Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Contextual RNN-GANs for Abstract Reasoning Diagram Generation

Viveka Kulharia,*[1] Arnab Ghosh,*[1] Amitabha Mukerjee,[1]
Vinay Namboodiri,[1] Mohit Bansal[2]
[1]IIT Kanpur [2]UNC Chapel Hill
{vivekakulharia,arnabghosh93}@gmail.com,{amit,vinaypn}@iitk.ac.in,
mbansal@cs.unc.edu

Abstract

Understanding, predicting, and generating object motions and transformations is a core problem in artificial intelligence. Modeling sequences of evolving images may provide better representations and models of motion and may ultimately be used for forecasting, simulation, or video generation. Diagrammatic Abstract Reasoning is an avenue in which diagrams evolve in complex patterns and one needs to infer the underlying pattern sequence and generate the next image in the sequence. For this, we develop a novel Contextual Generative Adversarial Network based on Recurrent Neural Networks (Context-RNN-GANs), where both the generator and the discriminator modules are based on contextual history (modeled as RNNs) and the adversarial discriminator guides the generator to produce realistic images for the particular time step in the image sequence. We evaluate the Context-RNN-GAN model (and its variants) on a novel dataset of Diagrammatic Abstract Reasoning, where it performs competitively with 10th-grade human performance but there is still scope for interesting improvements as compared to college-grade human performance. We also evaluate our model on a standard video next-frame prediction task, achieving improved performance over comparable state-of-the-art.

Introduction

The recent success of machine learning and neural networks in Atari and Go (Mnih et al., 2015; Silver et al., 2016) has sparked a renewed interest in artificial intelligence models that can perform well at tasks which even humans find challenging. An important task in this category is abstract reasoning, which measures one's lateral thinking skills or fluid intelligence, i.e., the ability to quickly identify patterns, logical rules and trends in data, integrate this information, and apply it to solve new problems. Specifically, we address the problem of diagrammatic abstract reasoning (DAR), a subset of Differential Aptitude Tests (DATs), which were introduced by (Bennett et al., 1947) to judge psychometric proficiency. It has featured in Intelligence Assessment Systems since 1950s and has been validated by (Berdie, 1951), who showed that proficiency in DAT-DARs were predictors in engineering training. A DAR task involves the generation of a future diagram based on the sequential evolution of component patterns in the given problem sequence.

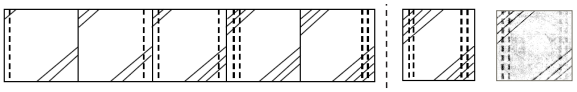

Figure 1: Example abstract reasoning problem, where our model was able to generate an image very close to the correct answer.

Fig. 1 shows an example problem from our DAT-DAR dataset and highlights the intricacies of the reasoning involved in inferring the correct answer (i.e., the next image in the sequence). Different pattern components on both the sides and both the corners are changing in different and multiple ways, making it an interesting challenge to correctly generate the next image in the sequence.[1] Accurate generation models developed for such a reasoning task can be used for general AI applications such as forecasting and simulation generation. These models will also be useful for generation of real-world images and videos, a recent research direction in computer vision and deep learning (Goodfellow et al., 2014; Radford et al., 2015; Denton et al., 2015; Im et al., 2016; Mathieu et al., 2015). In this direction, we present an application of our best models to the task of next frame generation in Moving MNIST videos.

Our sequential generation model is a temporally recurrent version of Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), which we name Context-RNN-GANs[2], where context refers to the sequential history (modeled as RNNs). This is especially well-suited to our sequential image-based reasoning tasks. In this model, the generator

*Equal Contribution

[1]An explanation of the ground truth is that the dashed line first goes to the left, then to the right, and then on both sides, and also changes from single to double, hence the ground truth should have double dashed lines on both the sides. On the corners, the number of slanted lines increase by one after every two images, hence the ground truth should have four slant lines on both the corners.

