Modular Action Concept Grounding in Semantic Video Prediction

Wei Yu\textsuperscript{1,2}, Wenxin Chen\textsuperscript{1,2}, Songheng Yin\textsuperscript{1,2}, Steve Easterbrook\textsuperscript{1}, Animesh Garg\textsuperscript{1,2,3}
\textsuperscript{1}University of Toronto, \textsuperscript{2}Vector Institute, \textsuperscript{3}Nvidia

Abstract

Recent works in video prediction have mainly focused on passive forecasting and low-level action-conditional prediction, which sidesteps the learning of interaction between agents and objects. We introduce the task of semantic action-conditional video prediction, which uses semantic action labels to describe those interactions and can be regarded as an inverse problem of action recognition. The challenge of this new task primarily lies in how to effectively inform the model of semantic action information. Inspired by the idea of Mixture of Experts, we embody each abstract label by a structured combination of various visual concept learners and propose a novel video prediction model, Modular Action Concept Network (MAC). Our method is evaluated on two newly designed synthetic datasets, CLEVR-Building-Blocks and Sapien-Kitchen, and one real-world dataset called Tower-Creation. Extensive experiments demonstrate that MAC can correctly condition on given instructions and generate corresponding future frames without need of bounding boxes. We further show that the trained model can make out-of-distribution generalization, be quickly adapted to new object categories and exploit its learnt features for object detection, showing the progression towards higher-level cognitive abilities. More visualizations can be found at http://www.pair.toronto.edu/mac/.

1. Introduction

Recently, video prediction has drawn a lot of attention due to its ability to capture meaningful representations through self-supervision [37, 42]. Although modern video prediction methods have made significant progress in improving predictive accuracy, most of their applications are limited in the scenarios of passive forecasting [5, 18, 33, 36], meaning models can only passively observe a short period of dynamics and accordingly make a short-term extrapolation. Such settings neglect the fact that the observer can also become an active participant in the environment.

To model the movements of active manipulators, several low-level action-conditional video prediction models have been proposed in the community [2, 8, 15, 25, 26, 27]. In this work, we go one step further by introducing the task of \textit{semantic action-conditional video prediction} which emphasizes the modeling of interactions between agents and environment. Instead of using low-level single-entity actions such as action vectors of robot arms as done in prior works [9, 22], our new task provides semantic descriptions of interactive actions, e.g. "Open the door", and asks the model to imagine "What if I open the door" in the form of future frames. This task requires the model to recognize the object identity, assign correct affordances to objects and envision the long-term expectation by planning a reasonable trajectory toward the goal, which resembles how humans might imagine conditional futures. The ability to predict correct and semantically consistent future perceptual information is indicative of conceptual grounding of actions, in a manner similar to object grounding in image-based detection and generation tasks.

The challenge of action-conditional video prediction primarily lies in how to correctly inform the model of more abstract semantic action information. Existing low-level counterparts usually achieve this by employing a naive concatenation [2, 9] with action vector of each timestep. While this implementation might enable model to move the desired objects, it fails to produce consistent long-term predictions.
2. Approach

We begin with defining the task of semantic action-conditional video prediction. Given an initial frame $x_0$ and a sequence of action labels $a_{1:T}$, the model is required to predict the corresponding future frames $x_{1:T}$. Each action label is a pre-defined semantic description of a spatiotemporal movement that involves multiple objects in a scene and spans over multiple frames such as "take the yellow cup on the table" from $t = 0$ to $t = 10$. So technically, one can regard this task as an inverse problem of action recognition. It should also be pointed out that our semantic task is different from common dense video prediction and generation tasks in the sense that it focuses on predicting time-agnostic events. Hence, we design the corresponding datasets as videos capturing sufficient key frames of entire actions. In future practices, we can further apply video interpolation methods in CV or motion planner algorithms in RL to make up the intermediate process if needed.

