-1 predictions for each video that are not currently annotated. This model used for this task was trained on the single label dataset and is provided by the authors of the original work [26].

3.2 Dataset Statistics

There are 802,244 video-label pairs in the training set, and 33,900 in the validation set of the Moments in Time Dataset. We increased this dataset to 2.01 million labels for 1.02 million videos by adding new videos, generating and annotating action candidates as described in the previous section and adding new action classes to the dataset. Our *Multi-Moments in Time* dataset includes 313 annotated action classes where new actions have been added to the Moments in Time vocabulary (e.g. *skateboarding*), ambiguous actions were removed (e.g. *working*) and similar actions have been merged into a single class (e.g. *rising* and *ascending* -> *ascending/rising*). In total we merged 37 distinct classes into 20 action clusters and added 22 novel actions not found in the original Moments in Time (e.g. *unplugging*). We additionally remove 31 actions that we deemed either too vague (e.g. *working*) or noisy (e.g. *fencing*). This new vocabulary should cover an increased breadth of events while improving the boundary between different classes. Using this new action set we were able to increase the training set to include over 2 million labels where 553,535 videos are annotated with more than one label and 257,491 videos are annotated with three or more labels. In addition, we have created new validation and test sets each consisting of 10K videos with over 30K labels each.

4 Multi-Label Loss Functions

In this section we present a set of multi-label loss functions that we use to train models on our multi-action dataset. For each loss we normalize by the number of labels in each data example to handle the variability in the number of labels per video and incorporate a class weighting term w_i that helps to balance learning when the training set has an unbalanced number of examples per label. This unbalance is common for action datasets as action labels tend to follow a long tail distribution in practice. Additionally, in prior work sampling was used to address the quadratic complexity of the different loss functions [23]. However we have found that parallelizing the loss computation through matrix operations eliminates the need for sampling. We compare the results of each approach in Section 7.

4.1 BCE

A common a approach to multi-label optimization is to optimize for binary cross entropy and treat each label as an independent classifier,

$$\mathcal{L}_{BCE} = -w_i[y_i \log x_i + (1 - y_i) \log(1 - x_i)]. \quad (1)$$

However, this has been shown to not consider correlations between different classes [10], [42], [23] and makes the assumption that cases where a class does not have a positive label are negative examples. While we have verified all positive labels in the proposed dataset, we do not assume that an unlabeled class is guaranteed to not be present.

4.2 WARP

Pair-wise ranking presents another approach to multi-label optimization and stems from the motivation that while it is important to correctly classify a positive label, it is also important to reduce the strength of a mistake by encouraging the model to generally assign higher ranks to positive labels [36]. The WARP loss function [11], [36],

$$\mathcal{L}_{WARP} = \frac{1}{|\mathcal{Y}|} \sum_{i \in \mathcal{Y}} w_i \mathcal{W}(\mathcal{R}(x_i)) \sum_{j \notin \mathcal{Y}} \max(0, 1 + x_j - x_i), \tag{2}$$

proposes a weighted pair-wise ranking function that prioritizes poorly ranked positive classes via an additional monotonically increasing weighting function $\mathcal{W}(x_i)$. The rank of a predicted class, $\mathcal{R}(x)$, is used to increase the error penalization for lower ranked classes. In practice we set $\alpha = 1$ and $\mathcal{W}(r) = \sum_{k=1}^{r} \frac{1}{k}$. The limitation of WARP lies in the non-smooth nature of the loss function which can make it difficult to optimize.

4.3 LSEP

The recently proposed multi-label ranking function LSEP [23] which takes the log of the BP-MLL function [42], with the addition of a single bias term, to increase the numerical stability,

$$\mathcal{L}_{LSEP} = \log\big(1 + \sum_{i \in \mathcal{Y}} \sum_{j \notin \mathcal{Y}} e^{x_j - x_i}\big). \tag{3}$$

This prioritizes positive classes that are poorly ranked without the need of an additional weighting function as in WARP.

wLSEP

It is not straightforward to add a weight balancing term to the LSEP loss as the gradient computation for each class is non-separable from other classes. We address this by modifying the LSEP loss to apply individually to each class with a positive label allowing us to simply add a weight term, w_i, to the function,

$$\mathcal{L}_{wLSEP} = \frac{1}{|\mathcal{Y}|} \sum_{i \in \mathcal{Y}} w_i \log\big(1 + \sum_{j \notin \mathcal{Y}} e^{x_j - x_i}\big). \tag{4}$$

This looses the prioritization of low ranked positive classes in the original loss function by summing the softmax of the ranking error $(x_j - x_i)$ for each positive label i, however it avoids the problem of taking a global softmax over all positive labels which can be dominated by positive labels that are ranked poorly rather than scalably weighting the ranks as in WARP.

5 Analyzing Multi-Label Models

5.1 Multi-Label Class Activation Mapping

Recent work in network interpretation performs a weighted inflation of the feature maps in the final convolutional layer to visualize the important visual regions the network is using to make a prediction [45]. These class activation maps (CAMs) can be thought of as a method for visualizing the learned attention model of the network and have been shown to localize actions occurring in videos [26].

To extend this technique to multi-label tasks the simplest approach is to inflate the CAM for each predicted action, taking the maximum value of each map so that we can visualize the discriminative features used for our multi-label prediction. However, in practice, the combined CAM filters cover a large area of the input image making it difficult to gain a useful understanding of the models decisions. To address this issue, before we take the maximum value of each actions CAM, we first compare the different CAM filters for each of our predicted classes and when two filters have similar values at the same pixel locations we set the values of the pixels for both filters to zero. This eliminates the overlapping area and, after performing maximum pooling over the altered CAM filters, creates a boundary between the distinct features associated with each predicted action. To improve the visualizations we also apply Gaussian smoothing with a 5x5 kernel to reduce the sharpness of some of the boundary edges. We show the results of this method for analyzing multi-label predictions in Section 8.2.