[2]In Context-RNN-GAN, 'context' refers to the adversary receiving previous images (modeled as an RNN) and the generator is also an RNN. The name distinguishes it from our simpler RNN-GAN model where the adversary is not contextual (as it only uses a single image) and only the generator is an RNN.

is an RNN which tries to generate the correct next image based on the previous sequence of images. The adversarial discriminator (playing a minimax game with the generator) is also an RNN which gets the previous timesteps' images as context and the current timestep's image (either the one generated by the generator as a negative sample or the correct image from the dataset for that timestep as a positive sample). In this way, the discriminator uses the images preceding the current timestep as contextual information to better distinguish whether the generator produced realistic images for the particular timestep in the sequence (as opposed to just producing a realistic image from the data distribution).

We also develop novel image representations using an unsupervised Siamese Network (Chopra et al., 2005) for modeling the joint representation of adjacent timesteps. This helps bring in much more information (as input features to the GAN model) about the temporal evolution across consecutive time steps. Modeling of the problem using an unsupervised setting and training as a sequential image language model on large quantities of sequences help the method to generalize better on unseen test sequences.

Empirically, we perform quantitative evaluation on the DAT-DAR dataset by first generating the successor image for each test sequence and then measuring similarity between the generated image and the candidate answer images, in a multiple-choice setting of the Intelligence Quotient (IQ) dataset. We then report the accuracy as the percentage of correct hits. We compare several baselines, model variants, and feature representations and find that our Context-RNN-GAN model with Siamese CNN features performs the best. Also, compared to human performance, we promisingly find that our final model is competitive with 10th-grade high school students but there is still scope for interesting improvements as compared to advanced engineering college students. We also demonstrate that our Context-RNN-GAN model can successfully model video next-frame generation via the Moving MNIST dataset, where we achieve improved performance over comparable state-of-the-art.

In summary, the main contributions of this paper are:

- A new abstract reasoning and image generation dataset with more than 1500 training problems (sequences of five images each, plus eight transformation types, leading to a total collection of more than 60000 training images); and an annotated evaluation test set of 100 problems.
- A novel temporally contextual, RNN-based adversarial generation model, where the adversary has access to the full context of previous preceding images as context for deciding real vs fake sample for that timestep.
- A novel feature representation of temporally adjacent images using a Siamese Network.
- Strong performances on two datasets (DAR and MNIST videos), competitive with 10th-grade humans and comparable state-of-the-art.

Related Work

(Stern, 1914) introduced IQ Tests to measure the success of an individual at adapting to a specific situation under a specific condition. Visual problems in intelligence tests have been among the earliest and continuously researched problems in AI. It has been looked at in terms of propositional logic beginning with (Evans, 1964) and more recently by (Prade and Richard, 2011). Recently, there has also been significant interest in building systems that compete with humans on a variety of tasks such as geometry-based problems (Seo et al., 2015), physics-based problems (Novak and Bulko, 1992), repetition and symmetry detection (Novak and Bulko, 1992), visual question answering (Antol et al., 2015), and verbal reasoning and analogy (Mikolov et al., 2013; Wang et al., 2015).

Our task closely relates to the problem of Raven's Progressive Matrices. There, the problems are more constrained; one of the squares of the matrix is missing and the sequential pattern evolves along the columns and rows. This has been addressed using a propositional logic based framework (Falkenhainer et al., 1989) and via a theorem prover based approach (Bringsjord and Schimanski, 2004). However, we focus on a novel task which involves more unrestricted pattern movements, bigger datasets, and does not rely on representability in terms of propositional logic.

Our task is also closely related to the task of next frame prediction in videos (Petrovic et al., 2006) which involves predicting the next frame based on previous frames. However, the change across consecutive real-world video frames is extremely small as compared to the evolving shapes and changing spatial dynamics in our diagrammatic reasoning task. Hence, this poses several challenges to us different from the task of video next-frame generation, in which modeling optical flows plays a major role in producing better-looking next frames. Other related work in the video prediction direction include language modeling based approaches (Ranzato et al., 2014), convolution-based LSTMs (Patraucean et al., 2015), adversarial CNNs (Mathieu et al., 2015), context encoders (Pathak et al., 2016) and data-conditioned GANs (Mirza and Osindero, 2014).