2.1. Motivation

The design of our new task is necessary for studying compositional generalization as it detaches the definition of object from its specific location. However, it also requires a successful model to figure out where the desired object is through leveraging abstract labels. Our main idea is that we create a large number of small, specialized and relatively independent learners called concept slots for each word in the dictionary of action labels to capture their corresponding spatial representations from observations. During training, action labels will be translated as constituency trees to control the activations of all related concept slots and to assemble the representations of given actions for next-frame prediction. As a result, this language-guided gating mechanism embeds the syntactic structures into the learning system and enables the proposed model to dynamically recombine its learnt concepts so that it can understand the combinatorial complexity of the world. In this paper, we demonstrate that our method possesses many key characteristics of system-2 learning \cite{10, 11}, including concept grounding, sample efficiency, counterfactual generations, out-of-distribution generalization and fast transfer.

2.2. Modular Action Concept Network

The MAC model is composed of 4 modules including encoder $E$, decoder $D$, concept slot module $C$ and recurrent predictor $P$. The goal of our model is to learn the following mapping:

$$\hat{x}_t = D(P(C(E(x_{t-1})|a_t)|h_{t-1}))$$  \hspace{1cm} (1)

where $x_t$, $a_t$ and $h_t$ are video frame, action labels and hidden states at time $t$. The overall architecture of our method is illustrated in Fig 2. In the case of stochastic video generation, another two modules, prior $p(z)$ and posterior $q(z)$, will be added to help estimate the latent distribution of trajectories.

**Encoder and Decoder:** At each timestep $t - 1$, the encoder $E$ receives visual input $x_{t-1}$ and extracts a set of multi-scale feature maps. In the deterministic setting, we employ a convolutional neural network with an architecture similar to DCGAN \cite{28}. The matching decoder $D$ is a mirrored version of the encoder with down-sampling operations.
Figure 2. The pipeline of MAC in which the computation of concept slot module is elaborated (Better viewed in color). Feature maps extracted by encoder are mapped into the concept slot tensors. Concept slot module receives an action label that controls the collection of concept slot tensors and outputs representations encapsulating this action. A recurrent predictor updates representations before sending them to decoder to predict the next frame.

Concept Slot Module: The concept slot module $C$ is the core module of MAC. It resembles the mixture of experts as each slot focuses on only one concept in the space of action labels and will be activated and assembled to represent the given actions through the language-guided gating functions.

Each atomic action label will first be decomposed into several constituents of sentence. A constituent is a verb or object phrase, like “pick” or “large red bowl”. Since we are mostly dealing with manipulation videos, atomic actions are usually divided into 3 constituents, verb, object$_1$, object$_2$, and more complex multiple-entity actions can be expanded into temporal sequences of several atomic two-object actions. For single-entity actions, object$_2$ will be filled with all zero tensors. Each constituent will have its own dictionary recording all pre-defined words or concepts and gating functions can be derived based on these dictionaries to establish bottom-up connections from concept slots. The computation of concept slot module is given as follows:

$$w^i = \Psi^i(f), c^j = \Phi^j(\text{Concat}(\{w^i|\forall i, \delta^j(i) = 1\}))$$  (2)

where $w$ and $c$ are concept and constituent representations and $\delta^j$ is the indicator function for gating function of $j$th constituent. More specifically, after the feature maps $f$ are extracted from the input image, they are fed into $K$ convolutional units $\Psi^i$, i.e. the concept slot layer, to create $K$ concept slot tensors of dimension $N_d$. Here, $K$ is the total number of possible concepts we pre-defined in the dictionary of action labels. Since verbs can be interpreted as spatiotemporal changes of relationships between objects, not only slots for objects but also slots for verbs, like ’take’ or ’put on’, are computed from the extracted feature maps.

Next, a gating function will collect all involved concept slot tensors and create an ensemble as input for each constituent. This assembly process simulates the formation of simplified constituency parse trees. Constituent slot layer $\Phi^j$ can either be resolution-preserving or upsampling operators as spatial information is important for our new task. Finally, outputs of all constituent slots are concatenated pixel-wisely to obtain the representation of actions before sending them to predictor. It is worth noticing that MAC is allowed to have multiple concurrent actions in a scene at inference time. In this case, we copy additional groups of trained constituent slots to represent other actions.

