Generating the Future with Adversarial Transformers

Carl Vondrick and Antonio Torralba
Massachusetts Institute of Technology
{vondrick,torralba}@mit.edu

Abstract

We learn models to generate the immediate future in video. This problem has two main challenges. Firstly, since the future is uncertain, models should be multi-modal, which can be difficult to learn. Secondly, since the future is similar to the past, models store low-level details, which complicates learning of high-level semantics. We propose a framework to tackle both of these challenges. We present a model that generates the future by transforming pixels in the past. Our approach explicitly disentangles the model's memory from the prediction, which helps the model learn desirable invariances. Experiments suggest that this model can generate short videos of plausible futures. We believe predictive models have many applications in robotics, health-care, and video understanding.

1. Introduction

Can you predict what the scene in Figure 1a will look like in the immediate future? The capability for machines to anticipate the future would enable several applications in robotics, health-care, and video understanding [13, 14]. Unfortunately, despite the availability of large video datasets and advances in data-driven learning methods, robust visual prediction models have been elusive.

We believe there are two primary obstacles in the future generation problem. Firstly, the future is uncertain [18, 35, 36]. In order to produce sharp generations, models must account for uncertainty, but multi-modal losses can be difficult to optimize. Secondly, the future is often similar to the past [5, 45], which models consequently must store. However, memorizing the past may complicate the learning of high-level semantics necessary for prediction. In this paper, we propose a framework to tackle both challenges.

We present an adversarial network that generates the future by transforming the pixels in the past. Rather than generating pixel intensities [28, 18, 35] (which may be too unconstrained) or generating fixed representations [47, 34, 36] (which may be too constrained), we propose a model that learns to transform the past pixels. This formulation untangles the memory of the past with the prediction of the future. We believe this formulation helps the network learn desirable invariances because each layer in the network is no longer required to store low-level details. Instead, the network only needs to store sufficient information to transform the input. Our experiments and visualizations suggest that generating transformations produces more realistic predictions and also helps learn some semantics.

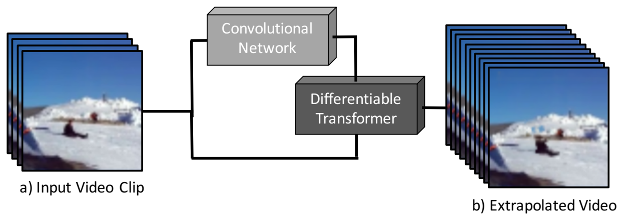

Figure 1: **Generating the Future:** We develop a large-scale model for generating the immediate future in unconstrained scenes. Our model uses adversarial learning to predict a transformation from the past into the future by learning from unlabeled video.

Since the future is uncertain, we instead train our model to generate a plausible future. We leverage recent advances in adversarial learning [7, 26] to train our model to generate one possible video of the future. Although the model is not guaranteed to generate the "correct" future, instead our approach hallucinates transformations for a future that is plausible. Experiments suggest that humans prefer predictions from our model better than simple baselines.

We capitalize on large amounts of unlabeled video downloaded from the web for learning. Although unlabeled video lacks labels, it contains rich signals about how objects behave and is abundantly available. Our model is trained end-to-end without supervision using unconstrained, in-the-wild data from consumer video footage.

The main contribution of this paper is the development of a large-scale approach for generating videos of the future by learning transformations with adversarial learning and unconstrained unlabeled video. The remainder of this

Figure 2: **Unlabeled Video:** We learn models from large amounts of unconstrained and unlabeled video to train models to generate the immediate future.

paper describes our approach in detail. In section 3, we describe the unlabeled video dataset we use for learning and evaluation. In section 4, we present our adversarial network for learning transformations into the future. In section 5, we present several experiments to analyze adversarial networks for future generation.

2. Related Work

Visual Anticipation: Our work builds upon several works in both action forecasting [13, 14, 34, 6, 50, 38] and future generation [28, 18, 35, 37, 47, 50, 11, 17, 45, 51, 36]. While a wide body of work has focused on predicting actions or motions, our work investigates predicting pixel values in the future, similar to [28, 18, 35]. However, rather than predicting unconstrained pixel intensities, we seek to learn a transformation from past pixels to the future. Prior work has explored learning transformations in restricted domains [5, 45], such as for robotic arms or clip art. In this paper, we seek to learn transformations from in-the-wild unlabeled video from consumer cameras.

Visual Transformations: This paper is also related to learning to understand transformations in images and videos [8, 9, 49, 39, 45, 5]. We also study transformations, but focus on learning the transformations for predicting the future in unconstrained and unlabeled video footage.

Generative Adversarial Models: Our technical approach takes advantage of advances in generative adversarial networks [7, 26, 2, 41, 25]. However, rather than generating novel images, we seek to generate videos conditioned on past frames. Our work is an instance of conditional generative adversarial networks [19, 18, 35, 25]. However, in our approach, the generator network outputs a transformation on the condition, which may help stabilize learning.

Neural Memory Models: Our work extends work in neural memory models, such as attention networks [44, 43], memory networks [42, 30], and pointer networks [33]. Our approach uses a similar idea as [33] to generate the output by pointing to the the inputs, but we apply it to vision instead. Our network learns to point to past pixels to produce the transformation into the future.