Lastly, generative adversarial networks (GANs) (Goodfellow et al., 2014) have also been extended with *spatial* (and spatio-temporal) recurrence, attention, and structure (Im et al., 2016; Gregor et al., 2015; Wang and Gupta, 2016; Vondrick et al., 2016), whereas we specifically focus on *temporal* recurrence constraints for frames of a video via RNNs. GANs have also been used for high resolution image generation (Radford et al., 2015), image manipulation (Zhu et al., 2016), and text-to-image synthesis (Reed et al., 2016).

(Vondrick et al., 2016) generates videos from a single image using GANs but the discriminator judges the entire video rather than individual frames conditioned on previous frames. (Patraucean et al., 2015) use Convolutional-LSTMs, similar to our GRU-RNN (with Shallow-CNN features) baseline but our final Context-RNN-GAN model with an adversarial loss gives better results. (Mathieu et al., 2015) predict multiscale videos using CNNs but only provide fixed number of previous frames as context to the discriminator which is not helpful for modeling short sequences. (Oh et al., 2015) can generate long-term future frames in action dependent games but the transition of frames is mostly smooth with very similar consecutive frames, unlike our DAR task which involves discontinuous movements in an

evolving diagrammatic pattern.

Models

We first describe our primary Context-RNN-GAN model and then briefly discuss two simplifications of that model, namely RNN-GAN and a regular RNN.

The major motivation for using an adversarial loss was the shortcomings of L-2 and L-1 losses (Fig. 3):

- When using an L-2 loss function, some of the generated images were superimpositions of the component parts and were too cluttered.
- When using an L-1 loss function, although it was sharper than using an L-2 loss, it was missing some components of the actual diagrams.

Context-RNN-GAN

Our Context-RNN-GAN model (Fig. 2) uses the sequential structure of the diagrammatic abstract reasoning problem in a GAN framework, i.e., to generate the next image after a sequence of previous context images. The basic principle underlying our model is the same as that of the original GAN model by (Goodfellow et al., 2014), which is that the discriminator and generator play the following minimax game:

$$min_\beta max_\theta F(\beta,\theta) = \mathbb{E}_{x\sim p_{data}}[logp_\theta(y=1|x)] + \mathbb{E}_{x\sim p_\beta}[logp_\theta(y=0|x)] \quad (1)$$

where p_{data} is data's distribution, p_β is generator's distribution with β as the vector of parameters for the generator, and p_θ is discriminator's distribution with θ as the vector of parameters for the discriminator. The discriminator tries to distinguish between inputs x sampled from the real data and from the generator's distribution by labeling them as 1 and 0 respectively. On the other hand, the generator tries to fool the discriminator by getting its generated image also labeled as 1 by the discriminator. A GAN model is trained well when the discriminator cannot discriminate between the images generated by the generator and the images from the actual data distribution from which it is sampled. Next, in our case, we importantly also add sequential context to both the generator and the discriminator of the GAN model (via RNNs) to capture the temporal sequence-of-images nature of our task, as described in detail next.

Generator We want the generator to generate an output image y_t given the previous images (in the question) in sequential order x_1 to x_t, such that this generated image y_t is as close as possible to the correct next timestep's image in the sequence x_{t+1}. Therefore, we choose the generator to be a sequential RNN model, which generates the sequentially next image, $y_t = G(x_1 \dots x_t)$, trying to fool the discriminator into believing that these actually follow their preceding input, x_1 to x_t (see Fig. 2). Note that our generator model can use LSTM-based or GRU-based or vanilla RNNs; we choose GRUs based on empirical evaluation.

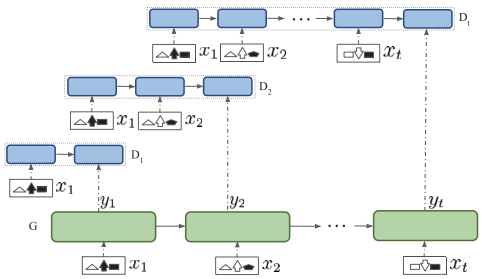

Figure 2: Context-RNN-GAN model, where the generator G and the discriminator D (where D_i represents its ith timestep snapshot) are both RNNs. G generates an image at every timestep while D receives all the preceding images as context to decide whether the current image output by G is real vs generated *for that particular timestep*. x_i are the input images.