5.2 Identifying Learned Action Features

Network interpretation has recently been extended to not just visualize important visual regions for a prediction but to also identify the different concepts a network has learned. This is important for understanding the representation of the network and diagnosing class biases that the network is learning. However, the previous work in network dissection (NetDissect) [3] does not include action concepts in its interpretation instead relying on objects, scenes, object parts, textures and materials.

To better analyze our learned action models we extend the set of concepts used by NetDissect to include actions. To do this we first build an image segmentation dataset for actions.

5.2.1 Annotation

As with annotating the action labels in the videos, we collect bounding box annotations via Amazon Mechanical Turk (AMT). We begin by selecting a single frame from the center of 500 randomly selected videos from each of the action classes in Moments in Time [26], Kinetics [19] and the Something-Something [12] datasets. We then present a binary annotation task to the workers on AMT asking if an action from the source videos label set is visible in the frame shown. This binary interface is very similar to that used for collecting the action labels for the Moments in Time dataset [26] with the main difference being the use of images rather than video. We run this task for at least 2 rounds of annotation to verify that the action is visible in each frame. We then take the set of verified action-frame pairs and pass them to a separate annotation interface on AMT that asks the workers to select the regions most important in the image for identifying the action. Multiple regions can be selected for an image, as in the jogging example in Figure 5, and the workers are allowed to skip an image if there are no useful regions for detecting the action (i.e. the action is not visible in the image).

We run this region selection task through multiple rounds and only consider overlapping regions from the different

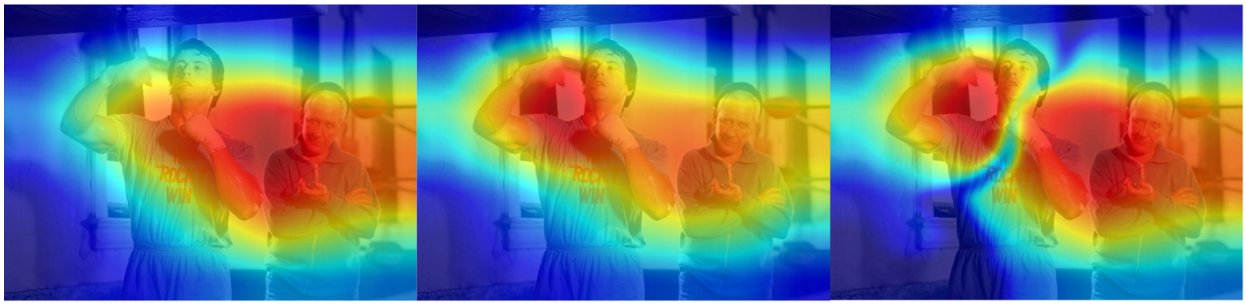


Fig. 3: **Region Separation:** Example showing single class CAM images and the separation of relevant features in a multi-class CAM image. The red regions specify important areas of the image used by the model to infer the detected action.