In the stochastic setting, we use invertible autoencoder introduced in CrevNet [42] instead as we find this information-preserving architecture can better preserve the attributes of randomly moving objects. The corresponding decoder is the backward pass, i.e. inverse computation, of the same network of the encoder. Readers can find more details about invertible autoencoder and coupling layer in Appendix B.
where $p$ can be initialized with randomly assigned relative positions of target frames to estimate a true distribution that we expect its prior counterpart $p(z)$ to mimic at test time.

**Learned Prior**: We leverage a technique called learned prior (Fig 3) from SVG [7] to model the stochastic movements in videos. In particular, we build two additional recurrent inference networks, prior and posterior respectively, to capture the randomness of motions. During training, the posterior inference network $q(z)$ can access to the representations of target frames to estimate a true distribution of trajectory that we expect its prior counterpart $p(z)$ to mimic at test time. Codes of motions $z_t$ estimated by posterior during training (or by prior during testing) will then be concatenated with latent representations before sent to predictor.

**Predictor**: The recurrent predictor $P$, implemented as a stack of residual ConvLSTM layers [31], calculates the spatiotemporal evolution for each action label respectively. The memory mechanism of ConvLSTM is essential for MAC predictions as visual inputs instead except for the first pass. To prevent interference between concurrent actions, hidden states are not shared between actions. The outputs of predictor for all action labels are added point-wisely.

**Training**: In the deterministic setting, we train our model by minimizing the mean squared error the between the target frames and the predictions. In the stochastic setting, we optimize the following variational lower bound (ELBO) using re-parameterization trick [21]:

$$L_{θ,ϕ,ψ}(x_{1:T}) = \sum_{t=1}^{T} \mathbb{E}_{q_θ(z_{1:t},x_{1:t})} \log p_θ(x_t|z_{1:t},x_{1:t-1})$$

$$- \beta D_{KL}(q_ψ(z_t|x_{1:t})||p_ψ(z_t|x_{1:t-1}))$$

(3)

where $p_θ$ is the future frame generator, $z_t$ represents the latent codes of motion, $p_ψ(z_t|x_{1:t-1})$ is the prior distribution, $q_θ(z_{1:t}|x_{1:t})$ is the posterior distribution and $D_{KL}$ denotes the Kullback–Leibler (KL) divergence which forces the posterior to approximate the prior distribution. Since $p_θ$ is modeled by conditional Gaussian, the likelihood term reduces to MSE measure between the ground truth frames and the predictions. The full derivation of ELBO is provided in the Appendix A.

At the inference phase, the model will use its previous predictions as visual inputs instead except for the first pass. Hence, a training strategy called scheduled sampling [4] is adopted to alleviate the discrepancy between training and inference.

### 3. Datasets

In this study, we create two new synthetic datasets, CLEVR-Building-blocks and Sapien-Kitchen, and label one real-world dataset called Tower-Creation from Roboturk [24] for evaluation. This is because most existing video datasets either don’t come with semantic action labels [2] or fail to provide necessary visual information in their first frames due to egomotions and occlusions [16]. Although there are several candidate datasets like Penn Action [44], BAIR [9] and KTH [30] for multi-modal learning, they all adopt the same single-entity setting which actually indicates they can be solved by a much simpler model. To tackle the above issues, we design each video in our datasets as a depiction of certain atomic action performed by an agent with objects which are observable in the starting frame. Furthermore, we add functions to generate bounding boxes of all objects for both synthetic datasets in order to train AG2Vid. It is worth noting that all three of these domains exhibit a key property named combinatorial explosion, resulting in factorial complexity growth in both spatial and temporal dimensions even with a small object set. For instance, a sequence with 6 (out of 32) objects and 6 actions can have 333,396,000 possibilities without considering any continuous factor. Hence, our model only sees a small fraction of these potential scenarios during training.