Unlabeled Video: Our work is related to a growing body of work that leverages massive amounts of unlabeled video for visual understanding, such for representation learning and cross-modal transfer [15, 40, 29, 10, 20, 21, 1, 27, 22, 24, 16, 34]. In our work, we use large amounts of unlabeled video for learning to generate the immediate future.

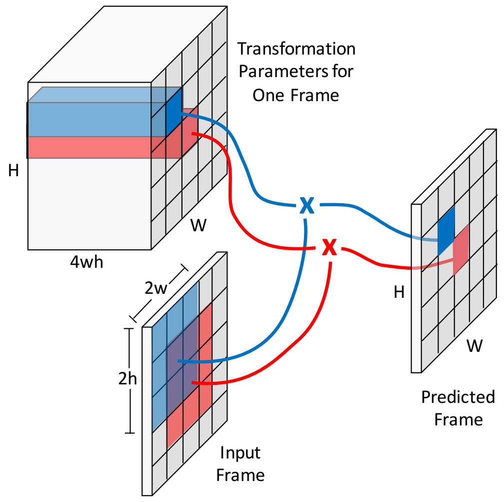

Figure 3: **Transformations:** For each (x, y, t) coordinate in the future, the network estimates a weighted combination of neighboring pixels from the input frame to render the predicted frame. The $\times$ denotes dot product. Note the transformation is applied by convolution.

3. Dataset

We use large amounts of unlabeled video from Flickr [32, 35] for both training and evaluation. This dataset is very challenging due to its unconstrained and "in-the-wild" nature. The videos depict everyday situations (e.g., parties, restaurants, vacations, families) with an open-world of objects, scenes, and actions. We download over $500,000$ videos, which we use for learning and evaluation. Given 4 frames as input, we aim to extrapolate the next 12 frames at full frame-rate into the future (for a total of 16 frames).

We do little pre-processing on the videos. As we are interested object motion and not camera motion, we stabilize the videos using SIFT and RANSAC. If the camera moves out of the frame, we fill in holes with neighboring values. We focus on small videos of 64×64 spatial resolution and 16 frames, consistent with [26, 35]. We scale the intensity to between -1 and 1. In contrast to prior work [35], we do not filter videos by scene categories.

4. Method

We present an approach for generating the immediate future in video. Given an input video clip $x \in \mathbb{R}^{t \times W \times H}$, we wish to extrapolate a future video clip $y \in \mathbb{R}^{T \times W \times H}$ where

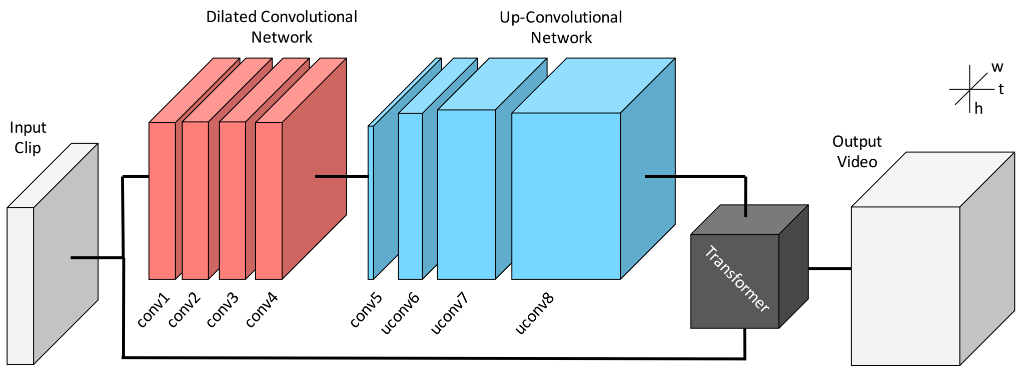

Figure 4: **Network Architecture:** We illustrate our convolutional network architecture for generating the future. The input clip goes through a series convolutions and nonlinearities that preserve resolution. After integrating information across multiple input frames (if multiple), the network up-samples temporally into the future. The network outputs codes for a transformation of the input frames, which produces the final video. For details on the transformer, see Figure 3.

	`conv1`	`conv2`	`conv3`	`conv4`	`conv5`	`uconv6`	`uconv7`	`uconv8`
Num Filters	32	64	128	256	32	32	32	15^2
Filter Size	$1 \times 3 \times 3$	$1 \times 3 \times 3$	$1 \times 3 \times 3$	$1 \times 3 \times 3$	$4 \times 1 \times 1$	$4 \times 1 \times 1$	$4 \times 1 \times 1$	$4 \times 1 \times 1$
Dilation	$1 \times 1 \times 1$	$1 \times 2 \times 2$	$1 \times 4 \times 4$	$1 \times 8 \times 8$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$
Padding	$0 \times 1 \times 1$	$0 \times 2 \times 2$	$0 \times 4 \times 4$	$0 \times 8 \times 8$	$0 \times 0 \times 0$	$1 \times 1 \times 1$	$2 \times 1 \times 1$	$2 \times 1 \times 1$
Stride	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$2 \times 1 \times 1$	$2 \times 1 \times 1$

Table 1: **Network Details:** We describe our network architecture in detail. The input is a $4 \times 64 \times 64$ clip. The output of `uconv8` is a $15^2 \times 12 \times 64 \times 64$ transformation code, which is fed into the transformer, producing a $12 \times 16 \times 16$ video. The dimensions are Time $\times$ Width $\times$ Height format.

t, T are durations in frames, and W and H are width and height respectively. We design a deep convolutional network $f(x; \omega)$ for the video extrapolation task.

