

Learning Action Concept Trees and Semantic Alignment Networks from Image-Description Data

Jiyang Gao^(✉) and Ram Nevatia

University of Southern California, Los Angeles, USA
jiyangga@usc.edu

Abstract. Action classification in still images has been a popular research topic in computer vision. Labelling large scale datasets for action classification requires tremendous manual work, which is hard to scale up. Besides, the action categories in such datasets are pre-defined and vocabularies are fixed. However humans may describe the same action with different phrases, which leads to the difficulty of vocabulary expansion for traditional fully-supervised methods. We observe that large amounts of images with sentence descriptions are readily available on the Internet. The sentence descriptions can be regarded as weak labels for the images, which contain rich information and could be used to learn flexible expressions of action categories. We propose a method to learn an Action Concept Tree (ACT) and an Action Semantic Alignment (ASA) model for classification from image-description data via a two-stage learning process. A new dataset for the task of *learning actions from descriptions* is built. Experimental results show that our method outperforms several baseline methods significantly.

1 Introduction

Action classification in still images has been a popular research topic in computer vision. Traditional fully-supervised learning methods for action classification rely on large amount of fully-labelled data (*i.e.* each image is labelled with one or more action categories) to learn action classifiers. However, labelling image data with action categories requires tremendous manual work, which is time-consuming and hard to scale-up. Another drawback of traditional supervised learning framework is that the action categories are pre-defined and limited, while humans may describe the same action with different phrases, for example, take out the chopping board and fetch out the wooden board. This drawback leads to the difficulty of vocabulary expansion, as CNN [1, 2] models or SVM classifiers just assign a label to the test image. Hence, CNN or SVM models would fail to classify the categories that are not in the training set.

We observe that large amounts of images with sentence descriptions are readily available on the Internet, such as videos with captions and social media, such as Flickr and Instagram. Such sentence descriptions can be regarded as weak labels of the images. Sentence descriptions are generated by humans and

Image-description pairs



Fig. 1. Descriptions, as weak labels to images, contain rich information about actions. Images and corresponding descriptions are from Visual Genome [3]

contain rich information about actions, which could be used to learn an expanding vocabulary of actions. Some example are shown in Fig. 1. Another observation we make is that action concepts are naturally represented as a hierarchy; for example, “play guitar” and “play violin” are subcategories of “play instrument”. If such hierarchical structure of action categories is available, classification methods can choose to use detailed knowledge if necessary or generalized knowledge when details are unavailable or irrelevant.

In this paper, we propose a method to tackle the problem of *learning actions from descriptions*: Given a set of image-description data (assuming descriptions containing human action information), learning to recognize human actions. Our method supports hierarchical clustering of action concepts and vocabulary expansion for action classification. Specifically, our method learns an Action Concept Tree (ACT) and an Action Semantic Alignment (ASA) model for classification via a two-stage learning process. ASA model contains a CNN to extract image-level features, an LSTM to extract text embeddings and a multi-layer neural network to align these two modalities. In the first stage, (a) we design a Hierarchical Action Concept Discovery (H-ACD) method to automatically discover action concepts from image-description data and cluster them into a hierarchical structure (*i.e.* ACT); (b) ASA is initialized by the image-description mapping task in stage-1. In the second stage, the target action categories are matched to the nodes in ACT and the associated image data are used to fine-tune ASA for this action classification task to improve the performance. Note that no image data from test domain are used for training.

To facilitate research on this task, we constructed a dataset based on Visual Genome [3], called Visual Genome Action (VGA). Although Visual Genome contains well-annotated region descriptions, we do not use the region information and treat the descriptions as image-level. There are 52931 image-description pairs in the training set and 4689 images of 45 categories in test set. More details of this dataset are given in Sect. 4.1 later.

In summary, our main contributions are:

- (1) A Hierarchical Action Concept Discovery (H-ACD) algorithm to automatically discover an Action Concept Tree (ACT) from image-description data and gather samples for each action node in ACT.
- (2) An end-to-end CNN-LSTM Action Semantic Alignment (ASA) network which aligns semantic and visual representation to classify actions with expanding vocabulary.
- (3) A dataset for the problem of *learning actions from descriptions*, which is built on Visual Genome, containing 52931 image-description pairs for training and 45 action categories for testing.

The paper is organized as follows. Section 2 discusses the related works. In Sect. 3, we will introduce our two-stage framework to learn actions from image-description data. We evaluate our model in Sect. 4 and give our conclusions in Sect. 5.

2 Related Work

Action Classification in Still Images: The use of convolutional neural network (CNN) has brought huge improvement in action classification [4]. [5] fine-tunes the CNN pre-trained on ImageNet and shows improvement over traditional methods. [6] designs a multi-task (person-detection, pose-estimation and action classification) model based on R-CNN. [7] develops an end-to-end deep convolutional neural network that utilizes contextual information of actions. HICO [8] introduces a new benchmark for recognizing human-object interactions, which contains a diverse set of interactions with common object categories, such as “hold banana” and “eat pizza”. Ramanathan *et al.* [9] proposes a neural network framework to jointly extract the relationship between actions and uses them for training better action retrieval models. These methods all rely on fully-labelled data.