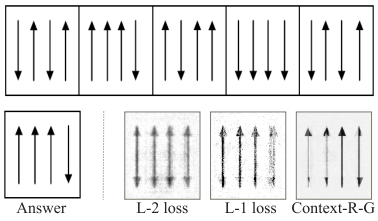

Figure 3: Comparison of image generation quality for L-1 and L-2 loss functions (in GRU-RNN) vs our Context-RNN-GAN model.

Discriminator We want the discriminator to be able to decide whether a given image actually follows the previous sequence of images $x_1 \dots x_t$ (and not just whether the given image is from the real data distribution, unlike a traditional GAN model's discriminator). For this, we inject context into the discriminator too, by including the preceding (context) images, $x_1 \dots x_t$, along with generator's generated image, y_t. It provides context for the discriminator to decide whether the generator's image actually follows the previous (context) images. To model this context, we choose the discriminator to be a sequence model as well, namely a GRU-RNN. This sequence model is essentially a sequence-to-label encoding model which receives the context images as the preceding timesteps and the generator's image (or actual image) as the last timestep (see Fig. 2). The final hidden state is mapped on to a sigmoid predicting whether it is an actual or fake image for that timestep.

Training the Discriminator (D) For a particular timestep (t+1), x_{t+1} is the correct image for the timestep and $y_t = G(x_1 \dots x_t)$ is the generated image output by the generator G. We train the discriminator D such that $(x_1 \dots x_t, x_{t+1})$ is classified as 1 (real) and $(x_1 \dots x_t, y_t)$ as 0 (fake). Therefore the loss function we use to train D is:

$$L^D_{adv} = \sum_{t=1}^{timesteps-1} L_{bce}(D(x_1 \dots x_t, x_{t+1}), 1) + L_{bce}(D(x_1 \dots x_t, y_t), 0) \quad (2)$$

where L_{bce} is the binary cross entropy loss:

$$L_{bce}(Y, \tilde{Y}) = -\tilde{Y}log(Y) + (1-\tilde{Y})log(1-Y) \\ \tilde{Y} \in \{0,1\}, Y \in [0,1] \tag{3}$$

Training the Generator (G) For a single sequence of images $x_1 \dots x_{timesteps}$, keeping the weights of the discriminator D fixed, we minimize the adversarial loss:

$$L^G_{adv} = \sum_{t=1}^{timesteps-1} L_{bce}(D(x_1 \dots x_t, G(x_1 \dots x_t)), 1) \tag{4}$$

Minimizing the above loss implies that the G tries to adjust its weights so that the generated image $y_t = G(x_1 \dots x_t)$ is as close to a "real" image following $x_1 \dots x_t$ as judged by the current state of the discriminator. Just minimizing the above loss can lead to instability in training because the generator can generate images y_t which are able to fool the discriminator but the manifold might be quite different to x_{t+1} which it wants to model correctly. Hence, similar to (Mathieu et al., 2015) and (Pathak et al., 2016), we train the model with a combination of L^G_{adv} loss (adversarial loss) and L_p loss, which is defined as:

$$L_p = \sum_{t=1}^{timesteps-1} \|x_{t+1} - y_t\|_p^p \tag{5}$$

The total loss to be minimized for the generator is then $L^G = \lambda_{adv} L^G_{adv} + \lambda_p L_p$

RNN-GAN

RNN-GAN model is a simplified version of the Context-RNN-GAN model, where the discriminator is simply a Multilayer Perceptron (i.e. a Fully Connected Network) that only gets the generated (and real) images at the current timestep and no previous context image. The objective of this discriminator is to classify whether the image provided to it is an image from the dataset's distribution or is it generated by the generator. i.e. the discriminator is replaced from being $D(x_1..x_t, y_t)$ to being just $D(y_t)$.

RNN

Finally, the simplest model is a regular RNN (again with GRU units, similar to the generator modules above) which just models the sequence of images by trying to predict features of each image given the features of all the previous images in the sequence. We have tried it with L_1 and L_2 loss functions.

Feed-forward Baseline

The feed-forward network baseline is simply a fully-connected multi-layered perceptron with 2 layers, and is trained using the first 4 images $x_1..x_{t-1}$ to predict the fifth image x_t; and during testing, the last 4 images $x_2..x_t$ are used to predict the features of the answer image x_{t+1}.

