Improving Task-Oriented Language Grounding using Disentangled Representations

Gaurav Mittal
Robotics Institute
Carnegie Mellon University
gauravm@andrew.cmu.edu

Tanya Marwah
Robotics Institute
Carnegie Mellon University
tmarwah@andrew.cmu.edu

Abstract

Task-oriented language grounding refers to the process of mapping the visual scene and actions in the environment with language instructions in order to perform a task based on a natural language instruction. Domain adaption and specifically, zero-shot task generalization in this scenario is an open research problem. We propose a novel multi-stage algorithm to improve the performance of the agent over existing methods. The proposed model aims to learn a disentangled representation of the visual scene using β-VAE and then combines it with the instruction-based text representation using a soft-attention mechanism. This generates a representation of the 'instruction-conditioned' visual scene which is robust to variations with respect to objects, actions and attributes like shape and color. This representation is then used to learn a policy using standard reinforcement learning methods to execute the instruction in the given scene. In addition to β-VAE, we also incorporate a recurrent attention mechanism in the VAE (making it similar to DRAW (1)) in an attempt to allow the foreground be distinguished from the background. We tested the effectiveness of the algorithm by running it on a 3D game simulator, ViZDoom. We found that for certain values of disentanglement parameter β, the training performance/accuracy indeed improves with different values of β performing well in different experimental settings.

1 Introduction

Consider the scenario of a restaurant equipped with next-generation robots whose task is waiting the tables. You are the manager of this high-tech restaurant with the task of instructing the robots about what to do in your native language. Naturally, the instructions will be quite similar in structure but will exhibit a large amount of variability in terms of objects and their various characteristic, such as which table to attend to and what food to serve to which table. In order to execute the given instruction, the robot should be able to simultaneously understand the scene based on raw pixel input, ground the concepts underlying the instruction to the visual elements and actions in the environment, and efficiently explore the environment to look for the relevant objects and choose the correct action towards duly fulfilling the instruction. Moreover, there is the possibility of encountering an instruction which, despite consisting of the same vocabulary as training, has never been seen before in terms of the combination of the objects and actions involved.

Generalization to such related but new tasks in a zero-shot manner requires the robots to build an understanding of the scene where the different concepts and their associations are learned separately and simultaneously so that inference is possible on novel compositions of these concepts. Due to data inefficiency and high-dimensionality of the data, the learnt policies are typically brittle to changes in the input data distribution and also lack model interpretability. The ability of the RL agent to deal with changes to the input distribution (known as *domain adaptation*) can be improved by learning representations which capture an underlying low-dimensional factorized representation of the world that is not task or domain specific. To this end, we propose to learn a disentangled representation of the visual scene using a β-VAE which is then combined with the language instruction to generate an embedding of the state which is much more robust for language-grounding based task generalization.

Disentangled representations are defined as interpretable, factorized latent representations where the latent units are sensitive to changes in single ground truth factors of variation used to generate the visual world, while being invariant to changes in other factors. Therefore, by learning an underlying low-dimensional factorized representation of the world the learned model can be allowed to be made task or domain agnostic.

While disentangled representations are helpful in devising a factorized low-dimensional representation of the visual scene, we additionally require the model to be able to distill out the concepts that are integral to solving the task from other redundant aspects of the visual scene. In other words, we need the model to be able to distinguish the foreground elements of the scene from the background. Humans have this amazing ability to do so due to foveation of the human optic system. Previous works have been able to emulate this ability through the use of Recurrent Attention Mechanism (2). More specifically, we use Deep Recurrent Attentive Writer (DRAW) (1) as part of our β-VAE. This enables the model to learn to reconstruct the visual scene as parts which in turn allows the model to capture partial glimpses of the scene making it effectively focus on the foreground separately from the background. As we show later through our experiments, using DRAW drastically improves the reconstruction as compared to vanilla-VAE.

The primary contribution of this paper is to present an analysis on how using disentangled representations in addition to reconstructing with a recurrent attention mechanism can potentially help improve the performance of the RL policy on task-oriented language grounding. Through various experiments, we attempt to establish that different degrees of disentanglement indeed affect the task performance, performing both better and worse than the previous methods.

2 Related Work

Prior work has explored use of deep learning for playing games, where they try to learn a policy for a variety of tasks, including navigation using raw visual pixel information (3; 4). Several previous works have tried to ground language with action sequence, especially in the context of robotics (5). A very recent work that our method draws the most similarity to is (6) trains an end-to-end learning model to address visual language grounding in a continuous 3D setting using raw-pixel inputs. Our work differs in that we make use of disentangled representations from the visual scene in place of an end-to-end trained convolutional encoder.

In a different line of research a large body of literature addresses the task of projecting raw observations into a factorized representation (7; 8; 9). Moreover, the theoretical utility of such disentangled representations for supervised and reinforcement learning has been described in (10; 11). Our work is an attempt at showcasing the importance of the same, especially in the task of language grounding. As an extension to the normal VAE, (12) introduced β-VAE by introducing a β factor in the KLD loss to encourage a more isotropic (disentangled) latent representation. DARLA (13) use this technique to improve zero-shot task generalization in reinforcement learning. As another extension, considering the important of recurrent attention mechanism (2), Deep Recurrent Attentive Writer (DRAW) (1) reconstructs the visual scene iteratively allowing the model to focus only of certain parts of the scene at any given time. Such a mechanism further allows the latent representation to perceive the visual scene compositionally and not holistically. Our approach takes advantage of both β factor and recurrent attention mechanism to build an architecture that we call β-DRAW. Through the experiments, we find that not only β-DRAW learns to factorize the concepts in the latent representation but it also learns to semantically separate the foreground elements from the background.