Weakly Supervised Action Concept Learning: Weakly supervised action concept learning relies on weakly-labelled data, such as video-caption stream data [10, 11] and focuses on automatically discovering and learning action concepts. [12] designs a method to automatically discover the main steps for specific tasks, such as “make coffee” and “change tire”, from narrated instructional videos. Their method solves two clustering problems, one in text and one in video, applied one after each other and linked by joint constraints to obtain a single coherent sequence of steps in both modalities. Ramanathan *et al.* [13] propose a

method to learn action and role recognition models based on natural language descriptions of the training videos. Yu *et al.* [14] discover Verb-Object (VO) pairs from the captions of the instructional videos and use the associated video clips as training samples. The learned classifiers are evaluated in event classification, compared with well defined action categories in HMDB51 [15] and UCF50 [16]. [17] proposes a general concept discovery method from image-sentence corpora and apply the concepts on image-sentence retrieval tasks.

ACD [18] solves a similar problem to ours. It automatically discovers action concepts from image-sentence corpora [19, 20], clusters them and trains classifier for each action concept cluster. However, there are two main drawbacks in this method: (1) no hierarchical clustering: once the action concepts are clustered, the detailed information are lost; (2) no vocabulary expansion: if the target test action categories are missed in the training set, ACD would fail to perform classification.

Language & Vision: Image captioning methods take an input image and generate a text description of the image content. Recently, methods based on convolutional neural networks and recurrent neural networks [21, 22] have shown to be an effective way on this task. VSA [23] is one of the recent successful models. It uses bidirectional recurrent neural networks over sentences, convolutional neural networks over image regions and a structured objective that aligns the two modalities through a multimodal embedding. Besides image captioning, other relevant work includes natural language object retrieval [24] or segmentation [25], which takes an input image and a query description and outputs a corresponding object bounding box or a segmentation mask.

3 Actions from Descriptions

In this section, we introduce the learning framework, which is a two-stage method. In the first stage, our target is to learn a general knowledge base of actions, which contains two parts: a hierarchical structure for action concepts and a general visual-semantic alignment model. In the second stage, the framework learns to classify specific action categories (*i.e.* target categories for test). The classifiers are fine-tuned from the visual-semantic alignment model learned in stage 1. The overall system is shown in Fig. 1.

3.1 Stage 1: Learning General Action Knowledge

As for general action knowledge, we refers to two concepts. The first one is a hierarchical structure of actions, which we call Action Concept Tree (ACT): each node in ACT contains an action concept, such as play frisbee and play basketball, and the related images; the action concepts are extracted from descriptions and the images come from the original image-description dataset. The second one is a general visual-semantic alignment model: the input of the model is an image and a description, and the output is a confidence score of the similarity of the image and the description. The framework of Stage 1 is shown in Fig. 2.

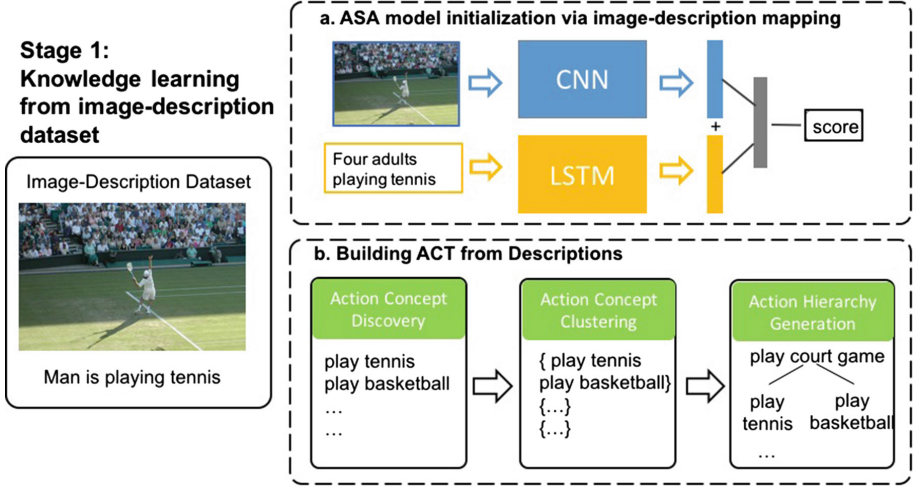


Fig. 2. Stage-1: model initialization via image-description matching and hierarchy action discovery

Hierarchical Action Concept Discovery (H-ACD): ACD [18] proposed an action concept discovery method working with image-sentence corpora. However, the discovered action clusters are not organized in a hierarchical structure, which may lose the detailed information after clustering. Hence, based on ACD, we propose a Hierarchical Action Concept Discovery (H-ACD) method, which automatically discovers action concepts from image-description data and organizes them in a hierarchical structure using WordNet [26]. The process of action concept discovery and clustering are similar to ACD. First we extract Verb-Object (VO) pairs from sentence descriptions and the visualness of these VO pairs are verified by two fold cross-validation. After visualness verification, we generate a multi-modal representation for each action concept and calculate similarity score for each pair of action concepts.