One strategy for predicting the future is to create a network that directly outputs y, such as [18, 35]. However, since the future is similar to the past, the model will need to store low-level details (e.g., colors or edges) about the input x at every layer of representation. Not only is this inefficient use of network capacity, but it may make it difficult for the network to learn desirable invariances that are necessary for future prediction (such as parts or object detectors). Consequently, we wish to develop a model that untangles the memory of the past from the prediction of the future.

4.1. Generating Transformations

Rather than directly predicting the future, we design f to output a transformation from the past to the future:

$$f(x; \omega) = \gamma(g(x; \omega), x) \tag{1}$$

where γ is transformation function and g is a convolutional neural network to predict these transformations. Since the input x is available to the transformation function at the end, g does not necessarily need to store low-level details about the image. Instead, g only needs to store information sufficient to transform x.

We employ a simple transformation model by interpolating between neighboring pixels [5]. The output pixel at location (i, j) in frame t is given by the inner product:

$$\gamma_{i,j,t}(g, x) = g_{i,j,t}^T \cdot x_{i-w:i+w,j-h:j+h} \tag{2}$$

where $x_{a:b,c:d} \in \mathbb{R}^{4wh}$ selects the block in the image x from (a, c) to (b, d), and flattens it to a vector. The transformation is applied relatively on the original input image, and each pixel can undergo different transformations from its neighbors. The hyper-parameters w and h define the receptive field of the transformation. $g_{i,j,t} \in \mathbb{R}^{4wh}$ produces the coefficients for each neighboring pixel, which we normalize to be positive and sum to unity. The model can support larger receptive fields at the expense of extra learnable parameters. We handle border-effects by padding with replication. We visualize the operation in Figure 3.

While the transformation could be applied recurrently on each frame, errors will accumulate, which may complicate learning. We instead apply the transformation relative to the input frame, which has the advantage of making the model more robust for longer time periods. While this requires a larger receptive field for the transformations, the increase in extra parameters is negligible compared to the amount of training data available (virtually unlimited).

Since we do not have ground truth labels to supervise g, we can train the prediction model f end-to-end because the transformation γ is differentiable, allowing us to use back-propagation. Moreover, this enables us to train the model without human supervision.

4.2. Adversarial Learning

While we could train $f(x; \omega)$ to regress the future y (e.g. with ℓ_2 loss), the model would be unable to handle the multi-modal nature of the problem [34, 18, 36], which often manifests as blurry predictions due to the regression-to-mean problem. Instead, we use a multi-modal loss.

Rather than training the network to predict the correct future (which may be a poorly defined task), we instead train the network to predict one plausible future. To do this, we take advantage of adversarial learning for video [18, 35, 50]. We simultaneously train a discriminator network $d(x, y)$ to distinguish real videos from generated videos. The predictor f seeks to maximally fool the discriminator d. Specifically, during learning, we optimize:

$$\min_{\rho} \max_{\omega} \sum_i L\left(d(x_i, y_i; \rho), 1\right) + L\left(d(x_i, f(x_i; \omega)\,; \rho), -1\right) \tag{3}$$

where L is the binary cross-entropy loss and ± 1 specifies the target category (real or not).

We use a deep spatio-temporal convolutional network as the discriminator network, similar to [35]. Since the discriminator receives both the input x and the future y, the network first concatenates x and y along the time dimension, which is fed into the rest of network. Consequently, the prediction network f can only fool the discriminator if it predicts a future video consistent with its input.

Several works have attempted to use adversarial learning for video prediction [18, 35, 50], however due to the instability of adversarial learning, they typically also use a combination of losses (such as regression or total variation losses). In this paper, we are able to only use an adversarial objective with unconstrained data.

4.3. Convolutional Network Architecture

We use a convolutional network to parametrize g. Since we desire to make a dense prediction for each space-time location, we design g to preserve the resolution of the input throughout the network. To capture long-range spatial dependencies, we employ dilated convolutions [46] that exponentially increase their receptive field size and maintain spatial resolution. To up-sample in time, we use a up-convolutional temporal network. We visualize the network architecture for f in Figure 4 and provide the complete configuration in Table 1.

4.4. Optimization

We optimize Equation 3 with mini-batch stochastic gradient descent. We alternate between one update of the discriminator and one update of the generator. We use a batch size of 32. During learning of the generator, maximizing ω often does not provide a strong gradient for learning, so we instead optimize $\min_\omega \sum_i L\left(d(x_i, f(x_i; \omega)\,; \rho), 1\right)$ in the generator, similar to [7]. We use the Adam optimizer [12] with a learning rate of 0.0002 and momentum term of 0.5. We train each model for $50,000$ iterations, which typically takes two days on a GPU. We use batch normalization on every layer in both the generator and discriminator. We use rectified linear units (ReLU) for the generator and leaky ReLU for the discriminator, following [26]. We generate videos that are 64×64 in spatial resolution that are up to 16 frames long at full frame (a little under a second of clock time). We use Torch7.

