Do Robots Dream of Perfectly Elastic Collisions?

Ifunanyachukwu E. Aniemeka, ianiemeka@gmail.com
Sept 6, 2018

Introduction

Human beings possess the ability to make predictions about how objects in the physical world behave without necessarily understanding the laws of Newtonian mechanics. We develop this intuition fairly early on in our development.

We understand that if we throw a ball into the air, it will travel in a roughly parabolic trajectory before hitting the ground. We know that if we roll that same ball across the surface of a table, it will eventually come to a stop. Moreover, we understand if we throw it towards a wall, it will bounce back. We know all of these things without necessarily knowing the equation for the gravitational force between two bodies, or the law of conservation of momentum, etc.

In fact, we may be able to predict the motion of objects *despite* an incorrect understanding of physics. If you were to ask a random person whether a bowling ball or a feather falls to the earth faster in a vacuum, many people would say the bowling ball does. However, we know for a fact that objects under the influence of gravity alone, as two objects in a vacuum would be, fall towards the earth at the same acceleration - 9.8 m/s/s.

Moreover, we are able to extrapolate from previous experience the behavior of unfamiliar objects. We don't need to see every variety of object thrown into the air to be able to give a rough estimate of its trajectory.

The motive behind my project was to discover whether a similar sort of intuition could be cultivated in a neural network via the same method humans use to develop physical intuition - observation of and interaction with the physical world. The proposed network would be able to make physical predictions without definitions of concepts like momentum, force, or friction being built-in. Furthermore, I'm interested in how few and what sorts of examples I need to provide to a network for it to understand not only the behavior of the objects present in the samples, but also for it to extrapolate to similar situations.

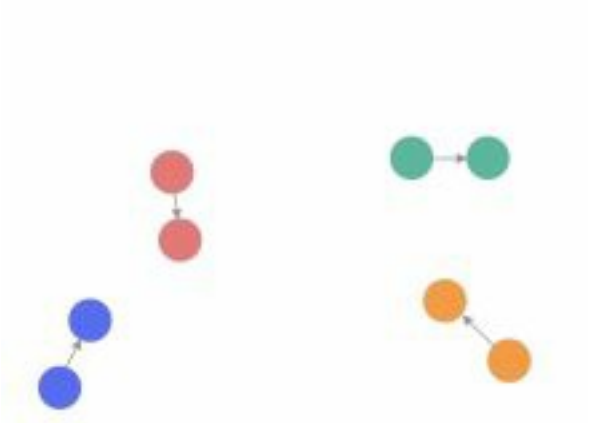
To train my network, I asked it to perform a task that most human beings would find fairly simple. Given two frames of a video in which some number of objects are moving, predict the positions of those objects in the following frame.

Let's say we have a recording of circles moving within a closed space and I showed you two sequential frames:

frame 1 at time t=0

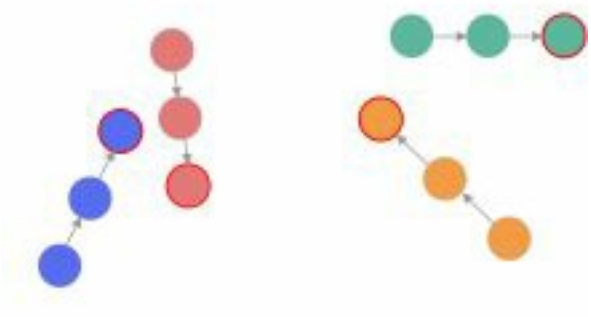
frame 2 at time t=1

You're likely able to roughly intuit the vectors describing the motion of the circles.



difference between frames 1 and 2

Having intuited the circles' motion vectors, you would then be able to guess at the position of the circles in the next frame.



frame 3 circles outlined in red

Related Work

There have been previous efforts to train neural networks with physical intuition. In their paper 'Learning to Poke by Poking: Experiential Learning of Intuitive Physics', Agrawal et al. describe training a Baxter robot to interact with its environment so as to produce a particular result. The robot is directed to poke at one to three of sixteen objects placed on a table in front of it. Objects can be pushed in any direction and the robot can poke at any part of an object. The latter allowance means that sometimes, when an object is poked, it rotates instead of or in addition to changing position.

The network implements forward and inverse models of the objects' movements. The forward model, is trained to predict the state the object will be in given the current state and an action, i.e. a poke. The inverse model predicts the appropriate poke given the current and target state.

The authors make use of a Siamese convolutional network, i.e. a pair of connected convolutional networks. A training sample consists of the image I_t of an object at time t, an image I_{t+1} of the same object at the next time step, and the poke (which is defined by the poke point on the object, poke direction, and poke length) that resulted in the change in the object's state. One of the subnetworks receives I_t and the other receives I_{t+1}.