After computing the similarity, we use the H-ACD algorithm to generate a hierarchical structure for action concepts. Note that nearest neighbor (NN) clustering algorithm is proposed in ACD [18]; we use it as a part of our H-ACD algorithm. We first apply NN-clustering algorithm (we fix the parameter C of NN-clustering as 4.) [18] on all the action concepts to get a list of action clusters. Then, inside each cluster, we continuously apply NN-clustering algorithm to get more smaller clusters; we do this recursively until no new cluster is generated. Each cluster is regarded as a node in the hierarchical structure and the node names are generated following a similar naming strategy of HAN [27] described in the following. For the object part, we find the lowest common hypernym in WordNet. For the verb part, we follow a simple strategy: if the verbs are the same, then the father node keeps the same verb; if the verbs are different, the father node is named as “interact with”. For example, for a node containing

{hold dish, hold pan}, the least shared parent of dish and pan is container and for the verb part, “hold” itself is the least shared parent. So the name of this action node is “hold container”. The H-ACD algorithm is shown in Algorithm 1.

Data: Concept similarity matrix M of size $l \times l$ and concept list L of size l
Result: Action Concept Tree (ACT)
Queue $q \leftarrow \text{NN-Clustering}(M, L)$;
TreeNode root;
root.addChild(q.all());
while q not empty **do**
 $L_{cluster} \leftarrow q.pop()$;
 node=ACT.getNode($L_{cluster}$);
 $M_{cluster} \leftarrow \text{getSimMat}(L_{cluster})$;
 tinyclusters=NN-Clustering($M_{cluster}, L_{cluster}$);
 q.push(tinyclusters);
 node.addChild(tinyclusters);
end
Generate node names following the naming strategy.

Algorithm 1. Hierarchical Action Concept Discovery (H-ACD) algorithm

ASA Model Initialization via Image-Description Mapping: Our final target is to classify action categories. Rather than training classifiers for each category, we want to build a connection between the semantic meaning and visual meaning of actions. Therefore, we formulate the action classification as a visual-semantic alignment problem between the image and action categories. The Action Semantic Alignment (ASA) model contains three parts: a CNN network to extract feature vector of the input image, an LSTM network to extract text embedding and an alignment network to compute the alignment score of the visual and semantic representations. Image-description mapping serves as a parameter initialization method for ASA model, which helps the model to learn a connection between semantic and visual spaces.

The input of ASA is an image I_i and the corresponding sentence description D_i . The image is processed by VGG-16, which outputs a d_{img} dimensional feature v_i . For a text sequence $S = (w_1, \dots, w_T)$ with T words, each word is transformed to a d_{w2v} dimensional vector by the word embedding matrix and then processed by an LSTM module sequentially. The word embedding matrix is trained by skip-gram model on English Wiki data. At the final time step $t = T$, LSTM outputs the final hidden state and we use it as the sentence-level embedding s_i , which is a d_{text} dimensional vector. v_i and s_i are concatenated to vs_{ii} with a length of $d_{vs} = d_{text} + d_{img}$, which is visual-semantic representation of the image and description. Then we train a two-layer alignment network, with a d_{alg} dimensional hidden layer. The alignment network take d_{vs} dimensional input and output a confidence score cs_{ii} , which indicates whether the image and

the description is aligned. The alignment network is implemented in a fully convolutional way as two $1 * 1$ convolutional layers (with ReLU function between them).

During the training time, we optimize the model inside each mini-batch. The loss function is as follows.

$$loss_1 = \sum_{i=0}^N [\alpha_c \log(1 + \exp(-cs_{i,i}))] + \sum_{j=0, j \neq i}^N \alpha_w \log(1 + \exp(cs_{i,j})) \quad (1)$$

where N is the batch size, α_c and α_w are the loss weights for correct and wrong image-description. The loss function encourages the network to output high score of correct image-description pairs and low score of incorrect image-description pairs. In practice, we find that training converges faster using higher loss weights for correct pairs and we use $\alpha_c = 1$ and $\alpha_w = 0.01$.

3.2 Stage 2: Action Classification on Target Categories

Given a set of action categories for classification (without training samples), we adjust the ASA model to the specific action classification task. The first step is to match the given action categories to some existing action nodes in ACT. Then we use the matched action nodes and the associated images to fine-tune our ASA model. The framework of Stage 2 is shown in Fig. 3.