#### 3.1. CLEVR-Building-blocks Dataset

CLEVR-Building-blocks dataset is built upon CLEVR environment [19]. For each video, the data generator initializes the scene with 4 - 6 randomly positioned and visually different objects. There are totally 32 combinations of shapes, colors and materials of objects and at most one instance of each combination is allowed to appear in a video sequence. The agent can perform one of the following 8 actions on objects $O_A$ and $O_B$: Pick $O_A$, Pick and Rotate $O_A$ transversely / longitudinally, Put $O_A$ on $O_B$, Put $O_A$ on the left / right side of $O_B$, Put $O_A$ in the front of / behind $O_B$. Each training sample contains a video of three consecutive Pick- and Put- action pairs and a sequence of semantic action labels of every frame.

#### 3.2. Sapien-Kitchen Dataset

Sapien-Kitchen Dataset describes a more complicated environment in the sense that: (a). It contains deformable actions like "open" and "close"; (b). The structures of different objects in the same category are highly diverse; (c). Objects can be initialized with randomly assigned relative positions like "along the wall" and "on the dishwasher". We collect totally 21 types of small movable objects in 3 categories,
bottle, kettle and kitchen pot, and 19 types of large openable appliances in another 3 categories, oven, refrigerator and dishwasher, from Sapien engine [41]. The agent can perform one of the following 6 atomic actions on small object $O_s$ and large appliance $O_l$: Take $O_s$ on $O_l$, Take $O_s$ in $O_l$, Put $O_s$ on $O_l$, Put $O_s$ in $O_l$, Open $O_l$ and Close $O_l$. Composite action sequences are defined as follows: "Take_on–Put_on", "Take_on–Open–Put_in–Close", "Open–Take_in–Close".

### 3.3. Tower-Creation Dataset

Each video in Tower-Creation Dataset depicts a robotic arm building a tower with flatware present on the table. We have labeled 524 videos in total since semantic descriptions are not provided and produce 1867 samples consists of two actions: Pick $O_A$ and Put $O_A$ on $O_B$. We use 1536 video clips for training and 331 for evaluation. It should be pointed out that the size of Tower-Creation dataset is small compared with commonly used datasets such as BAIR [9] which has 59k videos in total. Thus, our experiments can also tell whether evaluated methods are data efficient.

### 4. Experimental Evaluation

#### 4.1. Action-conditional video prediction

**Baselines and setup:** We evaluate the proposed model on CLEVR-Building-blocks and Sapien-Kitchen Datasets. AG2Vid [3] is re-implemented as the baseline model because it is the most related work. Every action graph used in AG2Vid can be equivalently translated into our cases because each atomic action graph in AG2Vid also involves at most two objects. But unlike our method which only needs visual input and action sequence, AG2Vid also requires bounding boxes of all objects for training and testing and it can only handle deterministic prediction. Furthermore, we conduct an ablation study by replacing concept slot module with the concatenation of features and tiled action vector, which is commonly used in low-level action-conditional video prediction [9], to show the effectiveness of our module.

**Metrics:** To estimate the fidelity of action-conditional video prediction, MSE, SSIM [38], PSNR and LPIPS [43] are calculated between the predictions and groundtruths. FID [13] and FVD [32] are not appropriate for this task because they cannot tell how faithfully the model obeys the given instructions. However, these metrics still may not effectively tell if actions are successfully completed due to the small sizes of the moving objects. Hence, we also perform a human study to assess the accuracy of performing the correct action in generated videos for each model. The human judges annotate whether the model can identify the desired objects, perform actions specified by action labels and maintain the consistent visual appearances of all objects in its generations and only videos meeting all three criterions are scored as correct. Also, we find it is technically infeasible to train an action recognition model to estimate the accuracy due to the innumerable action labels caused by the property of combinatorial explosion.

**Results:** The quantitative comparisons of all methods are summarized in Table 1. The MAC achieves the best scores on all metrics without access to additional information like bounding boxes, showing the superior performance of our concept slot module. The qualitative analysis in Fig 4 further reveals the drawbacks of other baselines. For CLEVR-Building-blocks, the concatenation-based variant fails to recognize the right objects due to its limited inductive bias. Although AG2Vid has no difficulty in identifying the desired objects, assumptions made by flow warping are too strong to handle rotation and occlusion. Consequently, the adversarial loss enforces AG2Vid to fix these errors by converting them to wrong poses or colors. These limitations of AG2Vid will be further amplified in a more complicated environment, i.e. Sapien-Kitchen. The same architecture used for CLEVR can only learn to remove the moving objects from their starting positions in Sapien-Kitchen because rotation and occlusion occur more often. The concatenation baseline performs bet-
4.2. Counterfactual generation

**Counterfactual generation**: The most intriguing application of MAC is counterfactual generation. More specifically, counterfactual generation means that our model will observe the same starting frame but receive different valid action labels to produce the corresponding future frames.