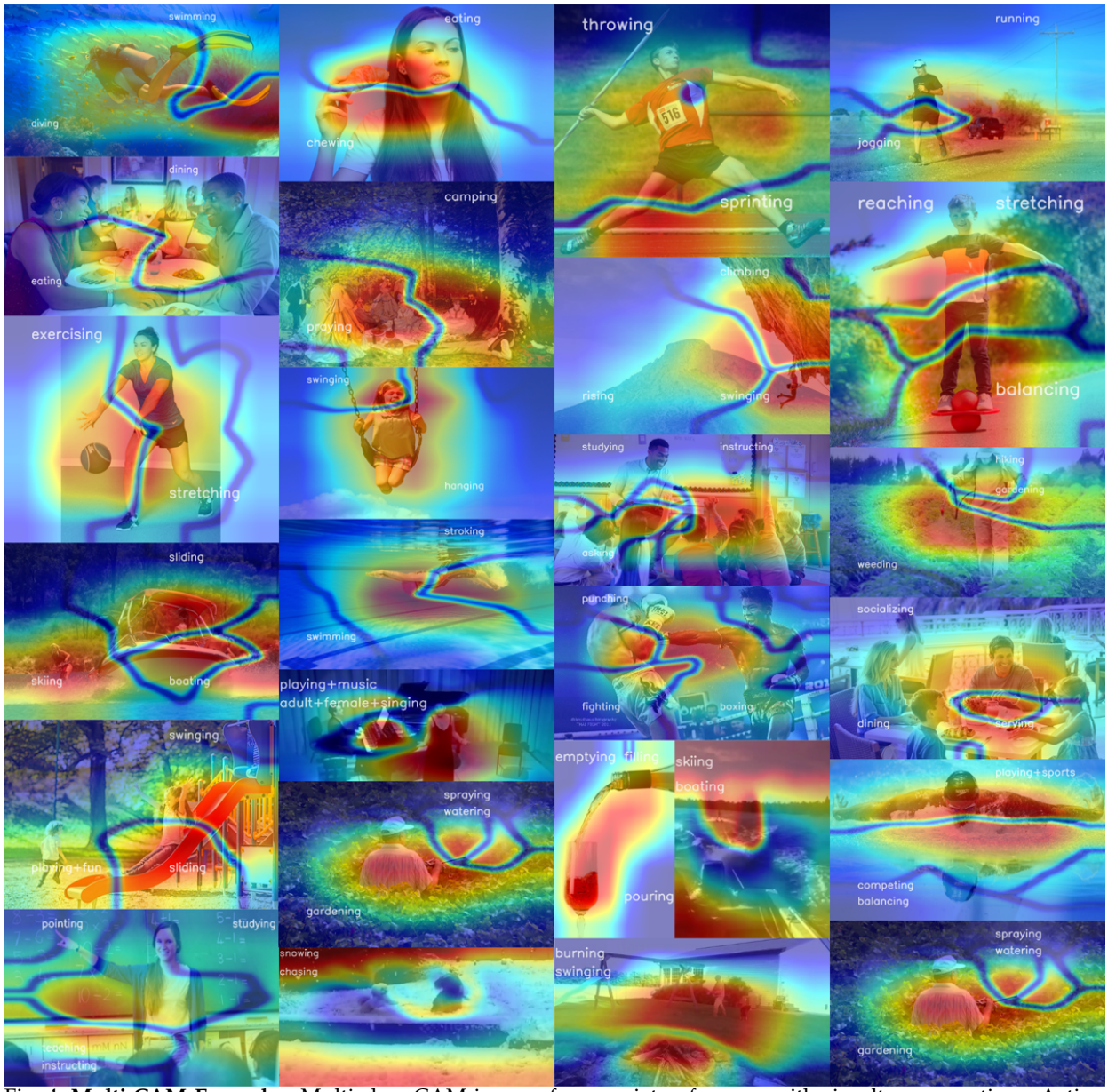


Fig. 4: **Multi-CAM Examples:** Multi-class CAM images for a variety of scenes with simultaneous actions. Action labels are placed near the important image regions used by the model for identifying each specific action. This area is signified by the red overlay with blue edges separating image regions distinct to each detected action showing that our model is able to localize multiple actions present in each scene despite not being trained for localization.

rounds as most important for detecting the actions. After this stage the regions selected are cropped from the original images and passed through the binary annotation task previously described for a final verification that the actions are present and recognizable in the selected regions. After our complete annotation process our total set of verified images with segmented action regions consists of 67,468 images from 549 different action classes. Figure 5 displays some examples of the selected regions collected through this process.

5.2.2 Learned Action Interpretation

To integrate our new action region dataset into the Network Dissection framework we first consider each selected region to be a mask on the segmented area of the image relating to the action. This is similar to part, material and object masks used for other segmentation datasets [47], [8], [28], [4]. With the data formatted in this manner we extend the Broden dataset to include our action segmentations and extract the set of learned action concepts detected via NetDissect. This process allows us to identify not just object, scene, texture and color concepts learned by our models, but action concepts as well. In Section 8.3 we show some of the key results from interpreting action networks in this way.

6 Baseline Results

We trained two architectures on our dataset, an inflated 3D Resnet50 (I3D) [7] for the visual modality and a SoundNet network [2] for learning audio features and compare across different multi-label loss functions.

6.1 Models

Inflated 3D Convolutional Networks (I3D)

I3D networks offer improved weight intiailization by simply *inflating* the convolutional and pooling kernels of pretrained 2D networks [7]. This is done by initializing the inflated 3D kernel with pretrained weights from 2D models by repeating the parameters from the 2D kernel over the temporal dimension. This greatly improves learning efficiency and performance since 3D models contain a large number of parameters and are difficult to train from scratch. For our experiments we use an inflated 3D resnet50 pretrained on Imagenet.

SoundNet Network (Audio)

In the Moments in Time dataset action classes are labeled for both visual and auditory information. Therefore we feel it would be incomplete to evaluate visual models and not include a model trained on audio. We finetune a SoundNet network [2] which was pretrained on unlabeled videos from Flickr.

Spatio-temporal-Auditory Fusion

We fuse the predictions from the audio and visual modalities by concatenating the spatio-temporal features of the I3D network with the auditory features from SoundNet and train a single linear layer to rank the detected action classes using the loss functions described in the following section.

Model	Loss	Top-1	Top-5	mAP
I3D	BCE	0.529	0.778	0.554
	WARP	0.519	0.792	0.565
	LSEP	0.527	0.788	0.561
	wLSEP	**0.585**	**0.814**	**0.617**
Audio	BCE	0.068	0.193	0.069
	WARP	0.063	0.186	0.066
	LSEP	0.053	0.161	0.060
	wLSEP	**0.069**	**0.209**	**0.069**
Fusion	wLSEP	**0.593**	**0.828**	**0.618**

TABLE 1: **Validation Results:** Accuracy of the baseline models with different loss functions on the multi-label version of the validation set. We show Top1, Top5 and mean average precision (mAP) scores.

6.2 Performance Metrics

Accuracy

We report both the top-1 and top-5 classification accuracy for each of our models in order to be consistent with the results reported from the original Moments in Time paper [26]. Top-1 accuracy indicates the percentage of testing videos where the top predicted class is a positive label for the video. Similarly, top-5 accuracy indicates the percentage of the testing videos where any of the top predicted 5 classes for a video is a positive label.

Mean Average Precision (mAP)

We use mAP as our main evaluation metric as it captures the errors in the ranking of relevant actions for a video. For each positive label, mAP computes the proportion of relevant labels that are ranked before it and then averages over all of the labels.