Target Action Categories Matching: We first match the actions via keyword searching. Suppose the target action category is c_i and the action node in ACT is represented by n_j . We extract the verb and object from the target action category c_i and search for them in the discovered action hierarchy to see if there is an exact match. For example, a target action category is “play instrument” and there is a node in action hierarchy named “play instrument”, then we match them and use the similarity score (calculated by ASA, see below) between them as a baseline score θ . If there is no exact match via keyword searching, we assign

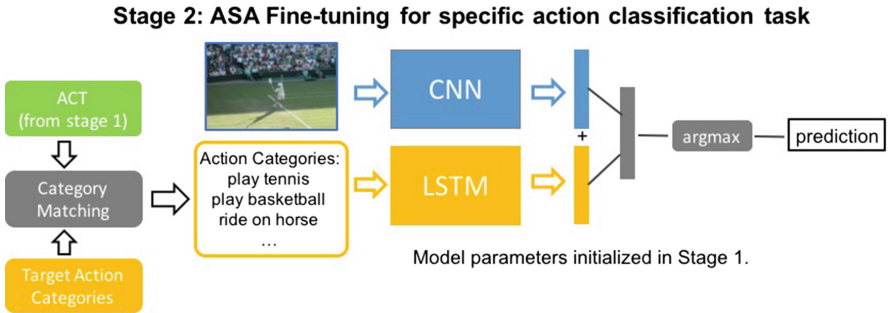


Fig. 3. Stage-2: adjust the model to a specific action classification task.

θ with a constant value. In the second step, we use the ASA model to compute a similarity score between the target action category c_i and all action nodes n_j in ACT. The associated images of n_j are $P_j = \{I_{jk}\}$, which has size m . The similarity score between c_i and n_j is

$$S(c_i, n_j) = \frac{1}{m} \sum_{k=0}^m \text{ASA}(I_{jk}, c_i) \quad (2)$$

For a specific action category c_i , we select the action node n_j that has the highest similarity score and if the score is larger than or equal to θ , we match $\langle c_i, n_j \rangle$. Note that some categories may still not be matched after the second step. After matching, we obtain a list of training samples $\{c_i, P_j\}$. The labels are c_i and the training images are the associated images of the corresponding matched node n_j . We don't assign any training data for the target categories with no matched node in ACT. The matching algorithm is detailed in Algorithm 2 below.

```

Data: ASA model, ACT and Target action categories  $C = \{c_i\}$ 
Result: Matched pairs  $\langle c_i, n_j \rangle$ 
for  $c_i$  in  $C$  do
  for  $n_j$  in  $ACT$  do
    if  $c_i.name = n_j.name$ : match  $\langle c_i, n_j \rangle$ ;
    break;
  end
  if  $c_i$  is matched:
    ExactMatch,  $\theta \leftarrow n_j, S(c_i, n_j)$ ;
    match  $\langle c_i, \text{ExactMatch} \rangle$ ;
  else:
    ExactMatch,  $\theta = \text{None}$ , InitializationValue;
    MaxScore=0;
    for  $n_j$  in  $ACT$  and  $n_j \neq \text{ExactMatch}$  do
      if  $S(c_i, n_j) > \text{MaxScore}$ :
        Node, MaxScore  $\leftarrow n_j, S(c_i, n_j)$ 
    end
    if MaxScore  $\geq \theta$ : match  $\langle c_i, \text{Node} \rangle$ ;
end

```

Algorithm 2. Target action categories matching algorithm

ASA Fine-Tuning for Specific Action Classification Task: We use the training samples obtained in last step to fine-tune the network. In stage 1, the loss function tends to match the correct image-description pair and it works as a parameter initialization method. In stage 2, our goal is to optimize the model to some specific classification task. We formulate the classification problem as a image-description matching problem. The name of the category is regarded as a text sequence, just like the sentence description. Suppose there are M categories,

leading to M corresponding category descriptions $\{CD_j, j = 0, 1, 2, \dots, M - 1\}$. Suppose the label of the input image I_i is t_i , then the loss function of stage 2 is as follows.

$$loss_2 = \sum_{i=0}^N [\alpha_c \log(1 + \exp(-cs_{i,t_i})) + \sum_{j=0, j \neq t_i}^M \alpha_w \log(1 + \exp(cs_{i,j}))] \quad (3)$$

where $cs_{i,j}$ is the matching score between I_i and CD_j , N is the batch size. We use $\alpha_c = 1$ and $\alpha_w = 0.01$. The loss function encourages the correct image-action pairs to output high positive score and other wrong pairs output low negative score.