Feature Representations

In this section, we discuss the various image embedding methods that we used to create input features for the sequence models such as Context-RNN-GAN discussed above.

Raw Pixels As the first baseline, each of the images was resized to the same dimension of 128×128 and its raw pixel values were used as features, with row-wise stacking of the pixel values. Features generated by each model were resized to visualize the generation.

Histogram of Oriented Gradients HOG features (Dalal and Triggs, 2005) are obtained by concatenating histograms of occurrences of gradient orientation in each of the cells of the images, hence capturing features corresponding to various edge types and proving to be a good feature detector.

Autoencoder (Vincent et al., 2008) have shown the effectiveness of autoencoders in reducing the dimensionality of the data. Hence, we used an unsupervised autoencoder with 1000 hidden units in the bottleneck layer and trained it on the images from the dataset. The activations of the bottleneck layer were then used as feature representations of the images.

Pretrained CNN (OverFeat Network) We used 4096 features from the penultimate layers of an OverFeat network (Sermanet et al., 2013) that won in the image localization task in ILSVRC 2013. Having been pre-trained on ImageNet dataset, it has been shown to be useful for tasks such as sketch recognition (Yu et al., 2015). The architecture consists of 7 stages, with 22 layers. Each stage consists of convolutions, rectified linear units (ReLU), and optionally of max-pooling layers. We extracted the output from the 21st layer that is the output of the fully connected layer in the seventh stage before final classification.

Fine-tuned AlexNet model We fine-tuned an AlexNet pretrained on the Imagenet Challenge 2012 (Krizhevsky et al., 2012). Labels were annotated for our DAT-DAR dataset's images by dividing each image into four quadrants and each of the four feature types (horizontal/vertical lines, slanted lines, and curved lines and the number of shaded regions) are counted in each quadrant of the image to get 16 labels for multioutput-multiclass classification. Counting of the features is done using the diagram parser used in (Seo et al., 2015). The model used Euclidean loss and produced 16 outputs.

Shallow CNN We trained an end-to-end shallow CNN with only three convolutional layers (separated by ReLU and Pooling layers) followed by a ReLU, dropout and a fully connected layer. We trained it solely on our images' labels described above. The resultant features of the penultimate layer were used. The learning rate and the kernel size was similar to that used by (Krizhevsky et al., 2012).

Siamese CNN For the initial timesteps, regular sequential models such as LSTMs face difficulty in generating the next image in the sequence because of lack of sufficient context (because initial timesteps have lesser or no previous context). To resolve this, we propose to learn a joint embedding of (temporally) adjacent images and use that as features for our sequential RNN models.

For this, we created a Siamese network (Chopra et al., 2005), which uses the pair of shallow CNNs followed by a fully connected layer on top of each of the CNNs. The CNNs have parameters shared between them and they help find the distance between two images in a feature plane. The Siamese network tries to minimize the distance between similar images and maximize the distance between dissimilar images using contrastive loss (Chopra et al., 2005). Our Siamese network was trained by initializing the two shared CNNs with the parameters of the shallow CNN. It was then fine-tuned on our image dataset via a contrastive loss that uses every pair of two temporally adjacent images in a problem as similar examples and uses all other pairs of images from different problems as dissimilar examples. The activations of the two fully connected layers of the network are fed as features to our GAN models.

Experimental Setup

Dataset We collected the data from several IQ test books and online resources (Aggarwal, 2016), (Sijwali and Sijwali, 2016), (Gupta, 2016) and (Jha and Siddiquee, 2011). We collected about 1500 training problems (15000 images) and an annotated test set of 100 problems, and then we used transforms such as rotation and mirror reflections across the axes to increase the data by eight times, leading to a total of 12000 problem sequences to train on, each containing a sequence of 5 diagrams. Each test problem consists of five figures for the input sequence question and five as the answer choices, i.e., in a multiple-choice setup to allow easier quantitative evaluation. All the proposed models have been trained only on the five figures from the question part, i.e. the training set consisting of 12000x5 images was used, whereas the (sixth) correct answer figure was not used for training. The models were validated (tuned) using the first 50% of the answer figure set and tested on the remaining unseen 50% of the answer figure set from the test/validation set of 100 questions.