5. Experiments

In this section, we present experiments to analyze and understand the behavior of our model.

5.1. Experimental Setup

We split our dataset into $470,824$ video clips for training, and $52,705$ video clips for testing. The clip are split by source video, so clips from the same video are part of the same partition. Our evaluation follows advice from [31], which recommends evaluating generative models for the task at hand. Since our model is trained to generate plausible futures, we use a human psychological study to eval-

		Not Preferred				
		Adv+Tra	Reg+Tra	Adv+Int	Reg+Int	Real
Preferred	Adv+Tra	-	**55.6**	**61.2**	**55.1**	30.6
	Reg+Tra	44.4	-	**60.8**	**54.1**	36.4
	Adv+Int	38.8	39.2	-	39.6	37.3
	Reg+Int	44.9	45.9	**60.4**	-	38.0
	Real	**69.4**	**63.6**	**62.7**	**62.0**	-

Table 2: **Future Generation Evaluation:** We ask workers on Mechanical Turk the two-alternative forced choice question "Which video has more realistic motion?" and report the percentage of times that subjects preferred a method over another. Rows indicate the method that workers preferred when compared to the one of the columns. For example, workers preferred the Adv+Tra method over the Reg+Tra method 55.6% of the time. Adv is for Adversarial, Tra is for Transformation, Reg is for Regression, and Int is for Intensity. Overall, predicting transformations with adversarial learning tends to produce more realistic motions.

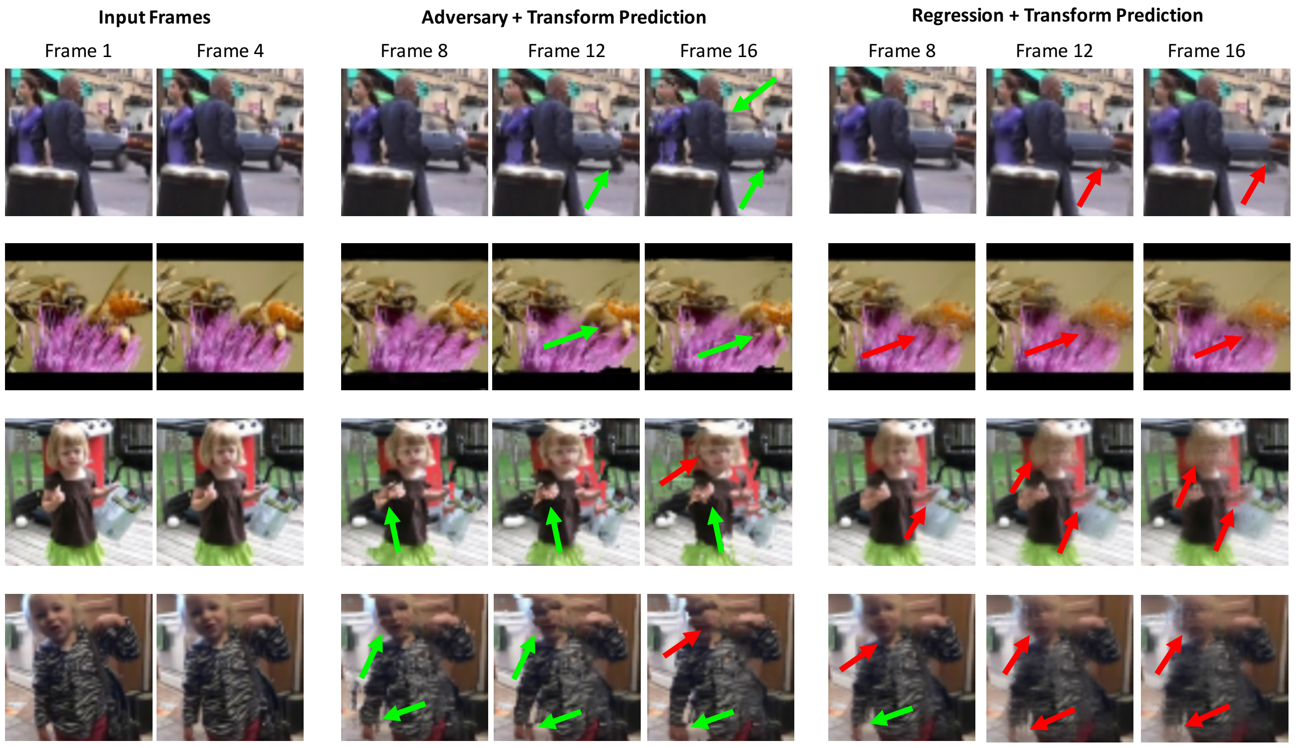

Figure 5: **Qualitative Video Generation Results:** We visualize some of the generated videos. The input is 4 frames and the model generates the next 12 frames. We qualitatively compare generations from adversary + transformation models versus regression + transformation models. The green arrows point to regions of good motion, and the red arrows point to regions of bad motion. The regression model typically adds motion to the scene, but it quickly becomes blurry. The adversary usually has sharper motion. Best viewed on screen.

uate our predictions, similar to [23, 35]. We use workers from Amazon Mechanical Turk to answer a two-alternative forced choice (2AFC) question. We show workers two videos generated by different methods, and ask them to answer "Which video has more realistic motion?" If workers tend to choose videos from a particular method more often, then it suggests that that method generates more realistic motion according to human subjects. We solicit $1,000$ opinions and pay workers 1 cent per decision. We required workers to have an historical approval rating of 95% to help ensure quality. We experimented with disqualifying workers who incorrectly said real videos were not real, but this did not change the relative ranking of methods.