Action Category Prediction: At test time, the prediction of an input image I_i is the argmax of the matching scores $cs_{i,j}$ between I_i and CD_j .

$$\text{prediction}(I_i) = \text{argmax}(cs_{i,j}), \quad j = 0, 1, 2, \dots, M - 1 \quad (4)$$

4 Evaluation

4.1 Experiments on VGA

Dataset: Visual Genome Action (VGA). There are many image-description datasets, which are suitable for *learning actions from descriptions*. However, none of them contain pre-defined action categories and category annotations for each image. Therefore, we construct a dataset from Visual Genome for this problem, called Visual Genome Action (VGA). We split Visual Genome into two parts: 75% for training and validation and 25% for testing. The training set and test set are carefully checked to ensure that there is no overlap of images between these two sets.

For the training split, since we only focus on human action learning, we filter out the descriptions which don't have verbs or human subjects; for example, "a dog is running on the grass" and "a man with a white shirt" are filtered out. 52931 image-description pairs remain after such filtering. The descriptions in Visual Genome are region based, but we treat them as image-level descriptions. For the test split, we extract Verb-Object (VO) pairs and filter out the ones with very few image samples. After that, we manually filter out the VOs with no visual meaning, such as "do things". Finally, there are 45 categories and 4689 images for testing. The 45 test action categories are listed in Table 1. Some categories overlap; for example, "hold racket" and "play tennis", "hit ball" and "play soccer". We manually checked each image of these categories and added additional labels if necessary. For example, if an image of the category "hold racket" also represents the action of "play tennis", then we also add this image to the category of "play tennis". In other cases, people may be just holding a tennis racket but not playing, then we don't add additional labels to such images.

Table 1. Action categories in VGA test set

boat	brush tooth	color hair	do trick	drink wine
eat fruit	eat pizza	enjoy outdoors	fly kite	hit ball
hold bag	hold banana	hold bat	hold camera	hold controller
hold dog	hold fork	hold kite	hold knife	hold pole
hold racket	hold sandwich	hold umbrella	jump	play baseball
play basketball	play frisbee	play soccer	play tennis	read book
ride elephant	ride horse	ride wave	run	sit
ride skateboard	ski	smile	stand	surf
swim	use phone	walk	watch game	wear necklace

Metric. We tested our model on action classification task on VGA. As for evaluation metric, we report the mean Average Precision (mAP), Recall@1 and Recall@5.

Network Implementation. We implemented ASA network in Tensorflow [28], including CNN network, LSTM network and the multi-layer alignment network. For the CNN part, We use VGG-16 architecture and the parameters are initialized by ImageNet [29] image classification dataset. We use a standard LSTM architecture with 1000-dimensional hidden state. The descriptions input to LSTM have maximum length of 6 for both stage-1 and stage-2. The hidden layer of the visual-semantic alignment network is 500-dimensional. We train a skip-gram [30] model for the word embedding matrix using the English Dump of Wikipedia. The dimension of the word vector is 500. The whole network is trained end-to-end in two stages. We use three Adam optimizers [31] to optimize CNN, LSTM and the visual-semantic alignment network. The learning rates are 0.0001, 0.001 and 0.001 respectively. The model is trained on a Tesla K40 GPU; the batch size is 96. It takes about 1 day to train the whole model for both stage-1 and stage-2.

System Variants. We experimented with variants of our system to test the effectiveness of our method. **ASA (Stage 1):** we only trained the ASA model for stage-1 using the image-description pairs. **ASA (Stage 2):** we only trained the ASA model for stage-2 using the matched action nodes in ACT. **ASA (Stage 1+2, w/o ACT):** ASA model is trained for stage-1 and stage-2, but we only use flat action concepts (*i.e.* only the leaf action nodes in ACT) to match the target action categories. **ASA (Stage 1+2, w/ ACT):** this is our full model; ASA model is trained for stage-1 and stage-2, and full ACT is used to match the target action categories.

Baseline Methods. We introduce the baseline methods we implemented.

ACD [18]+SVM+AdaBoost: In this baseline method, we use ACD [18] to discover a list of action concepts from the training set and train SVM classifiers [32] for each action concept. Then we match the test action categories with the discovered action concepts by keyword searching. Multiple action concepts may be matched to the same test categories and each of them can be regarded as weak classifier to the test category. To make use of all the related training data, we further use AdaBoost to build a stronger classifier.

ACD [18]+DeViSE [33]: In this baseline method, we first use ACD to discover a list of action concepts. Instead of training SVM classifiers for each of them, we apply DeViSE [33] methodology. The verb and the object of an action category are transformed to vectors using a word embedding matrix and are concatenated together. The word embedding matrix is trained by wiki dump data and the dimension of the word vector is 500. All the discovered action concepts and the associated images are used to train DeViSE model. At test time, the action categories are transformed to vectors using the same word embedding matrix. The prediction of an input image is the argmax of the matching scores between the image and the test categories.

Visual-semantic alignment [23]: This baseline method is similar to the model in VSA [23]. However, we use regular LSTM instead of BRNN to encode the input description. The image is processed by VGG-16 and 4096 dimensional fc7 vector is extracted as image-level feature. The training data are image-description pairs. The output of the model is a confidence score which indicates whether the image and description are matched. The test image is matched with all action categories and the prediction is the category with the highest score.