To train the inverse model, the results of the subnetworks, x_t and x_{t+1}, are concatenated and passed as input to fully connected layers that predict the poke point, length, and angle of the poke required to get from the state at t to the state at $t+1$. This prediction can then be compared to the poke in the sample. The forward model is trained by passing the poke element of the sample and x_t through fully connected layers. The result of this part of the network can then be compared to x_{t+1}.

Their method represents the state of objects using tuples of data or latent feature

representations. Images are present in the sample, but afterwards, only the feature representations are used to perform analysis. In contrast, my own network performs a per-pixel analysis in which the result of the network, an image, is compared to the target image.

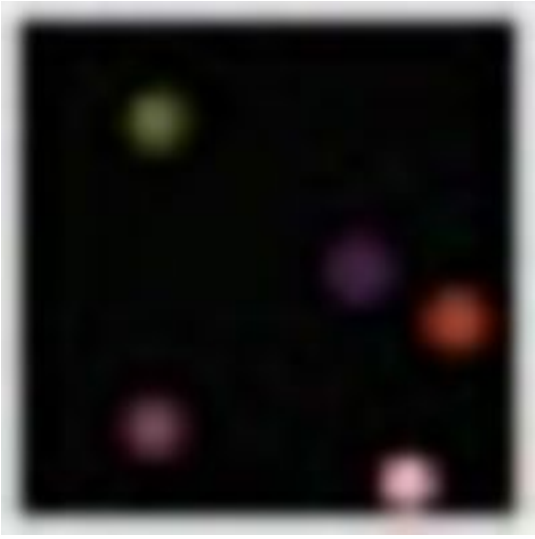
In 'FlowNet: Learning Optical Flow with Convolutional Networks', the authors train a network to produce optical flow fields from paired images. They developed two architectures - FlowNetSimple and FlowNetCorr, of which I will only discuss the former as it is most closely related to my own work. Two frames are stacked together and passed through first several convolution layers, then a series of deconvolution layers. The output of the network is compared against the known optical flow field for the two images.

My own work does something quite similar. The network I've built is, instead, asked to produce the next frame, given two adjacent frames.

Method

Data

The data was generated using the matter.js physics engine. The environment constructed is a black square bounded by four gray walls. Within the square, anywhere between one to five circles are set in motion. The circles can collide with one another as well as the walls. Each circle is initialized with a random color, initial position, and velocity.



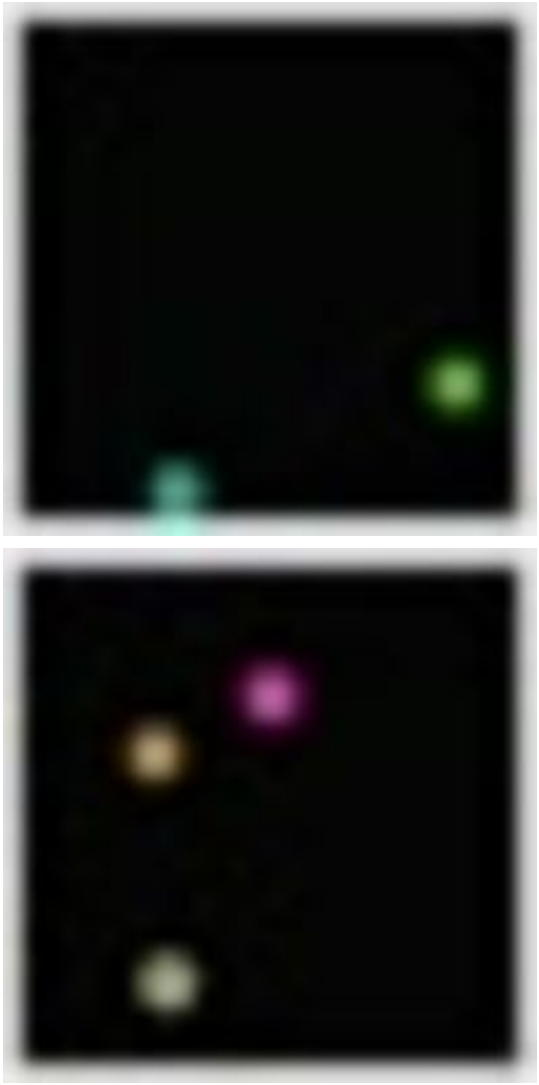


image samples

Several factors are removed which would be present in a realistic simulation of the physical world. The environment has no air or surface friction. In addition, all collisions are perfectly elastic and none of the circles rotate.

A total of 4,235 training samples were generated from 121 recordings of the circles. From each recording, 105 frames were extracted at a rate of 21 fps. Each frame is 28 x 28 pixels.

Model

The network accepts as input a pair of frames stacked such that the dimension of the result is 28 x 28 x 6. The second image is one timestep removed from the first. The target value is the frame that follows the second.