Training Details The best (validated) hyperparameters for the Context-RNN-GAN was a GRU with two layers of 400 hidden units each for the generator and a GRU with a single layer and 500 hidden units for the discriminator. The best (validated) hyperparameters for the RNN-GAN model was a GRU with two layers of 400 hidden units each for the generator and a mulilayered perceptron (MLP). The final models of Context-RNN-GAN and the RNN-GAN models use $\lambda_{adv} = 0.05$, $\lambda_p = 1$, and they use $p = 1$ for the DAT-DAR task and $p = 2$ for the next-frame prediction task, similar to (Mathieu et al., 2015) and based on empirical evidence from experiments. The best hyperparameters of the regular RNN was a GRU with one hidden layer of 1000 hidden units for both L_1 and L_2 based loss functions; adding more layers or more hidden units did not help. The best hyperparameters for the feedforward baseline was 2 layers with 1000 hidden units each for both L_1 and L_2 based loss functions and adding more layers or hidden units didn't help. In all of the above models a dropout of 0.50 was applied for each hidden layer. All of the models were trained using the Adam Optimizer (Kingma and Ba, 2014).

Evaluation Metrics In addition to qualitative evaluation via visualizations of the generated images, we importantly also perform quantitative evaluation by matching the generated image embedding with the embeddings of each of the five candidate answer images (based on cosine distance), and returning the closest matching image. The accuracy is then determined by the number of correct choices made by the model with respect to the total test set size.

	College-grade	10th-grade
Age range	20-22	14-16
#Students	21	48
Mean	44.17%	36.67%
Std	16.67%	17.67%
Max	66.67%	75.00%
Min	8.33%	8.33%

Table 1: Human performance on our DAR task.

Results and Analysis

Human Performance on DAR

To test the proficiency of humans on our DAT-DAR dataset, we conducted a set of experiments on two sets of individuals. The problems were divided into sets of 12 problems each and they were given as much time as required to complete all the questions. They were also first given an example of a problem with an explanation of the answer.

Advanced college students: 21 senior students from the computer science department of a premier university took part in the first experiment.

10th-grade high school students: 48 students from the 10th grade of a reputed high school took part in the second experiment.

As can be seen in Table 1, our diagrammatic abstract reasoning task is quite challenging even for humans, with the best performance of strong undergraduate college students being roughly 44% and of 10th-grade students achieving 37% accuracy.

Model Performance on DAR

We next report the model results obtained via our proposed context-RNN-GAN model and its several simpler variants (specified in Models section), combined with the various feature representation methods (specified in Feature Representations section). In Table 2, we first show baseline results for a regular RNN trained on different features. The shallow CNN, Siamese CNN, and HOG features perform best here. Next, we try these best feature settings (and the raw features) on our novel context-RNN-GAN model (and its simpler RNN-GAN variant). As shown in Table 2, our primary context-RNN-GAN model, combined with our novel Siamese CNN features, obtains the best results, even competitive with the performance of 10th-grade human performance. However, there is still a lot of scope for interesting model and feature improvements compared to college-level human performance (and beyond), making this a new challenging task and dataset for the community.

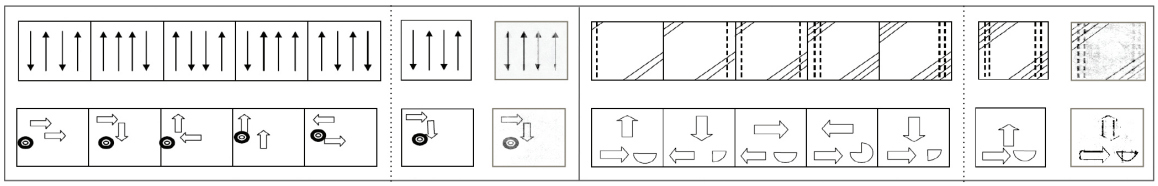

Figure 4: Visualizations of image generation. In each of the four problems, the first five images are the question sequence, the second-last column is the ground-truth, and the last column is our model generation.