5.2. Baselines

We compare our method against unsupervised future generation baselines.

Adversarial with Pixel Intensities: Rather than generating transformations, we could try to directly generate the pixel intensities of the future. To do this, we remove the transformation module and modify our network g to output 3-channels (for RGB) with tanh activation. We then train with adversarial learning. This is similar to [35, 18].

Regression with Transformations: Rather than learning with adversarial learning, we can instead train the model only with a regression loss. We use ℓ_1 loss.

Regression with Pixel Intensities: We also compared a regression model that directly regresses the pixel intensities, combining both of the previous two baselines.

Real Videos: Finally, we compare against the true video of the future. While we do not expect to generate photorealistic videos, it allows us to measure how indistinguishable our generations are from real (if at all).

5.3. Future Generation

Table 2 reports subjects preferences for different methods in the two-alternative force choice for generating 16 frame videos given the first 4 input video frames. Overall, generating transformations instead of pixel intensities produces more realistic motions in generating the immediate future. The adversarial learning with transformations tends to produce sharper motions, which workers found preferable to the baselines. We also compared generated videos versus real videos. As one might expect, workers usually

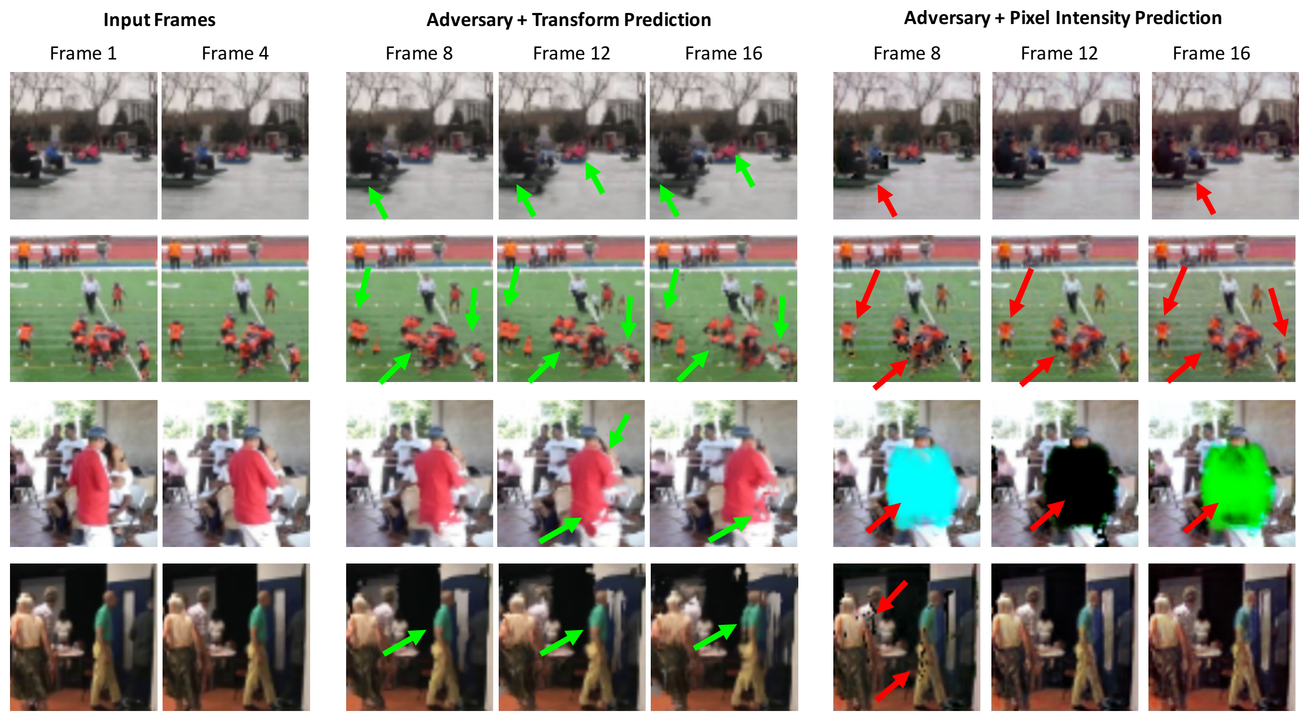

Figure 6: **Qualitative Video Generation Results:** We visualize some of the generated videos. The input is 4 frames and the model generates the next 12 frames. We qualitatively compare generations from adversary + transformation models versus adversary + pixel intensity models. The green arrows point to regions of good motion, and the red arrows point to regions of bad motion. The intensity model typically struggles to add any motion, often changing colors instead. Best viewed on screen.

prefer real videos over synthetic videos.