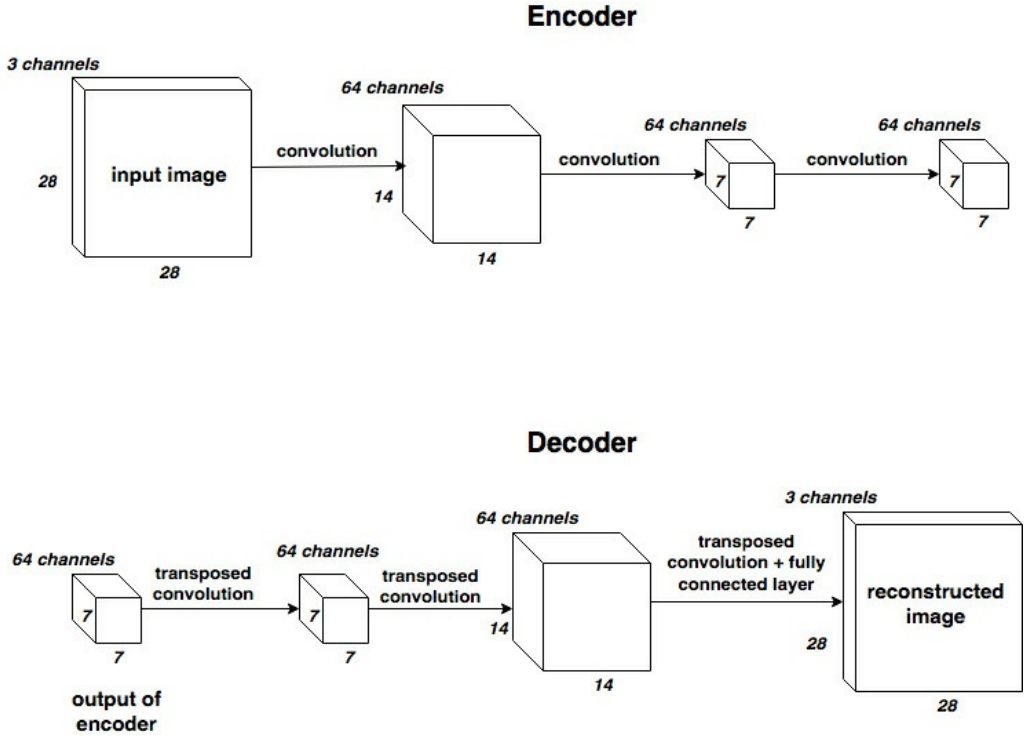
frame 1 / frame 2 / target

frame 1 / frame 2 / target

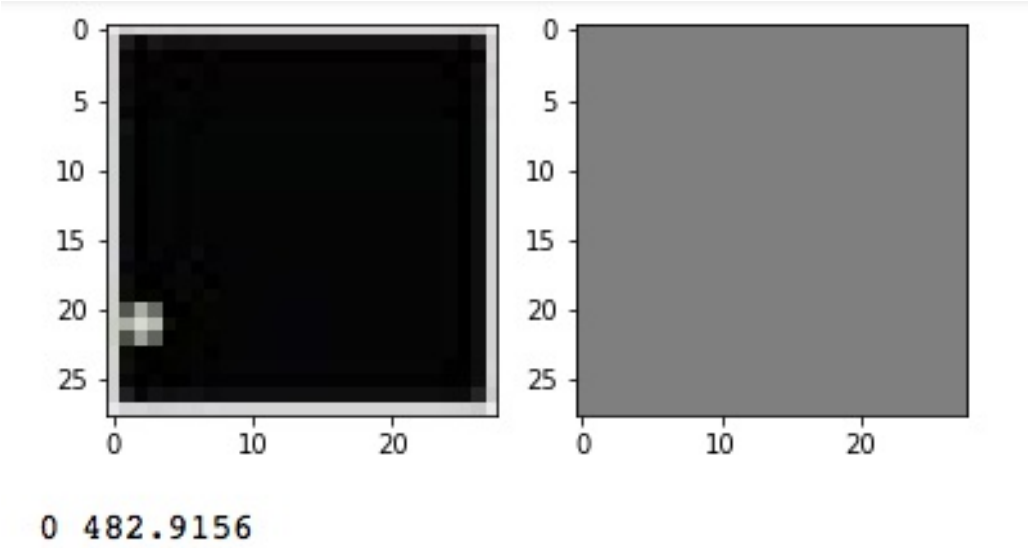
frame 1 / frame 2 / target

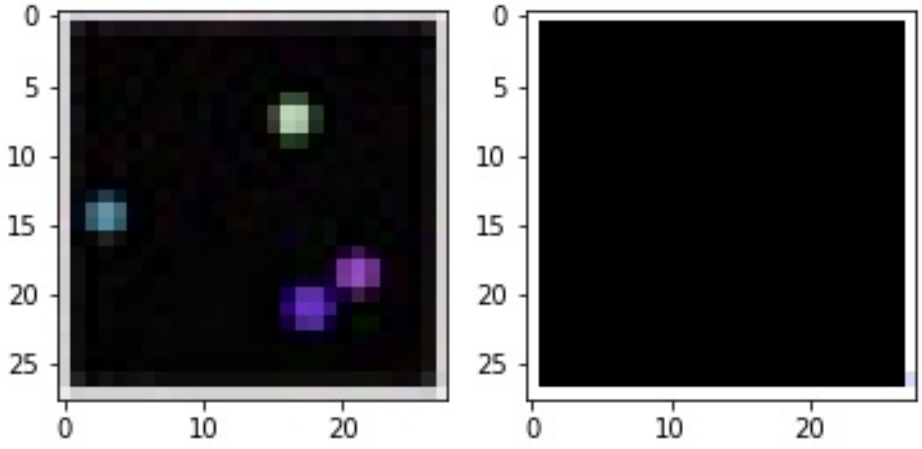
The model is a modification of the standard autoencoder architecture. It's based on the

variational autoencoder constructed in Felix Mohr's article on VAEs. The encoder consists of several