**Results**: The visual results of counterfactual generations on each dataset are displayed in Fig 5. As we can see, our model successfully identifies the desired objects, plans correct trajectories toward the target places and generates high-quality imaginations of counterfactual futures. It is also worth noticing that all displayed generations are long-term generations, i.e., more than 30 frames are predicted for each sequence. Our recurrent predictor plays an very important role in sustaining the spatiotemporal consistency and in reconstructing the fully-occluded objects.

4.3. Stochastic video generation

**Baselines and setup**: We continue to evaluate the stochastic version of MAC (sMAC) on Tower-Creation dataset. SVG-LP was extended to action-conditional version in [34] so that we can adopt it as the baseline model to demonstrate the effectiveness of concept slot module.

**Results**: The qualitative and frame-wise quantitative comparison between sMAC and action-conditional SVG-LP is provided in Fig 6. Although SVG-LP can partially understand the given action labels, it often fails to locate and manipulate the desired objects. Consequently, it will generate the moving object out of nowhere and often place it on a wrong target object. In contrast, sMAC can successfully simulate the trajectory of robotic arms and correctly animate the "Pick" and "Put" actions thanks to the concept slot module. Row 3 and 5 in Fig 6 show that sMAC is also capable of producing diverse future frames and predicting counterfactual results following different action instructions. The overall accuracy of sMAC estimated by human study on Tower-Creation is 65.3% compared with 31.8% of SVG-LP.

4.4. Compositional Generalization

We further explore other interesting features of our MAC. We first demonstrate that MAC is capable of making out-of-distribution generalization by designing two experiments. We evaluate how quickly our model can be adapted to new...
Figure 6. **Left**: Visual comparison between sMAC and SVG-LP on Tower-Creation. The supposed completions of *Pick* and *Put* in the final frames are highlighted by red and yellow boxes while incorrect completions in SVG-LP generations are labelled by grey boxes. The last two rows are counterfactual generations in which models are given different action labels. **Right**: Quantitative comparison per-frame. Higher SSIM and PSNR indicate better performance.

Figure 7. Compositional generalization and feature reuse.

**Top**: Unobserved scenarios. All red cubes are removed from the training data, but the trained model can still manipulate red cube at test time.

**Middle**: Concurrent actions. Inputting two action sequences at the same time. Both actions are depicted correctly.

**Bottom Left**: New-object adaptation. Even with a few training samples, MAC can be fast adapted for generation of new objects. Red arrows point to new objects present in images.

**Bottom Right**: Object detection. Learnt features can be directly applied for detection.

objects. It turns out for each new object, the trained MAC only requires a few training video examples to generate decent results. Finally, to verify that our model encodes the spatial information, we add SSD [23] head after the frozen encoder and concept slot layer to conduct object detection.

**Unobserved scenarios**: We design an interesting experiment where only a subset of CLEVR-Building-blocks data are used for training and check what will happen if we input the unobserved action labels to the trained model. More precisely, we exclude all videos manipulating red cubes in the training sets and send the instructions involving red cubes at test time. Note that we only remove one object to avoid high correlation among concept slots, otherwise it will violate the relative independence assumption. Since failure cases will not manipulate the correct objects and will produce very large pixelwise losses. We can set a threshold of MSE to calculate the accuracy of performing the correct actions, which is 75.6%. The visualization of this experiment can be found...
in Fig 7. As we can see, MAC can still identify and manipulate red cubes correctly, showing its ability to recombine the learnt concept to comprehend new objects.