Action Concept Tree (ACT). There are totally over 100 action concepts (*i.e.* leaf action nodes) discovered in the training set of VGA. These action concepts are clustered into a 4-layer action concept tree (ACT). Due to the limited space, we can't illustrate the whole ACT. Some nodes in ACT are shown

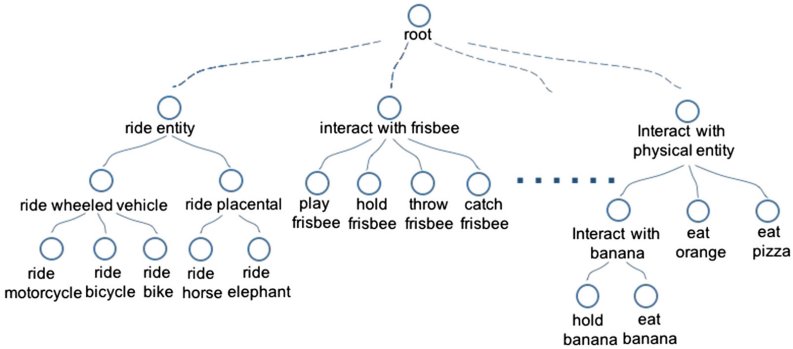


Fig. 4. Example nodes in ACT.

in Fig. 4. Under the node “ride entity”, we can see that “ride motorcycle”, “ride bicycle” and “ride bike” are clustered together and the automatically generated node name is “ride wheeled vehicle”, as “wheeled vehicle” is the lowest common hypernym of “bike”, “bicycle” and “motorcycle” in WordNet. Under the node “interact with frisbee”, there are four leaf nodes: “catch frisbee”, “hold frisbee”, “play frisbee” and “throw frisbee”. They have common object “frisbee” but different verb actions, so we generate the father node name as “interact with frisbee”. The node “interact with physical entity” is illustrated as a poor case of our naming strategy. The child nodes have no common verb and the lowest common hypernym of the objects in WordNet is “physical entity”, therefore the father node is named as “interact with physical object”, which is a very vague action name. Although, the naming strategy is not ideal in this case, the cluster itself still represents one meaningful action category: “interact with food”.

Action Classification Results. The experimental results on VGA are shown in Table 2. From the results, we can see that our 2-stage learning method outperforms several baseline methods. Training models with only stage-1 or stage-2 would lower the performance. Stage-1 only learns general image-description matching knowledge and it does not optimize the model to a specific action classification task; on the other hand, without stage-1, stage-2 optimizes the model from random parameters and it may overfit on such a small dataset of language. Using the hierarchical structure of action concepts (*i.e.* ASA (Stage1+2, w/ ACT)) brings a 1.7% improvement, compared with the flat structure of action concepts (*i.e.* ASA (Stage1+2, w/o ACT)). We believe the reason is that ACT and the node matching algorithm together provide a better way to organize and search for the generalized and detailed knowledge of actions. For example, compared with the flat action concept structure, the test category “brush tooth” is matched not only with the node of “brush tooth”, but also with the parent node of “hold toothbrush” and “brush tooth” in ACT, which allows ASA to use the additional data provided by “hold toothbrush”.

In Fig. 5, some example predictions are shown. We can see that failure could happen when subtle human-object interaction differences are involved; for example, “hold sandwich” and “hold banana” have the same verb action (*i.e.* hold) and visually similar objects.

Table 2. Comparison of different methods on the VGA action classification test set

Method	mAP(%)	R@1(%)	R@5(%)
ACD+SVM+AdaBoost	20.2	24.5	56.3
ACD+DeViSE	22.1	25.1	54.2
VSA	15.9	18.1	47.3
ASA (Stage 1)	20.1	25.3	56.5
ASA (Stage 2)	18.5	24.6	50.4
ASA (Stage 1+2, w/o ACT)	26.8	29.6	60.4
ASA (Stage 1+2, w/ ACT)	28.5	31.3	63.2