convolution layers. We specify sixty-four 4 x kernels at each convolution layer. For the purpose of preventing overfitting, following each of the convolution layers are dropout layers. Including the dropout resulted in a 7.32% decrease in the mean squared error.

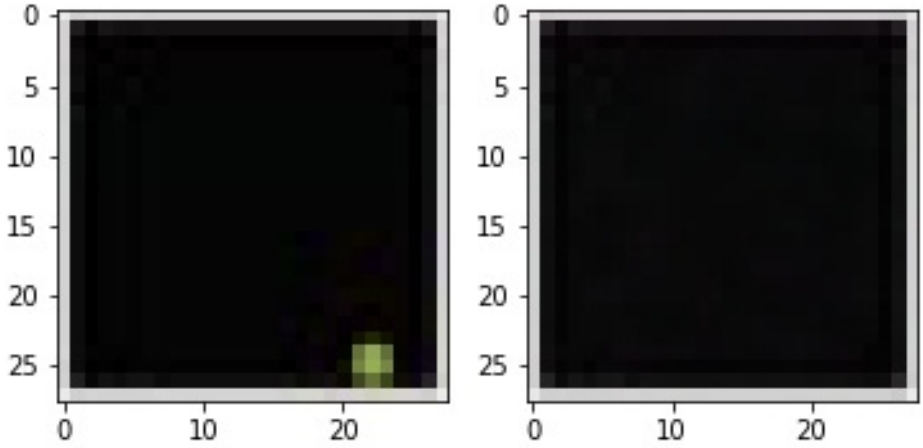


The number on the left is the epoch number. The number on the right is the mean squared error.