Features	Accuracy	
RNN		
	L_2	L_1
Raw Pixels	26.0%	28.4%
HOG Features	28.0%	29.6 %
Autoencoder	11.6%	18.7 %
Image Labels	21.1%	20.4 %
Pretrained CNN	18.6%	18.4 %
Fine-tuned AlexNet	22.3%	20.3%
Shallow CNN	28.6%	31.2 %
Siamese CNN	28.6%	30.8 %
Feedforward Baseline		
Raw Pixels	20.8%	21.9%
HOG	22.3 %	23.8 %
Shallow CNN	24.2 %	23.6 %
Siamese CNN	24.6 %	24.1 %
RNN-GAN		
Raw Pixels	28.1%	
HOG	30.1%	
Shallow CNN	30.6%	
Siamese CNN	31.2%	
Context-RNN-GAN		
Raw Pixels	**30.3%**	
HOG	**34.1%**	
Shallow CNN	**33.0%**	
Siamese CNN	**35.4%**	

Table 2: Primary DAR results of Context RNN-GAN and its variants using several feature representations.

Model	CE	SE
LSTM	341.2	208.1
AE-Conv-LSTM (w/o optical flow)	262.6	-
Context-RNN-GAN	**241.8**	**167.9**

Table 3: Results on Moving-MNIST videos. CE: cross-entropy error, SE: squared error. LSTM: Srivastava et al. (2015), AE-Conv-LSTM: Patraucean et al. (2015).

Next-Frame Generation on Moving-MNIST Videos We also tested our Context-RNN-GAN model on the popular Moving-MNIST task introduced by (Srivastava et al., 2015) and show results compared to their methods as well as the convolution-based LSTM model of (Patraucean et al., 2015). For fair comparison, we report their results of the AE-Conv-LSTM model which does not model the optical flow. As shown in Table 3, we perform better than such comparable state-of-the-art of multiple metrics, CE (cross entropy error) and SE (squared error on image patches).

Qualitiative Generation Visualization

We also present quantitative evaluation via visualizations of the images generated by our model for both the DAR (Fig. 4) and the Moving-MNIST (Fig. 5) tasks. We show cases where the generated image corresponds closely to the correct answer image. For example, in the DAR task (Fig. 4), the model is able to correctly infer and generate next-sequence diagrams with changing arrow directions, number and type of lines in different corners and sides, multiple shapes interacting in different ways, etc. Similarly, for the MNIST task (Fig. 5), our model is able to correctly generate digits moving in different amounts and directions.

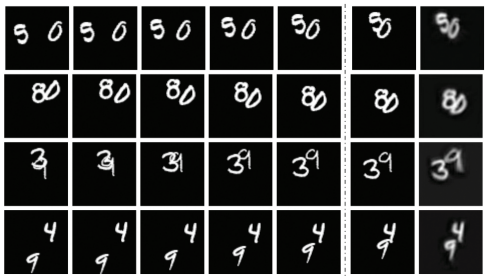

Figure 5: Visualizations for Moving MNIST. First five images: question sequence, second-last column: ground-truth, last column: our model generation.

Conclusion

We presented a novel Context-RNN-GAN model that can generate images for sequential reasoning scenarios such as our task of diagrammatic abstract reasoning. When combined with useful feature representations such as those from Siamese CNNs, our model performs competitively with 10th-grade humans but there is still scope for interesting improvements as compared to college-level human performance, making this a novel challenging task for the generation community. Our sequential GAN model is also general enough to be useful for tasks such as next frame generations in a video (where we also achieve strong results on the Moving-MNIST dataset) and other similarly important AI tasks such as forecasting and simulation.

Acknowledgment

We thank the reviewers and Dhruv Batra (GaTech), Kundan Kumar (IIT Kanpur), Devi Parikh (GaTech), Deepak Pathak (UC Berkeley), Greg Shakhnarovich (TTIC), and Shubham Tulsiani (UC Berkeley) for their helpful feedback.

References

Aggarwal, D. R. S. (2016). *A Modern Approach To Non-Verbal Reasoning*. S. Chand, New Delhi, India.

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. (2015). VQA: Visual question answering. In *ICCV*.

Bennett, G., Seashore, H., and Wesman, A. (1947). Differential aptitude tests. *Psychological Corporation*.

Berdie, R. F. (1951). The differential aptitude tests as predictors in engineering training. *Journal of Edu. Psychology*.