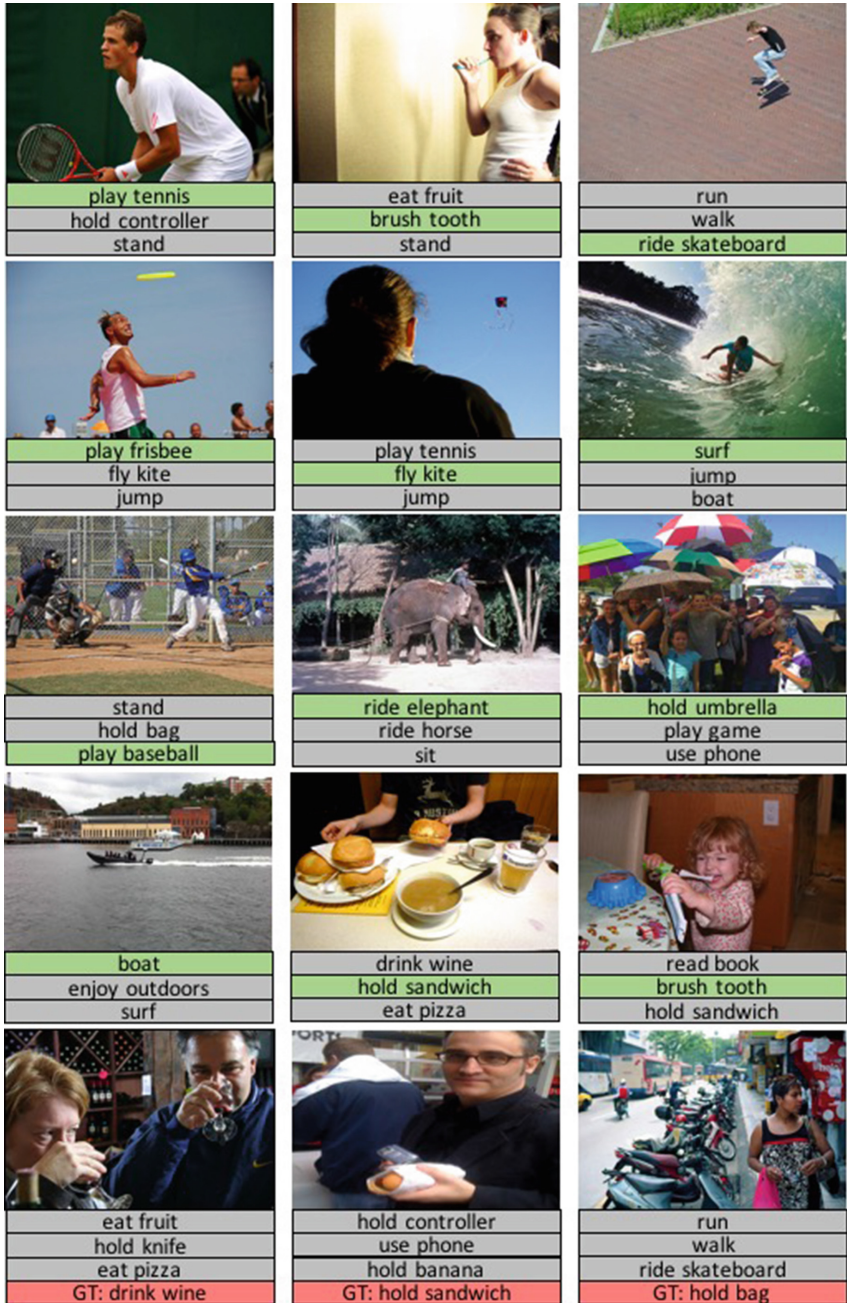


Fig. 5. Prediction examples of the top 3 results on VSA test set. The first four rows are positive examples and green represents the condition when the prediction matches the ground truth. The last row shows some failure cases. (Color figure online)

Table 3. Transfer learning from Flickr30k to PASCAL VOC 2012 action classification test set (AP%).

method	jump	phone	instr.	read	bike	horse	run	photo	comp.	walk	mAP
ACD [18]	62.2	15.4	78.8	29.6	84.5	85.9	60.8	24.0	69.2	32.4	54.3
ASA+ACT	63.5	15.5	80.9	28.9	86.7	92.0	60.7	24.1	69.3	30.9	55.2

4.2 Experiments on Flickr30k and PASCAL VOC

We use the same experiment setup as ACD [18]: using Flickr30k [20] as source image-description dataset and PASCAL VOC 2012 action classification as target test dataset. Flickr30k contains 30000 images and each image is captioned by 5 sentences. PASCAL VOC 2012 action classification dataset has 10 action categories. We train our full model (ASA + ACT) on Flickr30k and apply the action concepts on PASCAL VOC.

As shown in Table 3, our method outperforms ACD [18] in most categories and by 0.9% in mAP. For example, “ride bike” and “ride horse” are two separate subcategories in our ACT and provide more precise data for training, while ACD [18] may cluster these two with other categories such as “ride skateboard”.

5 Conclusion

We presented a two-stage learning framework to learn an Action Concept Tree (ACT) and an Action Semantic Alignment (ASA) model from image-description data. Stage-1 has two steps: (a) ACT is discovered and built by H-ACD algorithm, each node in the tree contains an action name and the relevant images; (b) ASA model is trained by image-description mapping task for parameter initialization. In stage two, we adjust the ASA model to a specific action classification task. The first step is to match the target action categories to the action nodes in ACT discovered in stage-1. After matching, we use the associated data to fine-tune ASA model to this action classification task. Experimental results show that our model outperforms several baseline methods significantly.