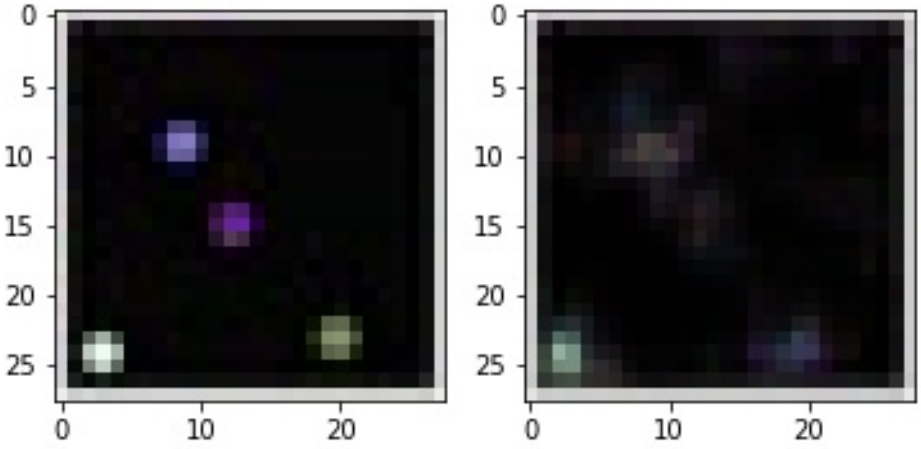




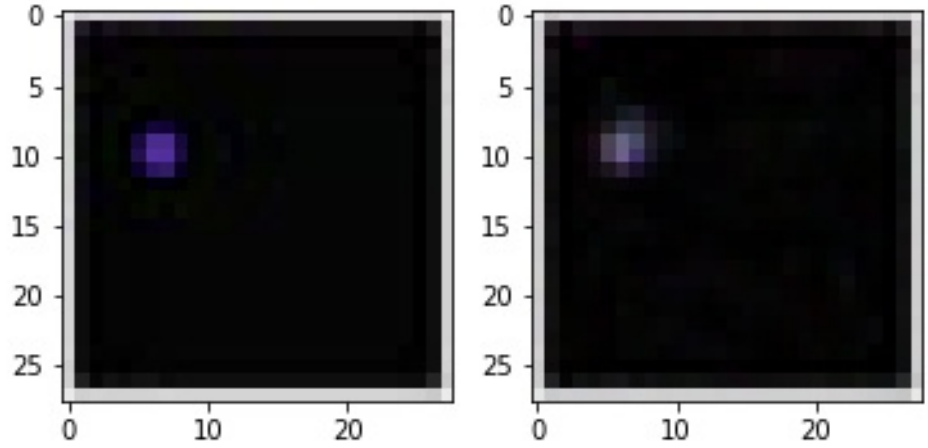
2000 22.329369



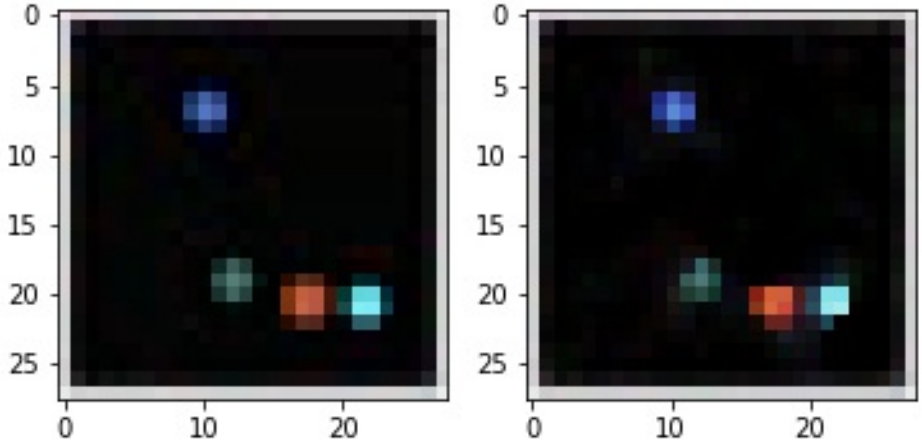
4000 13.42361



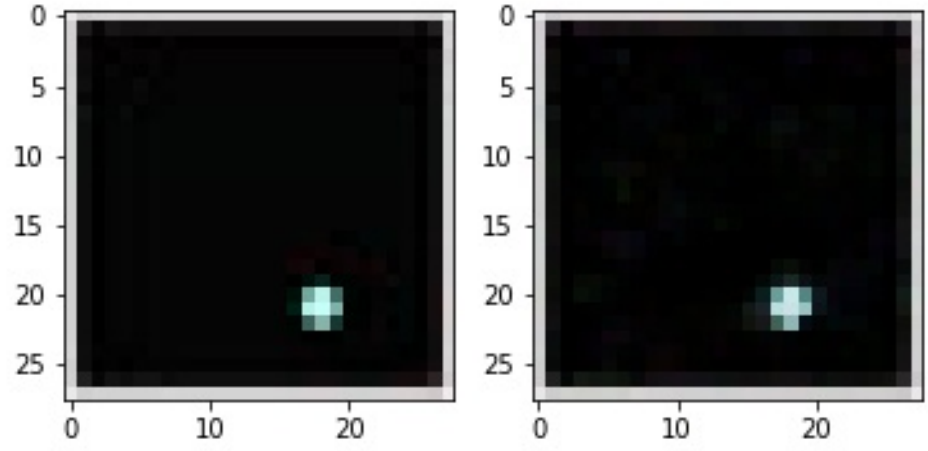
6000 7.2320013



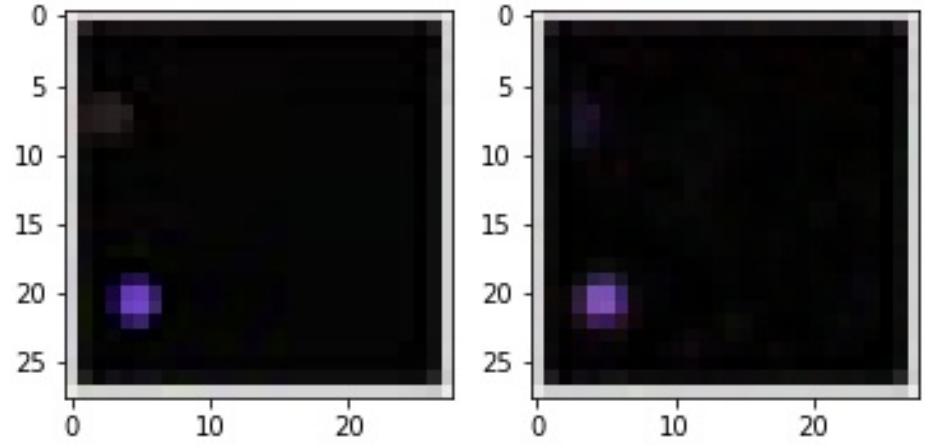
8000 3.9249089



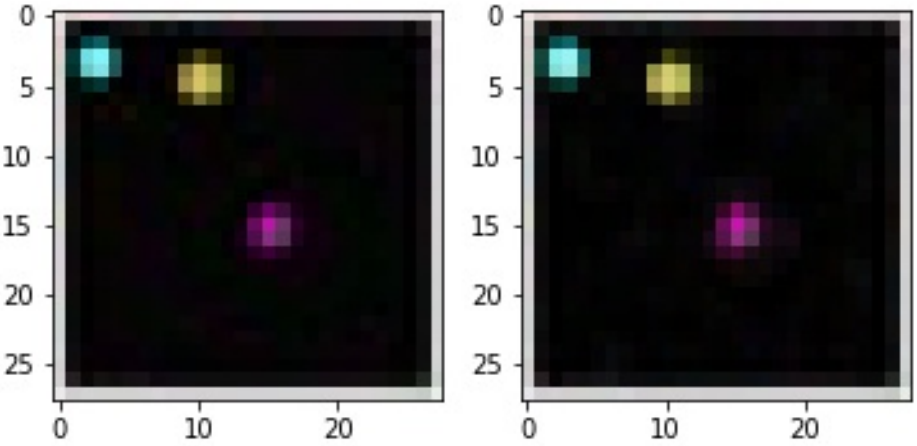
10000 1.9533062



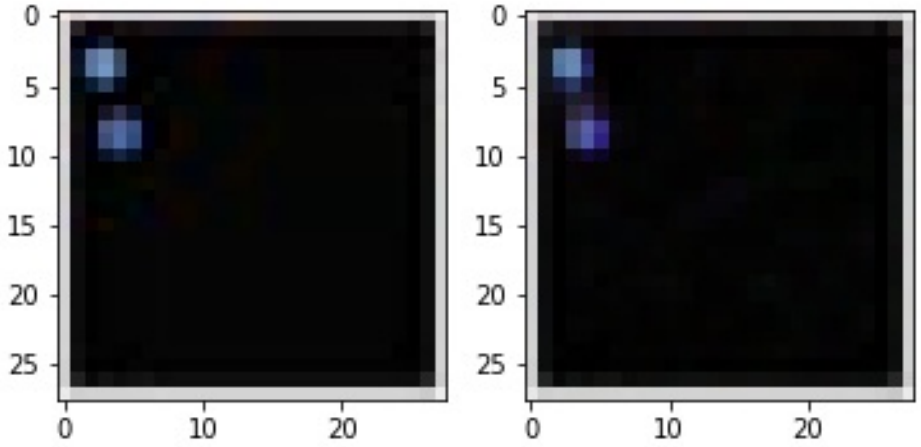
12000 1.0533991



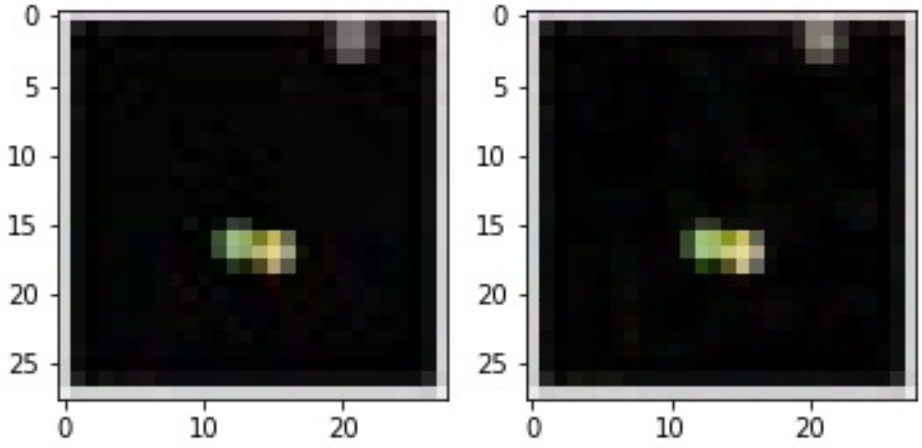
14000 0.8319657



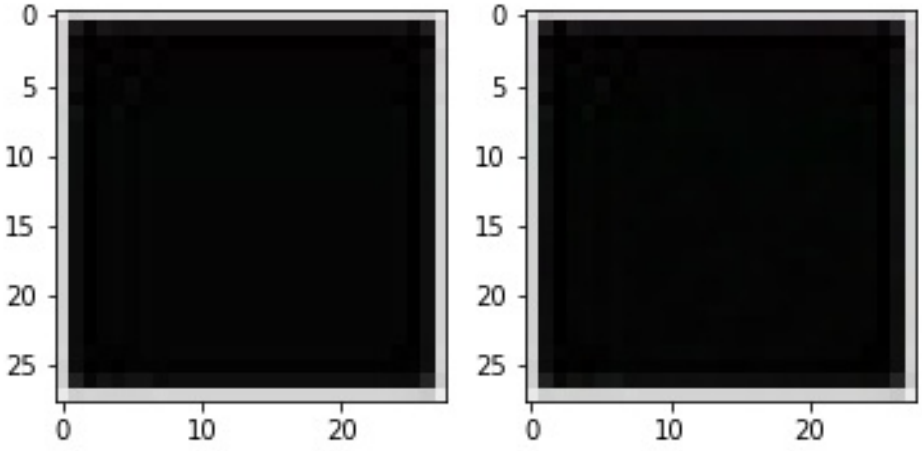
16000 0.73906547



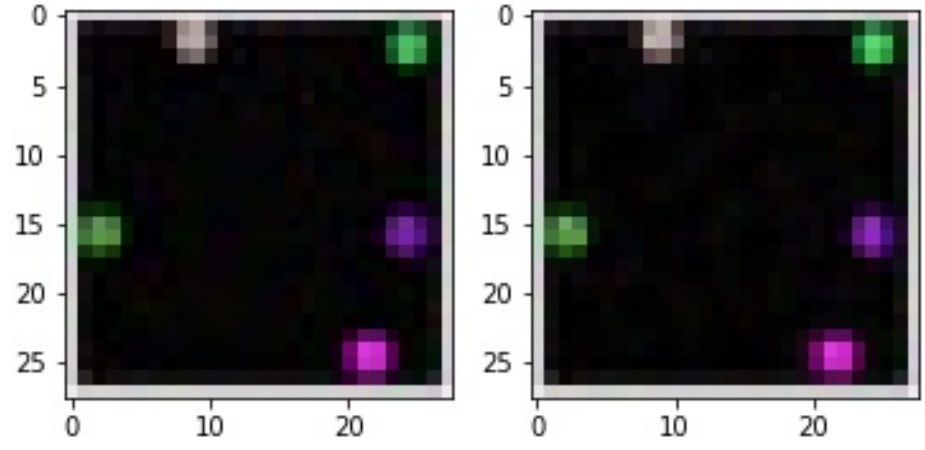
18000 0.56951064



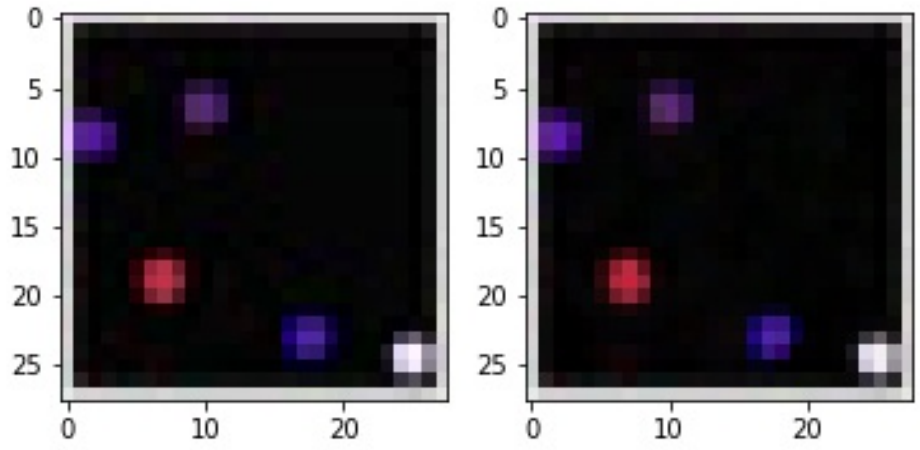
20000 0.5321298



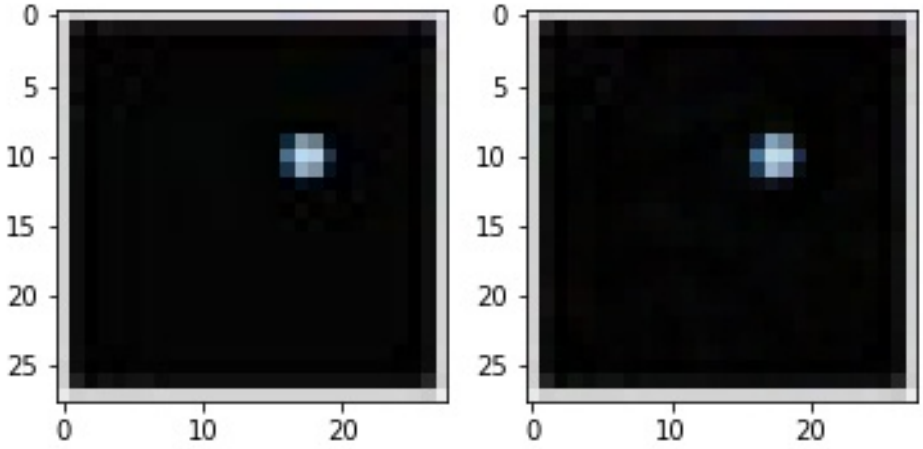
22000 0.40588522



24000 0.38744116



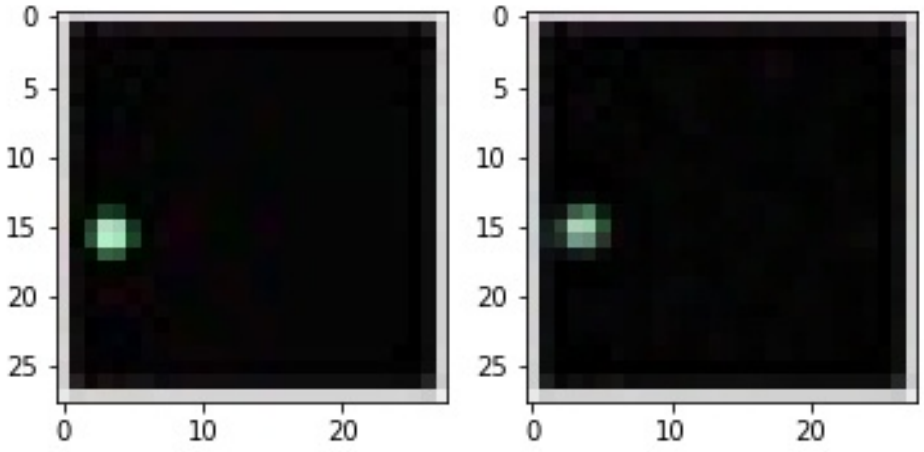
```
26000 0.37468433
```



```
28000 0.31552958
```

The results using the test set:

```
Evaluation
```




```
7.664101
```

In a standard convolutional autoencoder, the goal is for the network to reconstruct the input image. The convolution layers in the standard case discern the features that are most important for accurately reconstructing the input.