We show several qualitative examples of the generations in Figure 5 and Figure 6. We summarize a few our qualitative observations. The transformation models (both regression and adversary) tend to have the most motion, likely because the network is more efficiently storing the input. However, regression transformations tend to be blurry over longer time periods due to the regression-to-the-mean problem. The adversarial network that directly generates pixel intensities generally struggles to create motion and instead tends to change the colors, likely due to the instability of adversarial learning. However, adversarial learning with transformations may provide sufficient constraints to the output space that the generator learns efficiently. Overall, the adversarial network with transformations tend to produce motion that is sharper because the model seeks a mode of the future rather than the mean-of-modes.

We also visualize some of the internal transformations in Figure 7. The transformation parameters are colored by the direction and distance that pixels move. The visualization shows that the model often learns to transform edges rather than entire objects, likely because moving edges sufficiently fool the adversary. Moreover, the motion often is associated with objects, suggesting learning transformations may be a good signal for learning about objects.

5.4. Analyzing Invariances

We hypothesize that learning to generate transformations into the future helps the network learn desirable invariances that are useful for prediction and higher-level understanding. To evaluate the degree to which the network has learned any semantics, we experimented with fine-tuning our network for an object recognition task.

We use the PASCAL VOC object classification dataset [4] using the provided train/val splits. We cut the network from the input until `conv4` and fine-tune it to classify into the 20 object categories in PASCAL VOC. However, since our network preserves resolution, we must make one change to produce a category distribution output. We add a global max pooling layer after `conv4` to down-sample the $256 \times 64 \times 6$ hidden activations to a 256 dimensional vector, and add a linear layer to produce the output. We train the network with multi-class cross entropy loss.

We report performance in Table 3 using mean average

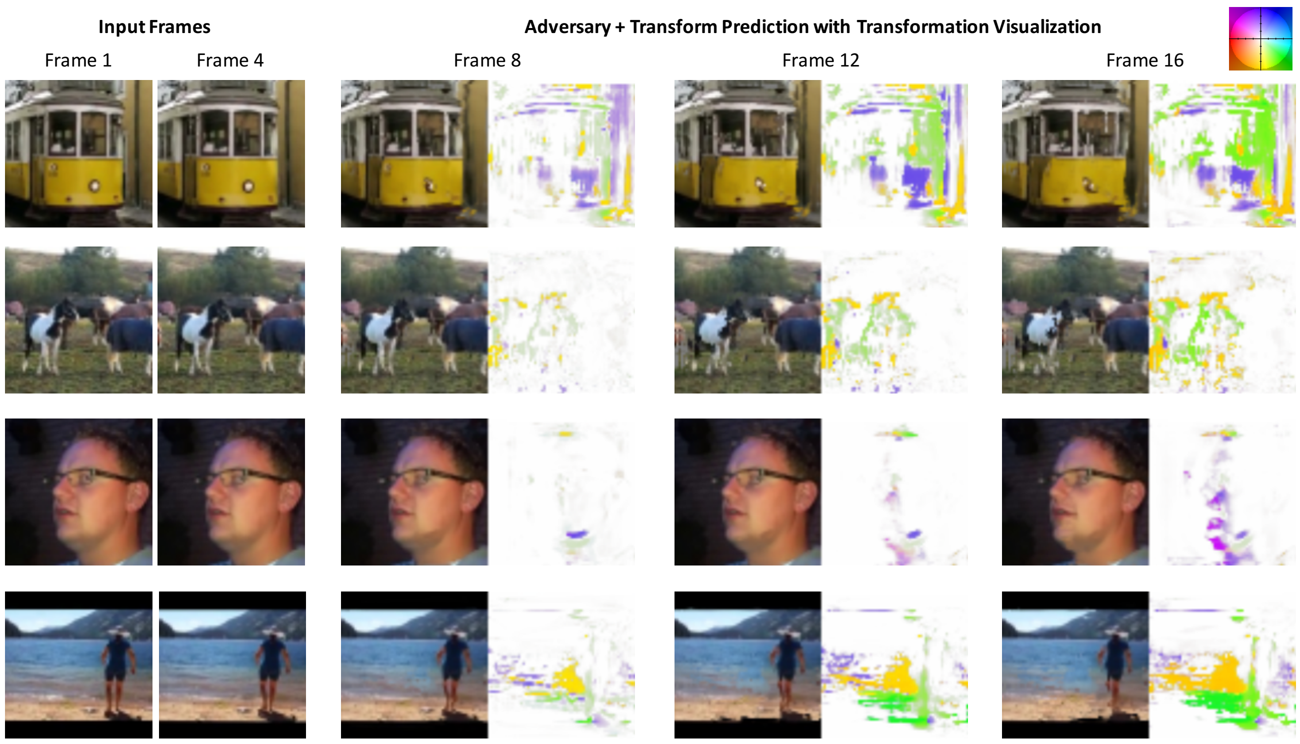
Input Frames
Adversary + Transform Prediction with Transformation Visualization
Frame 1
Frame 4
Frame 8
Frame 12
Frame 16

Figure 7: **Visualizing Inferred Transformation:** We visualize the transformation generated by our full model. Colors indicate direction that pixels move relative to the input frame under the transformation. The model learns to mix neighboring pixels of the input to generate the future frames.