7 Results

7.1 Loss Function Comparison

Table 1 displays results of the models trained using different loss functions on the proposed Multi-Moments in Time dataset. We can see that the proposed wLSEP loss function significantly outperforms the other approaches at optimization. The combination of stable pair-wise ranking with class balancing is very effective in training our multi-label network. The second best loss was WARP which outperformed LSEP and BCE. For our audio network the performance separation was much smaller but still results in our proposed wLSEP loss function achieving the best results. For audio we only train and evaluate our models on videos containing audio streams. Fusing the audio and I3D networks via an SVM results in slightly higher top-1 and top-5 accuracies than using I3D alone but does not significantly improve the mAP performance. This small improvement is likely due to the small number of videos in the validation set that contain audio streams as well as the dominance of visual-based labels in the dataset.

7.2 Transfer experiments

To evaluate the strength of the features learned from M-MiT we conducted a set of transfer experiments comparing ResNet50 I3D models pretrained on Kinetics [19], Moments in Time (MiT) and Multi-Moments in Time (M-MiT). We use a Resnet50 I3D model trained with the wLSEP loss function as

bicycling floating grooming writing jogging

Fig. 5: **Localized action regions:** Bounding boxes annotated around 549 different action categories in 67,468 image frames each selected from unique videos in the Moments in Time, Kinetics, and Something-Something datasets.

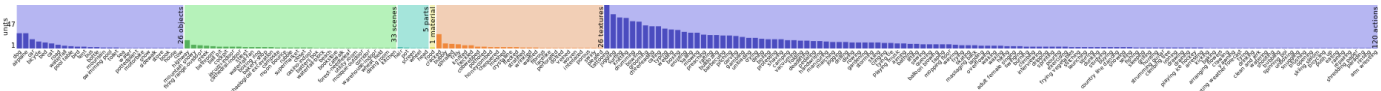


Fig. 6: **Graph of learned action concepts ordered by the number of features associated with each concept.**

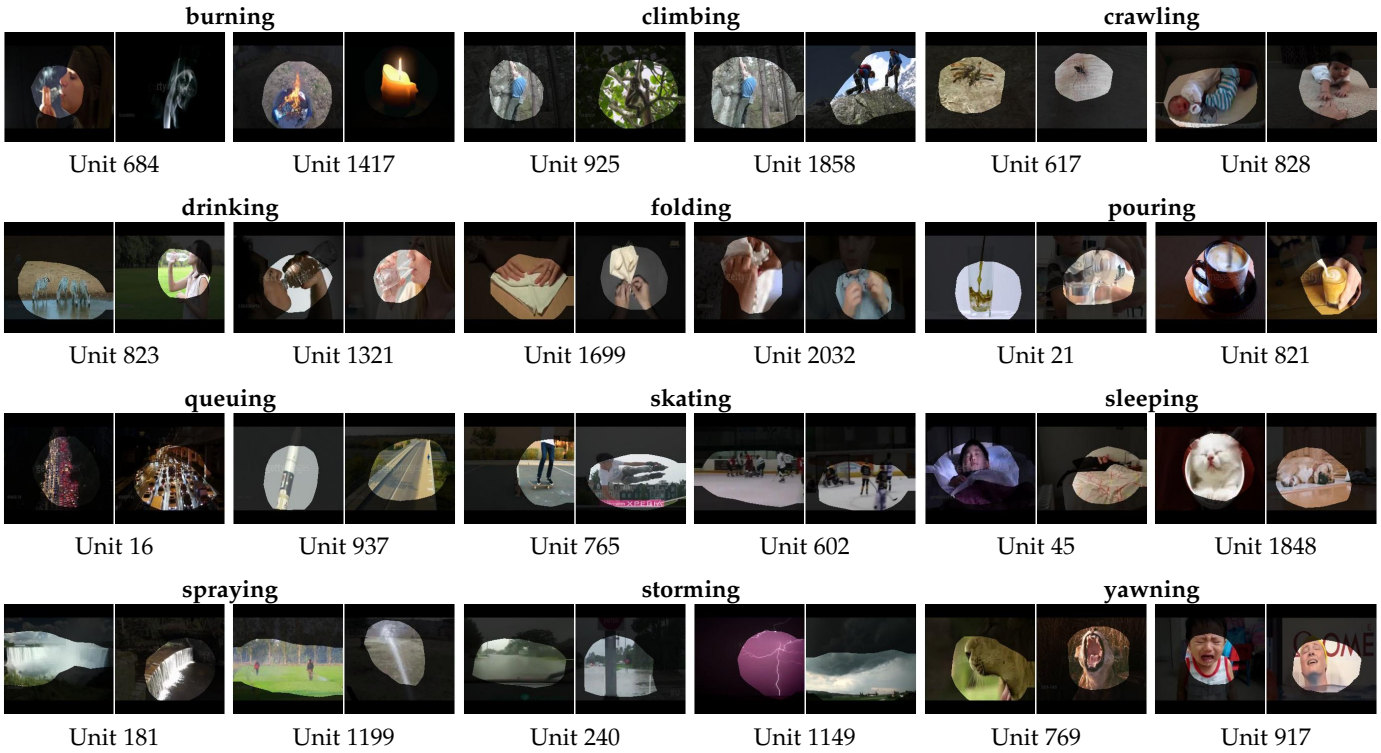


Fig. 7: **Visualization of learned action concepts:** Different features learn different representations of the same action. For example, units 684 and 1417 can both be interpreted as learning the concept of *burning* (top left). However, unit 684 learns to correlate *smoke* with the action while unit 1417 correlates it with a *flame*.