In contrast, the network above is trained to output the frame following the input, which is the two proceeding frames stacked along the z-axis. The decoder specifies transposed convolutional layers (also sometimes referred to as deconvolutional layers) that perform the reconstruction.

After 30,000 epochs, the network achieved an MSE of about 0.325. However, when given samples from the test set, the MSE increases to 7.664. This indicates that there is some overfitting in the network. Future iterations of the network may have to use higher dropout rates.

It appears that the network finds it simpler to predict the position of objects than their color. This makes some intuitive sense. Position can be described with two numbers. The coloring of each of the circles, however, requires a 3-dimensional matrix of numbers between 0 and 255.

Conclusion

The model demonstrates that it is possible for a network to understand rules around how objects move in a simple physical environment. However, there are many paths left to be explored.

For one, during data generation, the rate of frame extraction was set to 21. How accurate would a network trained at a higher or lower frame extraction rate be? It would be illuminating to compare the results of networks trained at various frame rates.

In addition, the network was trained in a square environment. There are plenty of other shapes I could have chosen. It's not yet known how the network would respond if given two frames in which circles are moving within a triangular space. I can't say for sure if the network is able to generalize that the circles should bounce off of all walls, regardless of how those walls are oriented. Similarly, it's unknown how the network would react to a barrier placed within the square.

A number of elements were removed from the environment that would normally be present. For one, there is no air friction. In a more realistic environment, the circles would slow down over time, eventually coming to a stand still. Another area to explore would be training a network to understand the motion of circles that undergo inelastic collisions.

One interesting variation on the network described in this paper would be a network that, given two frames, could predict the position of the circles two or more frames ahead. At each step, the network would use its previous prediction and the prediction before that to guess at the next positions of the circles.

All of these explorations still involve artificial environments. Ultimately, the goal is to have a network that can make predictions about objects in the real world. The real world is much messier, of course. Surfaces are bumpy. Air friction must be accounted for. Objects are made of all kinds of materials. It may not be possible, in many situations, to make a very accurate prediction of where an object will be after some number of time steps. Having said that, human beings can make rough estimates of real objects' motion. It would be interesting to know if a neural network could make predictions better than those of humans.

References

[1] P. Agrawal, A. Nair, P. Abbeel, J. Malik, and S. Levine. Learning to Poke by Poking: Experiential Learning of Intuitive Physics. arXiv:1606.07419v2, 2017.

[2] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, T. Brox. FlowNet: Learning Optical Flow with Convolutional Networks. arXiv:1504.06852v2, 2015.

[3] F. Mohr. Teaching a Variational Autoencoder (VAE) to Draw MNIST Characters. https://towardsdatascience.com/teaching-a-variational-autoencoder-vae-to-draw-mnist-characters-978675c95776.