Bringsjord, S. and Schimanski, B. (2004). Pulling it all together via psychometric ai. In *AAAI*.

Chopra, S., Hadsell, R., and LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. In *CVPR*.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *CVPR*.

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

Evans, T. (1964). A program for the solution of a class of geometric-analogy intelligence-test questions. Technical report, DTIC Document.

Falkenhainer, B., Forbus, K., and Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial intelligence*, 41(1):1–63.

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

Gregor, K., Danihelka, I., Graves, A., and Wierstra, D. (2015). Draw: A recurrent neural network for image generation. *arXiv:1502.04623*.

Gupta, R. (2016). *All About Reasoning (Verbal and Non-Verbal)*. Ramesh Publishing House.

Im, D. J., Kim, C. D., Jiang, H., and Memisevic, R. (2016). Generating images with recurrent adversarial networks. *arXiv:1602.05110*.

Jha, P. N. and Siddiquee, J. R. (2011). *Magical Book Series Non-Verbal Reasoning*. BSC Publication Co. Pvt. Ltd., Delhi, India.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv:1412.6980*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105.

Mathieu, M., Couprie, C., and LeCun, Y. (2015). Deep multi-scale video prediction beyond mean square error. *arXiv:1511.05440*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv:1301.3781*.

Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv:1411.1784*.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Novak, G. and Bulko, W. (1992). Uses of diagrams in solving physics problems. In *AAAI*.

Oh, J., Guo, X., Lee, H., Lewis, R. L., and Singh, S. (2015). Action-conditional video prediction using deep networks in atari games. In *NIPS*.

Pathak, D., Krähenbühl, P., Donahue, J., Darrell, T., and Efros, A. (2016). Context encoders: Feature learning by inpainting. In *CVPR*.

Patraucean, V., Handa, A., and Cipolla, R. (2015). Spatio-temporal video autoencoder with differentiable memory. *arXiv:1511.06309*.

Petrovic, N., Ivanovic, A., and Jojic, N. (2006). Recursive estimation of generative models of video. In *Proc. of CVPR*, volume 1, pages 79–86. IEEE.

Prade, H. and Richard, G. (2011). Analogy-making for solving iq tests: A logical view. In *ICCBR*.

Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv:1511.06434*.

Ranzato, M., Szlam, A., Bruna, J., Mathieu, M., Collobert, R., and Chopra, S. (2014). Video (language) modeling: a baseline for generative models of natural videos. *arXiv:1412.6604*.

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. (2016). Generative adversarial text to image synthesis. *arXiv:1605.05396*.

Seo, M., Hajishirzi, H., Farhadi, A., Etzioni, O., and Malcolm, C. (2015). Solving geometry problems: Combining text and diagram interpretation. *Proceedings of EMNLP*.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv:1312.6229*.

Sijwali, B. and Sijwali, I. (2016). *Non-Verbal Reasoning*. Arihant Publications (India) LTD.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Srivastava, N., Mansimov, E., and Salakhutdinov, R. (2015). Unsupervised learning of video representations using lstms. *CoRR, abs/1502.04681*, 2.

Stern, W. (1914). *The psychological methods of testing intelligence*. Number 13 in 1. Warwick & York.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P. (2008). Extracting and composing robust features with denoising autoencoders. In *ICML*, pages 1096–1103. ACM.

Vondrick, C., Pirsiavash, H., and Torralba, A. (2016). Generating videos with scene dynamics. *arXiv:1609.02612*.

Wang, H., Gao, B., Bian, J., Tian, F., and Liu, T.-Y. (2015). Solving verbal comprehension questions in iq test by knowledge-powered word embedding. *arXiv:1505.07909*.

Wang, X. and Gupta, A. (2016). Generative image modeling using style and structure adversarial networks. *arXiv:1603.05631*.

Yu, Q., Yang, Y., Song, Y.-Z., Xiang, T., and Hospedales, T. M. (2015). Sketch-a-net that beats humans. In *Proc. of BMVC*, pages 7–1.

Zhu, J., Krähenbühl, P., Shechtman, E., and Efros, A. (2016). Generative visual manipulation on the natural image manifold. In *ECCV*.