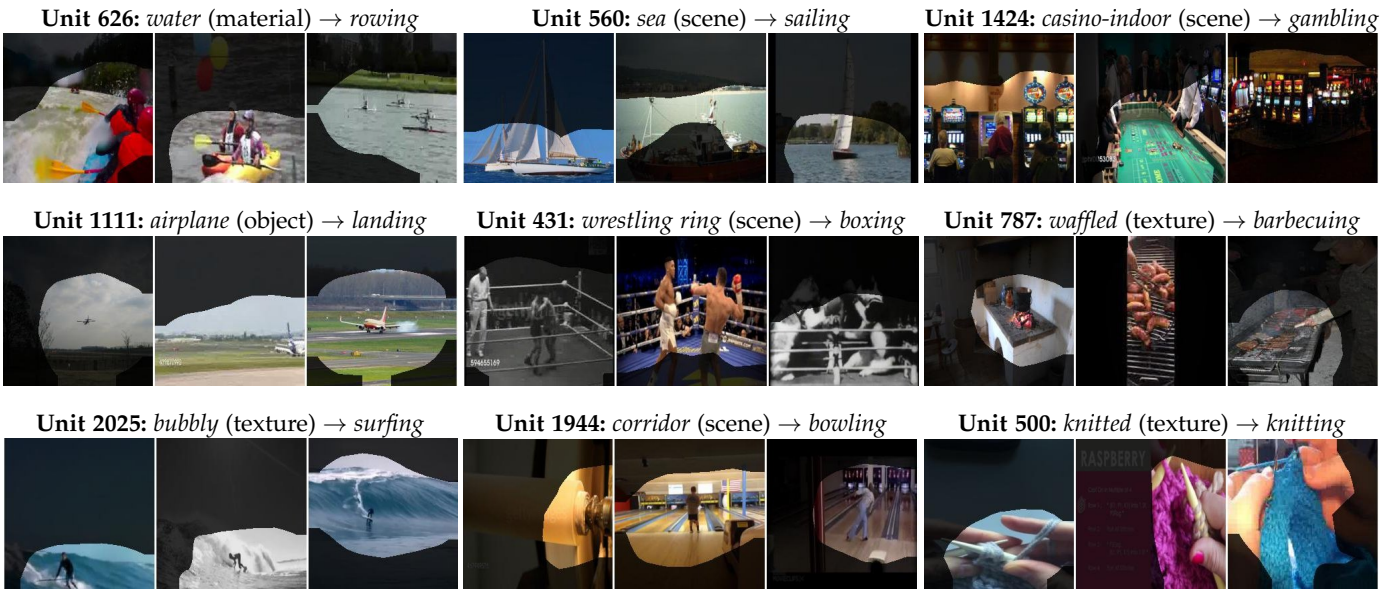


Fig. 8: **Improved feature interpretation:** Examples of the 953 units from the final residual block of a Resnet50 that changed their main interpretation when actions were added to the Broden dataset.

Pretrained	Fine-Tuned UCF-101			HMDB			AVA			MultiTHUMOS			Charades		
	Top-1	Top-5	mAP	Top-1	Top-5	mAP	Top-1	Top-5	mAP	Top-1	Top-5	mAP	Top-1	Top-5	mAP
Kinetics	0.905	0.987	0.942	0.726	0.911	0.809	0.781	0.973	0.772	0.837	0.964	0.821	0.388	0.731	0.303
MiT	**0.908**	0.986	**0.943**	**0.756**	**0.937**	**0.836**	0.802	0.976	0.791	0.849	0.968	0.838	0.383	0.711	0.305
M-MiT	0.892	0.980	0.932	0.740	0.921	0.819	**0.807**	**0.977**	**0.795**	**0.873**	**0.972**	**0.869**	**0.414**	**0.740**	**0.306**

TABLE 2: Dataset transfer performance using ResNet50 I3D models pretrained on Kinetics, Moments in Time (MiT) and the proposed Multi-Moments in Time dataset (M-MiT).

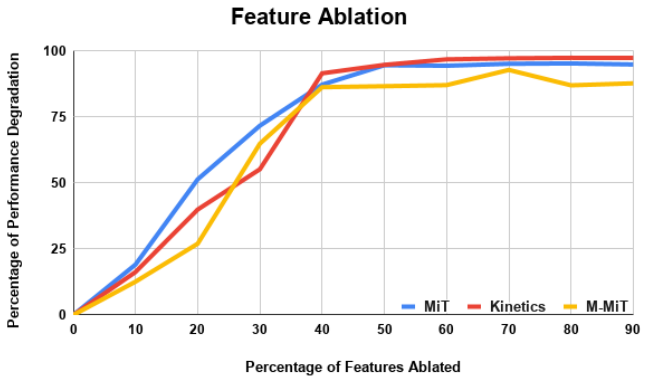


Fig. 9: **Feature Ablation** Comparing the effect of randomly deleting features in ResNet50 I3D models trained on Kinetics, Moments in Time (MiT) and the proposed Multi-Moments in Time dataset (M-MiT). The results show the mAP performance degradation of each model on their respective validation sets as an increasing percentage of features are ablated.

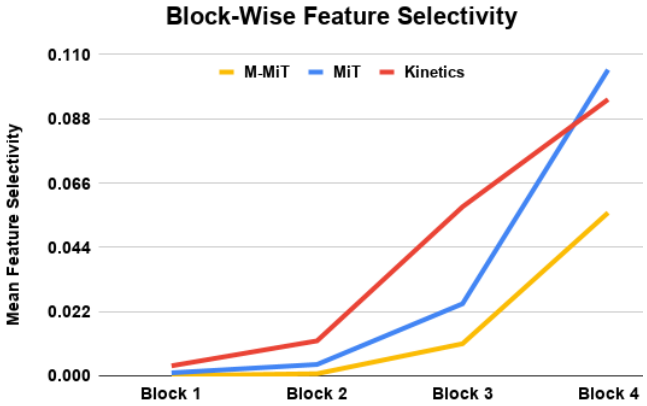


Fig. 10: **Feature Selectivity** Comparing the mean selectivity of the features after each residual block of ResNet50 I3D models trained on Kinetics, Moments in Time (MiT) and the proposed Multi-Moments in Time dataset (M-MiT).

this model gave us the best single stream performance on our dataset (Table 1). We compare our results when transferring to two single label datasets UCF-101 [31] and HMDB [21] as well as three multi-label datasets AVA [13], MultiThumos [39] and Charades [29].

