# GENERALIZABLE FEATURES FROM UNSUPERVISED LEARNING

**Mehdi Mirza & Aaron Courville & Yoshua Bengio**
MILA
Université de Montréal
{memirzamo, aaron.courville, yoshua.umontreal}@gmail.com

## ABSTRACT

Humans learn a predictive model of the world and use this model to reason about future events and the consequences of actions. In contrast to most machine predictors, we exhibit an impressive ability to generalize to unseen scenarios and reason intelligently in these settings. One important aspect of this ability is *physical intuition* (Lake et al., 2016). In this work, we explore the potential of unsupervised learning to find features that promote better generalization to settings outside the supervised training distribution. Our task is predicting the stability of towers of square blocks. We demonstrate that an unsupervised model, trained to predict future frames of a video sequence of stable and unstable block configurations, can yield features that support extrapolating stability prediction to blocks configurations outside the training set distribution.

## 1 INTRODUCTION

Humans learn a tremendous amount of knowledge about the world with almost no supervision and can construct a predictive model of the world. We use this model of the world to interact with our environment. As also argued by Lake et al. (2016) one of the core ingredients of human intelligence is intuitive physics. Children can learn and predict some of the common physical behaviors of our world just by observing and interacting without any direct supervision. And they form a sophisticated predictive model of the physical environment and expect the world to behave based on their mental model and have a reasonable expectation about unseen situations Téglás et al. (2011).

Despite impressive progress in the last few years in the training of the supervised models, we have not yet quite been able to achieve similar results in unsupervised learning, and it remains one of the challenging research areas in the field. The full potential of the application of unsupervised learning is yet to be realized.

In this work, we leverage unsupervised learning to train a predictive model over sequences. We use the imagined and predicted future sequence data to help a physical environment prediction model generalize better to unseen settings.

More specifically we focus on the task of predicting if a tower of square bricks will fall or not, as introduced by Lerer et al. (2016). They showed that a deep convolution neural network could predict the fall of the towers with super-human accuracy. But despite the strengths of convolution neural networks, Zhang et al. (2016) shows how deep neural networks have a hard time generalizing to novel situations in the same way as humans or simulation-based models can do. In this work, we show that deep neural networks are capable of generalizing to novel situations through a form of unsupervised learning. The core idea is to observe the world without any supervision and build a future predictive model of it, and in a later stage leverage and utilize the imagined future to train a better fall prediction model.

## 2 RELATED WORK

In the beginning, unsupervised learning and generative models emerged as pre-training method Hinton & Salakhutdinov (2006); Hinton et al. (2006); Bengio et al. (2007) to help other tasks such as supervised learning. But since Krizhevsky et al. (2012) many other regularization Srivastava et al. (2014), weight initialization Glorot & Bengio (2010) and normalization Ioffe & Szegedy (2015) techniques and architecture designs He et al. (2015) has been introduced that diminish the effect of pre-training. Although pre-training still could be useful in data scarce domains they are many other ways and applications that unsupervised learning are still very interesting models and it is a very active area of research. Just to name a few applications are semi-supervised learning Kingma et al. (2014); Salimans et al. (2016); Dumoulin et al. (2016) super resolution Sønderby et al. (2016).

Video generation is one active area of research with many applications, and many of the recent works have been using some of the states of the art neural networks for video generation. Srivastava et al. (2015) uses LSTM recurrent neural networks to train an unsupervised future predictive model for video generation. And here we use a very similar architecture as described in Section 4.1. Mathieu et al. (2015) combines the common mean-squared-error objective function with an adversarial training cost in order to generate sharper samples. Lotter et al. (2016) introduce another form of unsupervised video prediction training scheme that manages to predict future events such as the direction of the turn of a car which could have potential use in training of the self-driving cars.