Method	2007 mAP	2012 mAP
Chance	7.3	7.2
Random Initialization	26.7	30.6
Regression + No Transform	30.0	33.6
Adversary + No Transform	29.7	33.3
Regression + Transform	**32.6**	**38.8**
Adversary + Transform	**32.0**	**38.1**

Table 3: **Object Classification:** We experiment how well our prediction network learns semantics by fine-tuning them for a object recognition task with a little training data. We report mean average precision on the object classification task for PASCAL VOC without any additional supervision. Note we only compare to methods that classify low-resolution images (64×64).

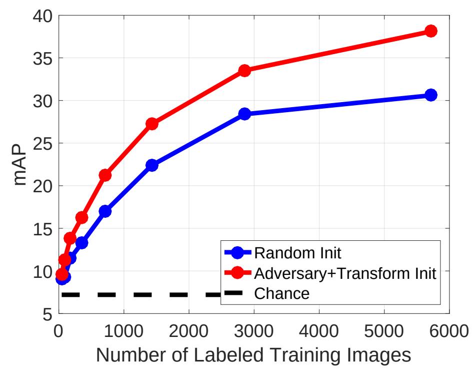

Figure 8: **Performance vs Labeled Dataset Size:** We analyze the performance (mean average precision) on object classification on PASCAL VOC 2012 for a network initialized with our future predictor versus random initialization. We obtain the same performance as the baseline using only **one third** of the labeled data.

precision. While all methods trained to predict the future outperform a randomly initialized network, our results suggest that learning transformations provides a better signal for learning semantics than directly producing pixel intensities. We believe this is because the memory of the past is decoupled from the prediction of the future, which allows the hidden representation to be robust to low-level signals (such as color) that are not as useful for object recognition.

We also analyzed the performance versus the amount of labeled training data. Figure 8 shows performance of our full model versus a network randomly initialized. For all dataset sizes, our approach outperforms a randomly initial-

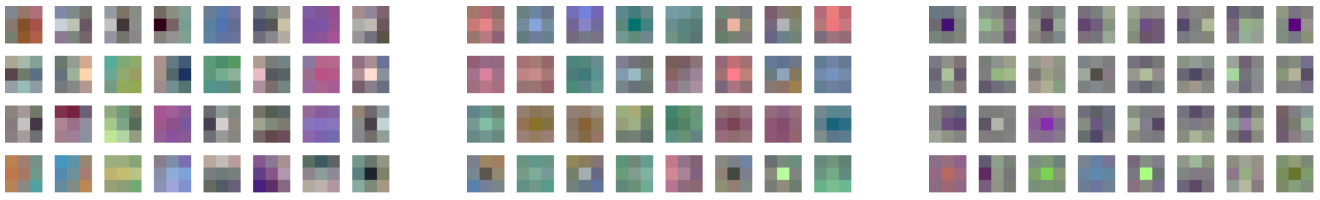
(a) Transform + Adversary (b) No Transform (c) No Adversary

Figure 9: **Visualizing conv1 Filters:** We visualize the learned filters in `conv1` of our network compared to baselines. (**a**) Our full network (adversary+transformation) learns simple edge and gradient detectors in the first layer. (**b**) Training without transformations causes the network to learn color detectors, rather than gradient detectors, because the baseline network now needs to store the input, which complicates learning of desirable invariances. (**c**) Training without the adversary learns some edge detectors, but not as rich as our full model.

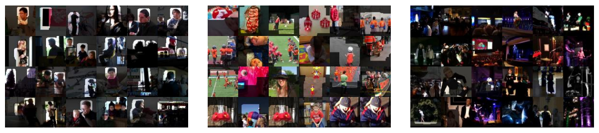
(a) Face-like Unit (b) Sports-like Unit (c) Show-like Unit

Figure 10: **Hidden Unit Visualization:** We visualize some of the hidden units in our prediction network. We feed images through our network and highlight regions that maximally activate a particular unit, similar to [48]. Since the network predicts transformations, some of the units are invariant to low-level features such as colors. Instead, they tend to be selective for patterns indicative of motion.

ized network. Interestingly, our results suggest that our full approach only needs one third of the labeled data to match performance as the randomly initialized network.

Although the goal in this paper is not to learn visual representations, these experiments suggest that transformation learning is a promising supervisory signal. Since transformation learning is orthogonal and complementary to [3, 24], scaling-up future generation could be a fruitful direction for learning representations. Our experiments are conducted on smaller images than usual (64×64 versus 224×224), and higher-resolution outputs may enable richer predictions.

5.5. Visualization

To better understand what our prediction network learns, we visualize some of the internal layers. Figure 9 visualizes the learned weights of the first convolutional layer of different models. Interestingly, our full model (with adversarial learning and transformations) learns several edge and gradient detectors. However, the baseline model without transformations tends to learn simple color detectors, which may happen because the network now needs to store the input throughout the layers. In contrast, the transformation model can learn more abstract representations because it does not need to store the past.

Figure 10 visualizes several hidden units in `conv4` by highlighting regions of input images that maximally activate a certain convolutional unit, similar to [48]. Some of the units are selective for higher-level objects, such as sporting events, music performances, or faces. In contrast, when we visualized the networks that directly predict intensity, the hidden units were highly correlated with colors. These visualizations suggest that learning to predict transformations helps learn desirable invariances.