Table 2 shows the results of the transfer task where the top-1, top-5 and mAP scores are calculated by evaluating on the validation set of the dataset used to fine-tune the model. We can see from the results that pretraining on M-MiT consistently results in better performance on the multi-label datasets. This makes sense as MiT and Kinetics are single-label datasets and were not trained to handle multiple co-occurring actions. For the single label datasets, MiT achieves very close performance to Kinetics on UCF-101 with M-MiT following in third. On HMDB, the M-MiT model performs a little worse than MiT and a little better than Kinetics. These results are fairly consistent with prior comparisons between Kinetics and MiT [26] and show that M-MiT pretrained models excel when transferring to multi-label settings.

8 Model Analysis

8.1 Robustness and Reliance on Single Directions

Prior work has proposed that models with more distributed representations, which are less reliant on single directions, benefit from improved generalization [27], [15]. In this section we explore two approaches for evaluating learned representations of models trained on Kinetics, Moments in Time (MiT) and the Multi-Moments in Time dataset (M-MiT).

8.1.1 Feature Ablation

In Figure 9 we examine the directional reliance of ResNet50 I3D models trained on Kinetics, Moments in Time (MiT) and the Multi-Moments in Time dataset (M-MiT). For each model, we randomly ablated (clamped activations to 0) an increasing percentage of the features in each model and report their degradation in performance as a percentage of their non-ablated validation mAP on each model's respective validation set. This allows us to compare models trained for different datasets as a function of their relative performance drop. The model trained on M-MiT is in general less susceptible to this ablation than the other models while the MiT model suffers from a strong performance drop early on with the Kinetics model only performing worse after at least 40% of the features were ablated.

8.1.2 Dependency on Single Directions

To further analyze the effect of training models using multiple labels we evaluate the mean class selectivity of the features after each residual block of ResNet50 I3D models trained on Kinetics, Moments in Time (MiT) and the Multi-Moments in Time dataset (M-MiT). This is similar to recent work in analyzing class selectivity in neural networks for image recognition [27] which suggested that class selectivity may be detrimental to network performance. We use the general index of selectivity [5] to estimate the dependence of our models on single features,

$$selectivity_i = 1 - \frac{\text{mean} c_i}{\max c_i}, \tag{5}$$

where $\text{mean} c_{\mathbf{f}_i}$ and $\max c_{\mathbf{f}_i}$ are the average and maximum class probabilities when the activation of feature i is set to 1 and the activation of all other features, and bias values, in the layer are set to 0. Figure 10 shows the block-wise selectivity, the average selectivity of all the output features of each residual block, of ResNet50 I3D models trained on Kinetics, Moments in Time (MiT) and the proposed Multi-Moments in Time dataset (M-MiT). The block-wise selectivity

of the M-MiT model is consistently lower than the MiT and Kinetics models for each residual block showing a lower dependency on specific features for class predictions.

8.2 Multi-Label Class Activation Mapping

In Figure 3 we see an example of applying our multi-label CAM filter to an image with two actions. First we show the use of the original CAM method for identifying regions of importance for the actions *coaching* and *punching* in the same image. We then show an example of using the proposed multiCAM method to separate the key areas of importance that are specific to each action in the image. We include this example to highlight how the regions can seem a little ambiguous between the two different actions using the original CAM approach. This similarity is likely due to the actions incorporating contextual cues beyond the dynamic action region to make their predictions. Applying multiCAM to the image allows us to more clearly separate the regions based on their relative importance to each class. The proposed method captures the subtle differences, and similarities, in each actions CAM and allows us to visualize the distinct regions of interest for each action. In Figure 4 we include a diverse set of examples showing results of this method for different action combinations.

8.3 Learned Action Interpretation

Using the approach described in Section 5.2 we are able to identify a total of 211 concepts consisting of 1 material, 5 part, 26 texture, 26 object, 33 scene and 120 action concepts learned in 2023 different features out of 2048 (Figure 6) units in the final convolutional layer (block4) of a Resnet50 network trained on the Multi-Moments in Time dataset. Figure 7 highlights some of the learned concepts. Interestingly the network seems to be recognizing the pattern of a person standing behind a podium as *preaching* which is definitely a common correlation in our dataset. Similarly, the network associates *crawling* with babies as many of our videos of crawling typically depict babies *crawling*. These are the types of data and class biases that are useful to identify via network interpretation that may have gone unnoticed without the ability to identify action concepts.

Table 3 highlights the fact that including actions in the Broden dataset helps to interpret a much larger portion of the features in block 4 of a ResNet50 trained for action recognition. With the original Broden set (no actions) NetDissect identified 185 concepts in 1351/2048 features. Adding actions to this set allowed us to identify 2023/2048 features that can be interpreted for 211 different concepts including 120 actions. This jump in the number of interpretable features makes sense for the final block of a model trained for action recognition and suggests that excluding action concepts misses a large portion of useful information when interpreting these models.