Model-based reinforcement learning (RL) is an active research area that holds the promise of making the RL agents less data hungry. Learning agents could explore, learn in an unsupervised way about their world, and learn even more by dreaming about future states. We believe that action-condition video prediction models are an important ingredient for this task. Fragkiadaki et al. (2015) learn the dynamics of billiards balls by supervised training of a neural net. Action-conditioned video prediction models have been applied to Atari playing agent Oh et al. (2015) as well as robotics (Finn et al., 2016; Finn & Levine, 2016).

## 3 Dataset

Recent datasets for predicting the stability of block configurations (Lerer et al., 2016; Zhang et al., 2016) only provide binary labels of stability, and exclude the video simulation of the block configuration. We, therefore, construct a new dataset, with a similar setup as Lerer et al. (2016); Zhang et al. (2016), that includes this video sequence. We use a Javascript based physics engine[1] to generate the data.

We construct towers made of $3-5$ square blocks. To sample a random tower configuration, we uniformly shift each block in its $x$ and $y$ position such that it touches the block below. Because taller towers are more unstable, this shift is smaller when we add more blocks. To simplify our learning setting, we balance the number of stable and unstable block configurations. For each tower height, we create 8000, 1000 and 3000 video clips for the training, validation, and test set, respectively. The video clips are sub-sampled in time to include more noticeable changes in the blocks configurations. We decided to keep 39 number of frames which with our sub-sampling rate was enough time for unstable towers to collapse. Each video frame is an RGB image of size 64x64. In addition to binary stability label, we include the number of blocks that fell down.

## 4 Architecture

The core idea of this paper is to use future state predictions of a generative video model to enhance the performance of a supervised prediction model. Our architecture consists of two separate modules:

**Frame predictor** A generative model to predict future frames of a video sequence. This model is trained to either generate the last frame or the complete sequence of frames.

**Stability predictor** In the original task, stability is predicted from a static image of a block configuration. We explore whether, in addition to initial configuration, the last frame prediction of our unsupervised model improves the performance of the stability prediction.

In the following sections, we explore several different architectures for both modules.

[1] https://chandlerprall.github.io/Physijs/

### 4.1 Future frame prediction

We consider two different model architectures for this task. The first one, named ConvDeconv, only takes the first frame as input and predicts the last frame of the video sequence. The architecture consist of a block of convolution and max-pooling layers. To compensate for the dimensionality reduction of the max-pooling layers, we have a fully-connected layer following the last max-pooling layer. And finally a subsequent block of deconvolution layers with the output size same as the model input size. All activation functions are ReLU(Nair & Hinton, 2010). See Table 1 for more details of the architecture. The objective function is the mean squared error between the generated last frame and the ground-truth frame; as a result, this training will not require any labels. We also experimented with an additional adversarial cost as in Mathieu et al. (2015) but did not observe any improvement for the stability prediction task. We hypothesize that although the adversarial objective function helps to have sharper images, such improved sample quality does not transfer to better stability prediction. Figure 1 shows a few examples of the generated data on the test set. Mean squared error is minimized using the AdaM Optimizer(Kingma & Ba, 2014) and we use early-stopping when the validation loss does not improve for 100 epochs.

We extend this ConvDeconv model in a second architecture, named ConvLSTMDeconv, to predict the next frame at each timestep. This model is composed of an LSTM architecture. The same convolutional and deconvolutional blocks as ConvDeconv is utilized to respectively input the current frame to the LSTM transition and output the next frame from the current LSTM state. The details of the ConvLSTMDeconv model architecture are shown in Table 2 and Figure 3 shows the diagram of the both architectures. During the training at each time step the ground-truth data feeds in to the model, but during the test time only the initial time step gets the first frame from the data and for subsequent time steps the generated frames from the previous time steps feed in to the model. The is similar setup to recurrent neural network language models Mikolov (2012), and this is necessary as during the test time we only have access to the first frame. As before, the model is trained to predict the next frame at each time step by minimizing the predictive mean-squared-error using AdaM optimizer and early-stopping. For training, we further subsample in time dimension and reduce the sequence length to 5-time steps. Figure 2 shows some sample generated sequences from the test set.