Acknowledgement. This research was supported, in part, by the Office of Naval Research under grant N00014-13-1-0493. We would like to thank Chen Sun for valuable discussions.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
2. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
3. Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.J., Shamma, D.A., Bernstein, M., Fei-Fei, L.: Visual genome: connecting language and vision using crowdsourced dense image annotations (2016)

4. Guo, G., Lai, A.: A survey on still image based human action recognition. *Pattern Recogn.* **47**, 3343–3361 (2014)
5. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: *CVPR* (2014)
6. Gkioxari, G., Hariharan, B., Girshick, R., Malik, J.: R-CNNs for pose estimation and action detection. *arXiv preprint [arxiv:1406.5212](https://arxiv.org/abs/1406.5212)* (2014)
7. Gkioxari, G., Girshick, R., Malik, J.: Contextual action recognition with R*CNN. In: *ICCV* (2015)
8. Chao, Y.W., Wang, Z., He, Y., Wang, J., Deng, J.: Hico: A benchmark for recognizing human-object interactions in images. In: *ICCV* (2015)
9. Ramanathan, V., Li, C., Deng, J., Han, W., Li, Z., Gu, K., Song, Y., Bengio, S., Rossenber, C., Fei-Fei, L.: Learning semantic relationships for better action retrieval in images. In: *CVPR* (2015)
10. Rohrbach, A., Rohrbach, M., Tandon, N., Schiele, B.: A dataset for movie description. In: *CVPR* (2015)
11. Torabi, A., Pal, C., Larochelle, H., Courville, A.: Using descriptive video services to create a large data source for video annotation research. *arXiv preprint [arxiv:1503.01070](https://arxiv.org/abs/1503.01070)* (2015)
12. Alayrac, J.B., Bojanowski, P., Agrawal, N., Sivic, J., Laptev, I., Lacoste-Julien, S.: Learning from narrated instruction videos (2016)
13. Ramanathan, V., Liang, P., Fei-Fei, L.: Video event understanding using natural language descriptions. In: *ICCV* (2013)
14. Yu, S.I., Jiang, L., Hauptmann, A.: Instructional videos for unsupervised harvesting and learning of action examples. In: *ACM MM* (2014)
15. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: a large video database for human motion recognition. In: *ICCV* (2011)
16. Reddy, K.K., Shah, M.: Recognizing 50 human action categories of web videos. *Mach. Vis. Appl.* **24**, 971–981 (2013)
17. Sun, C., Gan, C., Nevatia, R.: Automatic concept discovery from parallel text and visual corpora. In: *ICCV* (2015)
18. Gao, J., Sun, C., Nevatia, R.: ACD: action concept discovery from image-sentence corpora. In: *ICMR* (2016)
19. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) *ECCV 2014*. LNCS, vol. 8693, pp. 740–755. Springer, Heidelberg (2014). doi:[10.1007/978-3-319-10602-1_48](https://doi.org/10.1007/978-3-319-10602-1_48)
20. Young, P., Lai, A., Hodosh, M., Hockenmaier, J.: From image descriptions to visual denotations: new similarity metrics for semantic inference over event descriptions. *TACL* **2**, 67–78 (2014)
21. Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: *CVPR* (2015)
22. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: a neural image caption generator. In: *CVPR* (2015)
23. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. In: *CVPR* (2015)
24. Hu, R., Xu, H., Rohrbach, M., Feng, J., Saenko, K., Darrell, T.: Natural language object retrieval. In: *CVPR* (2016)
25. Hu, R., Rohrbach, M., Darrell, T.: Segmentation from natural language expressions. *arXiv preprint [arxiv:1603.06180](https://arxiv.org/abs/1603.06180)* (2016)

26. Miller, G.A.: Wordnet: a lexical database for english. *Commun. ACM* **38**, 39–41 (1995)
27. Cao, S., Chen, K., Nevatia, R.: Abstraction hierarchy and self annotation update for fine grained activity recognition. In: *WACV* (2016)
28. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: large-scale machine learning on heterogeneous systems (2015). <https://www.tensorflow.org/>
29. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: *CVPR* (2009)
30. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *NIPS* (2013)
31. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. In: *ICLR* (2015)
32. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: a library for large linear classification. *JMLR* **9**, 1871–1874 (2008)
33. Frome, A., Corrado, G.S., Shlens, J., Bengio, S., Dean, J., Mikolov, T., et al.: Devise: a deep visual-semantic embedding model. In: *NIPS* (2013)

Computer Vision – ACCV 2016

13th Asian Conference on Computer Vision, Taipei,
Taiwan, November 20–24, 2016, Revised Selected
Papers, Part II

Lai, S.-H.; Lepetit, V.; Nishino, K.; Sato, Y. (Eds.)

2017, XIII, 436 p. 156 illus., Softcover

ISBN: 978-3-319-54183-9