The results from the combined set highlight that some of the features previously interpreted by the original Broden set as object or texture concepts are closely aligned with actions. For example, unit 13 was classified using the Broden set as learning the concept *potted plant* with an IoU of 0.06, but if we include action concepts the unit is found to be more correlated with the action *gardening* with an IoU of 0.15. Similarly, unit 290 was identified by Broden as learning the texture concept *cracked* with an IoU of 0.1 and including actions we found a greater association with the action *typing* with an IoU of 0.34. Features for identifying the ridges between the keys in the keyboards commonly found in actions of *typing* were correctly activating for the texture *cracked*. Figure 8 shows a few examples of this behavior and how adding actions to the Broden dataset improves our ability to interpret our action model.

Category	Concepts	Interpretable Features
Broden	185	1351
Action Regions	140	1995
Broden+Action Regions	211	2023

TABLE 3: Comparison of the number of concepts and interpretable features identified by NetDissect given the Broden dataset, the Action Region dataset and the combined dataset on block 4 of a ResNet50 trained for action recognition.

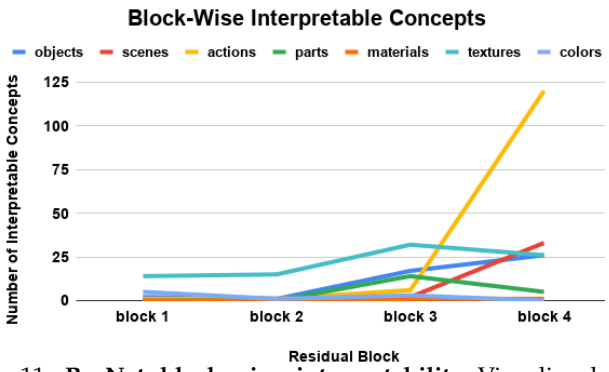


Fig. 11: **ResNet block-wise interpretability** Visualize how different semantic concepts - objects, scenes and actions emerge across residual blocks of the ResNet50 network.

8.3.1 *Block-wise Interpretability*

To understand how individual units evolve over residual blocks we evaluate the interpretability of features from different blocks of a resnet50 network trained for action recognition on the Moments in Time dataset [26] in terms of objects, scenes, actions and textures. In Figure 11 we observe that action features mainly emerge in the last convolutional block (block 4) of the model. It is interesting to note that object and scene features are learned even if the model is not explicitly trained to recognize objects or scenes suggesting that object and scene recognition aids action classification.

8.3.2 *Interpretable feature relationships*

Examining these interpretable features and how they contribute to a networks output allows us to build a better understanding of how our models will function when presented with different data. For example, we can consider a feature in the final residual block of the network that has been interpreted as a *highway* scene feature. If we activate only this unit by setting its values to 1 and all other feature (including bias) values to 0 and examine the output we can identify which action classes are using the fact that a video may take place on a *highway*. In this case the action classes that achieve the highest output are *hitchhiking*, *towing*, *swerving*, *riding*, and *driving*. These interpretable feature-class

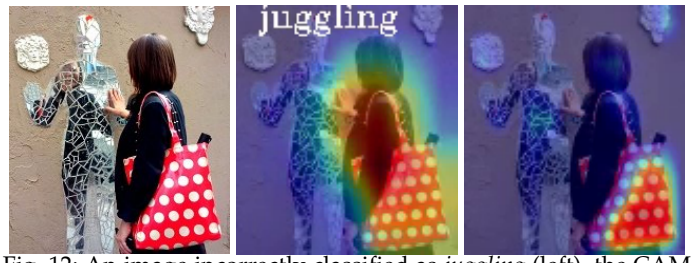


Fig. 12: An image incorrectly classified as *juggling* (left), the CAM of the model showing a strong activation on the purse (middle) and a heatmap overlay of the average feature activations from the 3rd residual block of the model showing an even stronger focus on the purse (right).

Fig. 13: Examples of the top block 4 features strongly associated with the output class *juggling*.

Fig. 14: Examples of the top block 3 features associated with *juggling* features in block 4. These examples show a bias learned by the model towards detecting the action *juggling* when an image/video contains white polka dotted patterns

relationships make sense as all of these actions are likely to occur near a *highway*.

Similarly, we can examine the relationships between different interpretable features at different layers in the network to help us understand the concept hierarchy the model is using to make a decision. Understanding this process can be very useful in diagnosing why a network makes a mistakes in its output. For example, if we pass the image in Figure 12 through our action recognition model we get a top prediction of *juggling*. Of course the image does not depict the action *juggling*. To diagnose why the network was incorrect we performed the inverse of the operation described in the previous paragraph and iteratively set the value of each feature in block4 of the model to 1, and the others to 0, compared the resulting outputs for the action *juggling* and identified the interpretable features that contributed the most to the mistake (i.e. resulted in the highest output for *juggling*). Unfortunately, in this case the top 5 features that contribute to the class *juggling* are also interpreted as *juggling* features and none of these features share a significant correlation with any other concept in the Broden dataset.

To address this we ran the same process again on block3 by considering the features in block4 as the output and compared the activations of the *juggling* features. This step was much more informative on how the model arrived at its mistake. We found that the interpretable concepts in block3 that contributed the most to the *juggling* features in block4 were the textures *dotted* and *polka dotted* and the color *white*. We can see that the woman in the image is holding a red and white polka-dotted purse and running Class Activation Mapping (CAM) [46] on the image confirms that the area around the purse is an area of interest for the model (Figure 12). We were thus able to identify the hierarchical concept path within the network that led to the incorrect classification.