| Layer | Type | Output channels/dimensions | Kernel/Pool size |
|---|---|---|---|
| 1 | Conv | 64 | $3 \times 3$ |
| 2 | MaxPool | 64 | $4 \times 4$ |
| 3 | Conv | 128 | $3 \times 3$ |
| 4 | MaxPool | 64 | $3 \times 3$ |
| 5 | Conv | 64 | $3 \times 3$ |
| 6 | MaxPool | 64 | $3 \times 3$ |
| 7 | FC | $64 \times 64 \times 16 = 65536$ | |
| 8 | DeConv | 64 | $3 \times 3$ |
| 9 | DeConv | 128 | $3 \times 3$ |
| 10 | DeConv | 64 | $3 \times 3$ |
| 11 | DeConv | 3 | $3 \times 3$ |

Table 1: ConvDeconv model architecture. FC stands for "Fully Connected".

| Layer | Type | Output channels/Dimension | Kernel/Pool size |
|---|---|---|---|
| 1 | Conv | 64 | $3 \times 3$ |
| 2 | MaxPool | 64 | $4 \times 4$ |
| 3 | Conv | 128 | $3 \times 3$ |
| 4 | MaxPool | 64 | $3 \times 3$ |
| 5 | Conv | 64 | $3 \times 3$ |
| 6 | MaxPool | 64 | $3 \times 3$ |
| 7 | FC LSTM | 2000 | |
| 8 | FC | $64 \times 64 \times 3$ | |
| 9 | DeConv | 64 | $3 \times 3$ |
| 10 | DeConv | 64 | $3 \times 3$ |
| 11 | DeConv | 3 | $3 \times 3$ |

Table 2: ConvLSTMDeconv model architecture. FC stands for "Fully Connected".

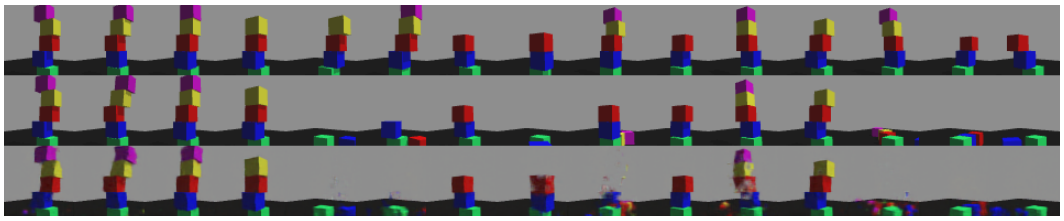

Figure 1: Samples from the ConvDeconv model. First and second rows show first and last frame respectively from the test data. And the third row shows generated last frame samples.

### 4.2 Stability prediction

We have two supervised models for stability prediction. The first one will be a baseline that takes as input the first frame and predict the fall of the tower. For this model we use 50 layer ResNet

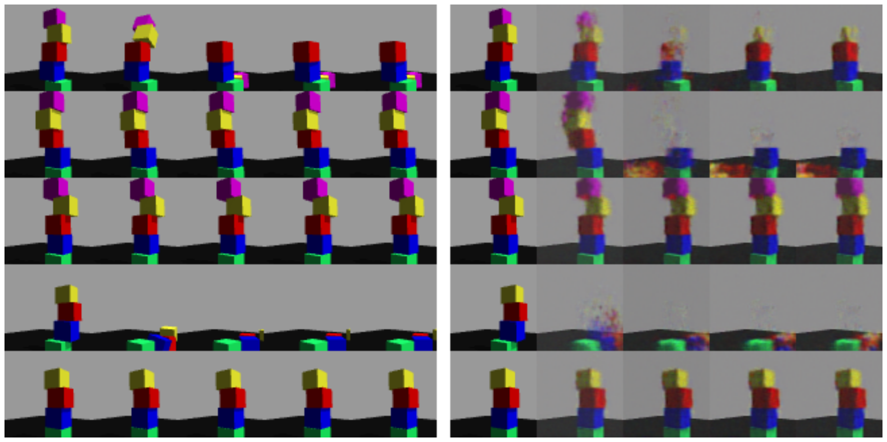

Figure 2: Samples from the ConvLSTMDeconv model. Each row is a different sample. The left sequence is the data and the right sequence is the generated data. Note that during generation model only see the first frame and for next time steps uses its own output from the last timestep.