6. Conclusion

We presented a framework to learn to generate the immediate future in video by learning from large amounts of unconstrained unlabeled video. Our approach tackles two challenges in future generation: handling the uncertainty of the future, and handling the memory of the past. We proposed a model that untangles the memory of the past from the prediction of the future by learning transformations. Experiments and visualizations suggest that learning transformations helps produce more realistic predictions as well as helps the model learn some semantics. Our results suggest that explicit memory models for future prediction can yield better predictions and desirable invariances.

Acknowledgements: Funding for this work was partially provided by the Google PhD fellowship to CV. We acknowledge NVidia Corporation for hardware donations.

References

[1] C.-Y. Chen and K. Grauman. Watching unlabeled video helps learn new human actions from very few labeled snapshots. In *CVPR*, 2013.

[2] E. L. Denton, S. Chintala, R. Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, 2015.

[3] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015.

[4] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[5] C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. *arXiv*, 2016.

[6] K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik. Recurrent network models for human dynamics. In *ICCV*, 2015.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014.

[8] P. Isola, J. J. Lim, and E. H. Adelson. Discovering states and transformations in image collections. In *CVPR*, 2015.

[9] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *NIPS*, 2015.

[10] D. Jayaraman and K. Grauman. Learning image representations tied to ego-motion. In *ICCV*, 2015.

[11] N. Kalchbrenner, A. v. d. Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu. Video pixel networks. *arXiv*, 2016.

[12] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv*, 2014.

[13] K. Kitani, B. Ziebart, J. Bagnell, and M. Hebert. Activity forecasting. *ECCV*, 2012.

[14] H. S. Koppula and A. Saxena. Anticipating human activities using object affordances for reactive robotic response. *TPAMI*, 2016.

[15] Q. V. Le. Building high-level features using large scale unsupervised learning. In *CASSP*, 2013.

[16] Y. Li, M. Paluri, J. M. Rehg, and P. Dollár. Unsupervised learning of edges. *arXiv*, 2015.

[17] W. Lotter, G. Kreiman, and D. Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv*, 2016.

[18] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv*, 2015.

[19] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv*, 2014.

[20] I. Misra, C. L. Zitnick, and M. Hebert. Shuffle and Learn: Unsupervised Learning using Temporal Order Verification. In *ECCV*, 2016.

[21] H. Mobahi, R. Collobert, and J. Weston. Deep learning from temporal coherence in video. In *ICML*, 2009.

[22] P. X. Nguyen, G. Rogez, C. Fowlkes, and D. Ramanan. The open world of micro-videos. *arXiv*, 2016.

[23] A. Owens, P. Isola, J. McDermott, A. Torralba, E. H. Adelson, and W. T. Freeman. Visually indicated sounds. *arXiv*, 2015.

[24] A. Owens, J. Wu, J. H. McDermott, W. T. Freeman, and A. Torralba. Ambient sound provides supervision for visual learning. *arXiv*, 2016.

[25] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. *arXiv*, 2016.

[26] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv*, 2015.

[27] V. Ramanathan, K. Tang, G. Mori, and L. Fei-Fei. Learning temporal embeddings for complex video analysis. In *CVPR*, 2015.

[28] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra. Video (language) modeling: a baseline for generative models of natural videos. *arXiv*, 2014.

[29] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. *arXiv*, 2015.

[30] S. Sukhbaatar, J. Weston, R. Fergus, et al. End-to-end memory networks. In *NIPS*, 2015.

[31] L. Theis, A. v. d. Oord, and M. Bethge. A note on the evaluation of generative models. *arXiv*, 2015.

[32] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li. Yfcc100m: The new data in multimedia research. *ACM*, 2016.

[33] O. Vinyals, M. Fortunato, and N. Jaitly. Pointer networks. In *NIPS*, 2015.

[34] C. Vondrick, H. Pirsiavash, and A. Torralba. Anticipating visual representations from unlabeled video. *CVPR*, 2015.

[35] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. *arXiv preprint arXiv:1609.02612*, 2016.

[36] J. Walker, C. Doersch, A. Gupta, and M. Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *ECCV*, 2016.

[37] J. Walker, A. Gupta, and M. Hebert. Patch to the future: Unsupervised visual prediction. In *CVPR*, 2014.

[38] J. Walker, A. Gupta, and M. Hebert. Dense optical flow prediction from a static image. In *ICCV*, 2015.

[39] X. Wang, A. Farhadi, and A. Gupta. Actions ˜ transformations. In *CVPR*, 2016.

[40] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015.

[41] X. Wang and A. Gupta. Generative image modeling using style and structure adversarial networks. *arXiv*, 2016.

[42] J. Weston, S. Chopra, and A. Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[43] H. Xu and K. Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. *arXiv*, 2015.

[44] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2015.

[45] T. Xue, J. Wu, K. L. Bouman, and W. T. Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. *arXiv*, 2016.

[46] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[47] J. Yuen and A. Torralba. A data-driven approach for event prediction. In *ECCV*. 2010.

[48] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene cnns. *arXiv*, 2014.

[49] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. A. Efros. View synthesis by appearance flow. *arXiv*, 2016.

[50] Y. Zhou and T. L. Berg. Temporal perception and prediction in ego-centric video. In *ICCV*, 2015.

[51] Y. Zhou and T. L. Berg. Learning temporal transformations from time-lapse videos. In *ECCV*, 2016.