9 Conclusion

Progress in the field of video understanding will come from many fronts, including training our models with richer and more complete information, so they can start achieving recognition performances in par with humans. Augmenting a large-scale dataset by doubling the number of activity labels, we present baseline results on the Multi-Moments dataset as well as improved methods for visualizing and interpreting models trained for multi-label action detection.

Acknowledgements: This work was supported by the MIT-IBM Watson AI Lab, Google faculty award and SystemsThatLearn@CSAIL award (to A.O), as well as the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/ Interior Business Center (DOI/IBC) contract number D17PC00341. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions 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 IARPA, DOI/IBC, or the U.S. Government.

Special thanks to David Bau for assisting in extending the Broden Dataset for Network Dissection and Allen Lee for aiding in the design of our demo.

REFERENCES

[1] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016.

[2] Y. Aytar, C. Vondrick, and A. Torralba. Soundnet: Learning sound representations from unlabeled video. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*. 2016.

[3] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Computer Vision and Pattern Recognition*, 2017.

[4] S. Bell, P. Upchurch, N. Snavely, and K. Bala. OpenSurfaces: A richly annotated catalog of surface appearance. *ACM Trans. on Graphics (SIGGRAPH)*, 32(4), 2013.

[5] E. L. Bienenstock, L. N. Cooper, and P. W. Munro. Neurocomputing: Foundations of research. chapter Theory for the Development of Neuron Selectivity: Orientation Specificity and Binocular Interaction in Visual Cortex, pages 437–455. MIT Press, Cambridge, MA, USA, 1988.

[6] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015.

[7] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. *Proc. ICCV*, 2017.

[8] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, and A. Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[10] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pages 681–687, Cambridge, MA, USA, 2001. MIT Press.

[11] Y. Gong, Y. Jia, T. Leung, A. Toshev, and S. Ioffe. Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894*, 2013.

[12] R. Goyal, S. Kahou, V. Michalski, J. Materzyńska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. *arXiv preprint arXiv:1706.04261*, 2017.

[13] C. Gu, C. Sun, S. Vijayanarasimhan, C. Pantofaru, D. A. Ross, G. Toderici, Y. Li, S. Ricco, R. Sukthankar, C. Schmid, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. *arXiv preprint arXiv:1705.08421*, 2017.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] G. E. Hinton, J. L. McClelland, and D. E. Rumelhart. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Distributed Representations, pages 77–109. MIT Press, Cambridge, MA, USA, 1986.

[16] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1):185 – 203, 1981.

[17] P. Hu, G. Wang, X. Kong, J. Kuen, and Y.-P. Tan. Motion-guided cascaded refinement network for video object segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[18] Y. Jiang, J. Liu, A. R. Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. Thumos challenge: Action recognition with a large number of classes, 2014.

[19] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[21] H. Kuehne, H. Jhuang, R. Stiefelhagen, and T. Serre. Hmdb51: A large video database for human motion recognition. In W. E. Nagel, D. H. Kröner, and M. M. Resch, editors, *High Performance Computing in Science and Engineering '12*, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[22] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[23] Y. Li, Y. Song, and J. Luo. Improving pairwise ranking for multi-label image classification. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1837–1845, 2017.

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[25] G. A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, Nov. 1995.

[26] M. Monfort, A. Andonian, B. Zhou, K. Ramakrishnan, S. A. Bargal, T. Yan, L. Brown, Q. Fan, D. Gutfruend, C. Vondrick, et al. Moments in time dataset: one million videos for event understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–8, 2019.

[27] A. S. Morcos, D. G. Barrett, N. C. Rabinowitz, and M. Botvinick. On the importance of single directions for generalization. In *ICLR*, 2018.

[28] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille. The role of context for object detection and semantic segmentation in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[29] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. *CoRR*, abs/1604.01753, 2016.

[30] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[31] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[33] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *CVPR*, 2015.

[34] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 2285–2294. IEEE, 2016.

[35] Y. Wang and M. Hoai. Pulling actions out of context: Explicit separation for effective combination. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[36] J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three*, IJCAI'11, pages 2764–2770. AAAI Press, 2011.

[37] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition*, 2017.

[38] H. Yang, J. Tianyi Zhou, Y. Zhang, B.-B. Gao, J. Wu, and J. Cai. Exploit bounding box annotations for multi-label object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 280–288, 2016.

[39] S. Yeung, O. Russakovsky, N. Jin, M. Andriluka, G. Mori, and L. Fei-Fei. Every moment counts: Dense detailed labeling of actions in complex videos. *arXiv preprint arXiv:1507.05738*, 2015.

[40] J. Yuen, B. Russell, C. Liu, and A. Torralba. Labelme video: Building a video database with human annotations. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1451–1458. IEEE, 2009.

[41] J. Zhang, Q. Wu, C. Shen, J. Zhang, and J. Lu. Multi-label image classification with regional latent semantic dependencies. *IEEE Transactions on Multimedia*, 2018.

[42] M.-L. Zhang and Z.-H. Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Trans. on Knowl. and Data Eng.*, 18(10):1338–1351, Oct. 2006.

[43] B. Zhou, A. Andonian, A. Oliva, and A. Torralba. Temporal relational reasoning in videos. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[44] B. Zhou, A. Andonian, and A. Torralba. Temporal relational reasoning in videos. *arXiv preprint arXiv:1711.08496*, 2017.

[45] B. Zhou, A. Khosla, L. A., A. Oliva, and A. Torralba. Learning Deep Features for Discriminative Localization. *CVPR*, 2016.

[46] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016.

[47] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Semantic understanding of scenes through the ade20k dataset. *arXiv preprint arXiv:1608.05442*, 2016.