architecture from He et al. (2016). We trained the baseline model on each of the different tower heights 3, 4, 5. We call it the single model and name experiments 3S, 4S, 5S respectively for the number of blocks that it was trained on. The second model will be the one using the generated data: it takes as input the first frame and the generated last frame. It consisted of two 50 Layer ResNet blocks in parallel, one for the first frame and one for last frame and the last hidden layer of both models are concatenated together before a logistic regression layer (or Softmax in the case of non-binary labels). Both ResNet blocks share parameters. Based on whether the generated data is coming from ConvDeconv model or ConvLSTMDeconv model we labeled experiments as 3CD, 4CD, 5CD and 3CLD, 4CLD, 5CLD respectively.

None of the models are pre-trained and all the weights are randomly initialized. As in 4.1, we use AdaM and we stopped the training when the validation accuracy was not improved for 100 epochs. All images are contrast normalized independently and we augment our training set using random horizontal flip of the images and randomly changing the contrast and brightness.

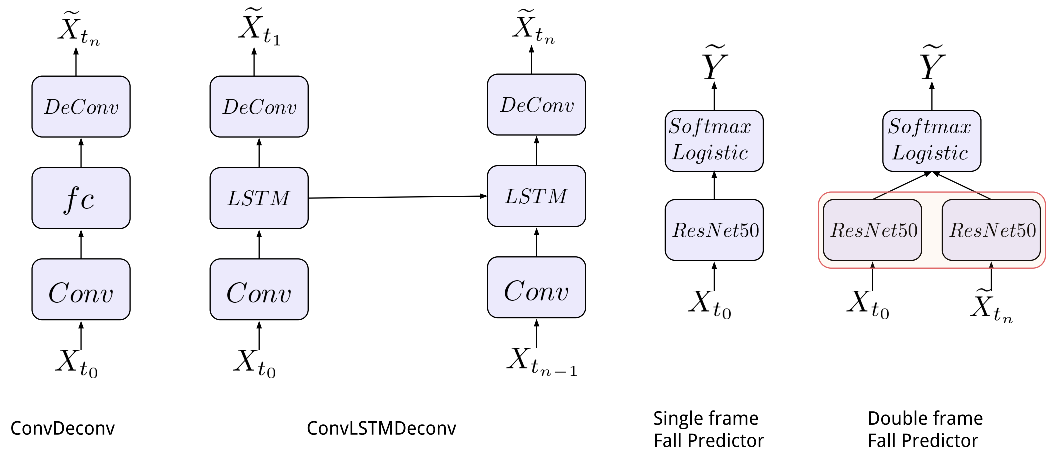

Figure 3: Different model architectures. The first two on the left are ConvDeconv and ConvLSTMDeconv described in Section 4.1. And the two on the right are models used for the supervised fall prediction described in Section 4.2. Single frame predictor is the baseline model. And the double frame predictor is the model that uses the generated data.

## 5 RESULTS

Figure 4 shows the classification results for each of the 9 models described in Section 4.2 tested on 3, 4 and 5 blocks. Each test case is shown with a different color. And Table 3 shows all the 27 test case results' numerical values. In almost all cases the generated data improves the generalization performance to test cases with a different number of blocks than it was trained on. For comparison we have included results from Zhang et al. (2016) in Table 4. Since Zhang et al. (2016) only reports the results when the models are trained on tower of 4 blocks, the corresponding results would be the second block row in Table 3, models 4S, 4CD and 4CLD. Even though the datasets are not the same, but it can be observed that the range of performance of the baseline 4S model is consistent with the range of performance of AlexNet model on Table 4. It can be seen that how the results of the 4CD model are significantly better than both IPE and human performance reported in Zhang et al. (2016), while the baselines have similar performances.