**Concurrent actions:** Concurrent actions means multiple action inputs at the same time. It can be considered as out-of-distribution generalization because our model only observes single-action videos during training. Generating concurrent-action videos needs to employ copied constituent slots and parallel hidden states. As illustrated in Fig 7, MAC can linearly integrate the action information in the latent space and correctly portray 2 concurrent actions in the same scene.

**Adaptation:** We add a new openable category "safe" and a new movable category "dispenser" into Sapien-Kitchen and generate 100 video sequences for each new object showing its interaction with other objects. Approximately, there are about 5 new sequences created for each new action pair between 2 objects. Blank concept slots for new categories are attached to trained MAC and we finetune it on this small new training set. Visualization in Fig 7 shows that even with a few training samples, MAC is accurately adapted for video generation of new objects. This is because, with the help of concept slots, MAC can disentangle actions into relatively independent grounded concepts. When it learns new concepts, MAC reuses and integrates prior knowledge learnt from different cases.

**Object detection:** The quantitative results of object detection and more visualizations can be found in Appendix D. We observe that the features learnt by MAC can be easily transferred for detection as our video prediction task is highly location-dependent. This result indicates that utilizing bounding boxes might be a little redundant for some video tasks because videos already provide rich motion information that can be used for salient object detection.

5. Related Work

**Video prediction:** ConvLSTM [31] was the first deep learning model that employed a hybrid of convolutional and recurrent units for passive video prediction. This architectural design was soon followed by studies looking at a similar problem [25, 35, 37, 42]. However, the capability of passive video prediction framework is very limited as models usually don’t have sufficient information to predict the long-term future due to partial observation, egomotion and randomness. It also prevents models from interacting with environment.

On the other hand, the low-level action-conditional video prediction task provides an action vector at each timestep as additional input to guide the prediction [2, 6, 27, 39]. CDNA [9] is a representative of such models. In CDNA, the states and action vectors of the robotic manipulator are first spatially tiled and integrated into the model through concatenation. SVG [7] was initially proposed for stochastic video generation but later was extended to action-conditional version in [34]. SVG also used concatenation to incorporate action information. Such implementations are prevalent in low-level prediction because the action vector only encodes the spatial information of a single entity, usually a robotic manipulator [9] or a human hand. A common failure case for such models is the presence of multiple entities [20], a scenario that our task definition and datasets focus on.

**Modularity:** Mixture of Experts refers to a classical machine learning technique where various learners are employed, each of which specializes in one particular function, and their output are aggregated through a gating function. This modular design makes each submodule relatively independent and thus leads to better generalization and robustness to compositional changes, which has been studied in several works [1, 11, 12, 29]. In this work, we hypothesis that the underlying syntactic structures of semantic labels can tell how to aggregate the representations of individual concept learners. By translating labels into constituency trees, action graphs are embedded into the learning system to get the entire perspective of ongoing activities while each concept learner can focus on its specific subtask.

6. Limitations

While the results of MAC are very impressive, there are still several limitations to this work, including (a). **Uniqueness:** We didn’t design specific mechanisms that enable MAC to randomly select one of repeated objects. We assume each object is unique in the scene. (b). **More flexible semantic instructions:** In this work, we use semantic labels pre-defined in a relatively fixed format. Therefore, we can translate each label to a constituency tree without using any learnable function. (c). **Ego-motion:** All videos we evaluated on were captured by fixed cameras. Videos with ego-motions can provide a more abundant source of training data.

7. Conclusion

In this work, we propose the new task of semantic action-conditional video prediction and introduce 3 new datasets that are meant to bridge the gap towards a robust solution to this task in complex interactive scenarios. MAC, a novel video prediction model, was also designed by utilizing the idea of MoE to ground action concept for video generation. Our proposed model can generate alternative futures without requiring additional auxiliary data such as bounding boxes, and is shown to be both quickly extendible and adaptable to novel scenarios and entities. It is our hope that our contributions will advance progress and understanding within this new task space, and that a model robust enough for real-world applications (i.e. in robotic systems) in control will be eventually proposed as a descendant of this work.

**Acknowledgement** This work is supported by CIFAR AI Chair, NSERC Discovery Award, University of Toronto XSeed award, and gifts from LG.
References