One observation is the fact that the improvements are more significant when it's been tested on scenarios with more bricks than during training. It also improves the reverse case, i.e. fewer bricks than during training, but the improvement is not as significant. It is worth mentioning that testing on a lower number of bricks is a much harder problem as pointed out in Zhang et al. (2016) too. In their case, the prediction performance was almost random when going from 4 blocks to 3 blocks, which is not the case in our experiments[2]. One possible explanation for performance loss is that a balanced tower with fewer blocks corresponds to an unstable configuration for a tower with more blocks e.g. a tower with 3 blocks is classified as unstable for a prediction model trained on towers of 5 blocks. One solution could be to train these models to predict how many blocks have fallen instead of a binary stability label. Because we have access to this data in our dataset, we explored the same experiments using these labels. Unfortunately, we did not observe any significant improvement. The main reason could be that the distribution of the number of fallen blocks is extremely unbalanced. It is hard to collect data with a balanced number of fallen blocks because some configurations are thus very unlikely e.g. a tower of 5 blocks with only two blocks falls (the majority of the time the whole tower collapses).

The another observation is the fact that models that use ConvDeconv generated data performed slightly better than those that use ConvLSTMDeconv. As seen in Figure 2 the samples in the ConvLSTMDeconv case are more noisy and less sharper than those in Figure 1. This could be caused since after the first time step the model outputs from the last time step is used as input for the next time step, the samples degenerates the longer the sequence is.

Data augmentation was crucial to increase the generalization performance of the stability prediction e.g. 5CD model tested on 4 bricks achieved only $50\%$ without data augmentation while reaching $74.5\%$ accuracy with data augmentation. This significant improvement from data augmentation could be partly because our dataset was relatively small.

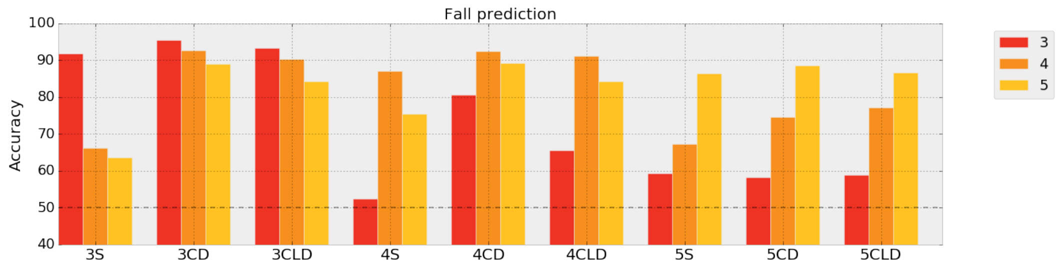

Figure 4: Accuracy in percentage for each of the 9 models tested on test sets with a different number of blocks. Each color represents the number of blocks that the model was tested on. $50\%$ is chance.

[2]We are not using the same dataset as Zhang et al. (2016) and hence direct comparison is not possible.

| Model | Train set | Test set | Accuracy |
|---|---|---|---|
| 3S | 3 | 3 | 91.87 % |
| 3S | 3 | 4 | 66.1 % |
| 3S | 3 | 5 | 63.7 % |
| 3CD | 3 | 3 | 95.5 % |
| 3CD | 3 | 4 | 92.63 % |
| 3CD | 3 | 5 | 89 % |
| 3CLD | 3 | 3 | 93.3 % |
| 3CLD | 3 | 4 | 90.33 % |
| 3CLD | 3 | 5 | 84.30 % |
| 4S | 4 | 3 | 52.5 % |
| 4S | 4 | 4 | 87 % |
| 4S | 4 | 5 | 75.53 % |
| 4CD | 4 | 3 | 80.53 % |
| 4CD | 4 | 4 | 92.5 % |
| 4CD | 4 | 5 | 89.1 % |
| 4CLD | 4 | 3 | 65.53 % |
| 4CLD | 4 | 4 | 91.20 % |
| 4CLD | 4 | 5 | 84.20 % |
| 5S | 5 | 3 | 59.26 % |
| 5S | 5 | 4 | 67.23 % |
| 5S | 5 | 5 | 86.50 % |
| 5CD | 5 | 3 | 58.27 % |
| 5CD | 5 | 4 | 74.50 % |
| 5CD | 5 | 5 | 88.53 % |
| 5CLD | 5 | 3 | 58.90 % |
| 5CLD | 5 | 4 | 74.50 % |
| 5CLD | 5 | 5 | 88.53 % |

Table 3: The results from our experiments

| Model | Train set | Test set | Accuracy |
|---|---|---|---|
| AlexNet | 4 | 3 | 51 % |
| AlexNet | 4 | 4 | 95 % |
| AlexNet | 4 | 5 | 78.5 % |
| IPE | N/A | 3 | 72 % |
| IPE | N/A | 4 | 64 % |
| IPE | N/A | 5 | 56 % |
| Human | N/A | 3 | 76.5 % |
| Human | N/A | 4 | 68.5 % |
| Human | N/A | 5 | 59 % |

Table 4: The results reported on Zhang et al. (2016). We emphasize that these results are on a different dataset.

## 6 Conclusion

In this paper, we showed that data generated from an unsupervised model could help a supervised learner to generalize to unseen scenarios. We argue that this ability of transfer learning and generalization by observing the world could be one of the ingredients to construct a model of the world that could have applications in many tasks, such as model-based RL. We aim to extend this work in future by looking at the videos of robots manipulating objects and being able to predict their failure beforehand, which could help an RL agent to explore more intelligently.

### Acknowledgments

We would like to thank Harm de Vries and Laurent Dinh for their help and feedback in writing the paper. And also thank Adam Lerer and Jiajun Wu for sharing their dataset. We thank NSERC, CIFAR, IBM, Canada Research Chairs, Google and Samsung for funding.

## References

Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. *arXiv preprint arXiv:1610.00696*, 2016.

Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. *arXiv preprint arXiv:1605.07157*, 2016.

Katerina Fragkiadaki, Pulkit Agrawal, Sergey Levine, and Jitendra Malik. Learning visual predictive models of physics for playing billiards. *arXiv preprint arXiv:1511.07404*, 2015.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pp. 249–256, 2010.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *arXiv preprint arXiv:1603.05027*, 2016.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pp. 3581–3589, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *arXiv preprint arXiv:1604.00289*, 2016.

Adam Lerer, Sam Gross, and Rob Fergus. Learning physical intuition of block towers by example. *arXiv preprint arXiv:1603.01312*, 2016.

William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016.

Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.

Tomáš Mikolov. Statistical language models based on neural networks. *Presentation at Google, Mountain View, 2nd April*, 2012.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 807–814, 2010.

Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in Neural Information Processing Systems*, pp. 2863–2871, 2015.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016.

Casper Kaae Sønderby, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár. Amortised map inference for image super-resolution. *arXiv preprint arXiv:1610.04490*, 2016.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using lstms. *CoRR, abs/1502.04681*, 2, 2015.

Ernő Téglás, Edward Vul, Vittorio Girotto, Michel Gonzalez, Joshua B Tenenbaum, and Luca L Bonatti. Pure reasoning in 12-month-old infants as probabilistic inference. *science*, 332(6033): 1054–1059, 2011.

Renqiao Zhang, Jiajun Wu, Chengkai Zhang, William T Freeman, and Joshua B Tenenbaum. A comparative evaluation of approximate probabilistic simulation and deep neural networks as accounts of human physical scene understanding. *arXiv preprint arXiv:1605.01138*, 2016.