3 Methodology

Assume $\mathcal{M}$ is a set of all MDPs, such that each MDP $\mathcal{D}_i$ is sampled from $\mathcal{M}$. We further assume that $\mathcal{M}$ is defined over a state-space $\hat{\mathcal{S}}$ which contains all possible variations in the high-level attributes of visual scene. We assume that the state-space of $\mathcal{D}_i$ is a subset of $\hat{\mathcal{S}}$. In other words, we assume that there is a shared underlying structure between each and every sample drawn from $\mathcal{M}$.

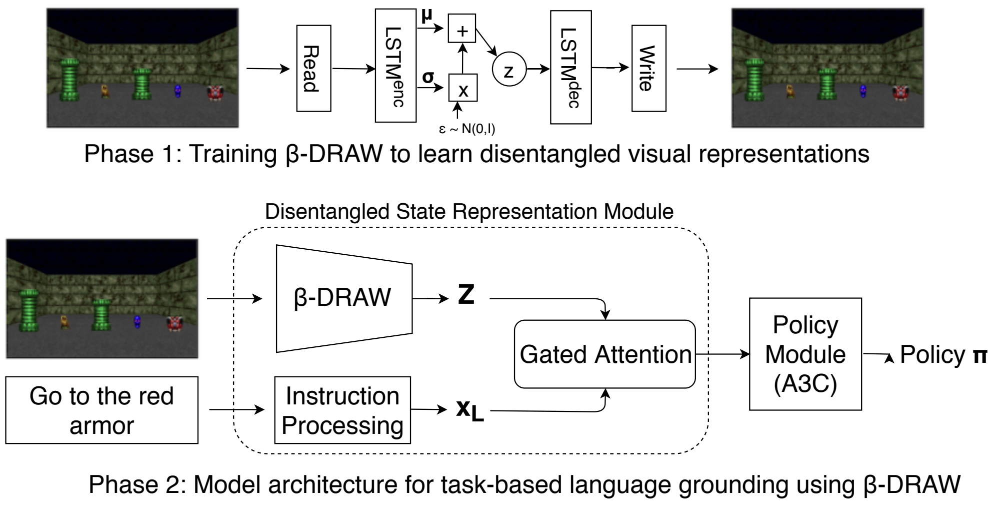

Read
LSTM^enc
μ
σ
+
x
ε ~ N(0,I)
z
LSTM^dec
Write
Phase 1: Training β-DRAW to learn disentangled visual representations
Disentangled State Representation Module
β-DRAW
Z
Go to the red armor
Instruction Processing
x_L
Gated Attention
Policy Module (A3C)
Policy π
Phase 2: Model architecture for task-based language grounding using β-DRAW

Figure 1: The two phases of the proposed methodology

Let $\mathcal{S}^l$ be the observation space of $\mathcal{D}_i$ corresponding to the raw pixel space and $\mathcal{S}^z$ be the internal latent space. We assume that the raw observations in $\mathcal{S}^o$ are sampled from a distribution that is defined over all possible factors of generation of $\hat{\mathcal{S}}$. This means that while the true data generative factors of variation $\hat{\mathcal{S}}$ remain the same the raw observation space can be made to differ for different tasks. Example, while the training space might just contain a ball colored blue, the testing space might contain a ball colored red (here the generative factors are object and color).

A typical RL agent operating in MDP $\mathcal{D}_i \in \mathcal{M}$ learns an end-to-end mapping from the raw observation space $\mathcal{S}_i^o$ to the action space $\mathcal{A}_i$. While doing so the agent implicitly learns a function $\mathcal{F} : \mathcal{S}_i^o \to \mathcal{S}_i^z$ mapping the high dimensional raw observation space to a low dimensional latent space. However, learning this function implicitly will result in an entangled representation which will not generalize well if to unseen samples during testing. Therefore, we introduce a β-VAE to explicitly learn the latent space in terms of factorized data generative models. Continuing from the previous example, originally the model will associate the color blue with the object ball, therefore during testing, it will fail to understand the red ball. However, with disentangled features, the concept of ball and color are separate making it easier for the model to accommodate to a novel combination of the two.

3.1 Disentangled State Representation Module

We make use of β-VAE (12), a state-of-the-art unsupervised model for automated discovery of factorized latent representations from raw images. It is a modification over variational autoencoder framework introduced by (9), that controls the nature of the latent representations by introducing an adjustable hyperparameter β to balance reconstruction accuracy with latent channel capacity and independence constraints. The objective that is maximized is as follows:

$$\mathcal{L}(\theta, \phi; \boldsymbol{x}, \boldsymbol{z}, \beta) = E_{q_\theta(z|x)}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \beta D_{KL}(q_\theta(\boldsymbol{z}|\boldsymbol{x}) \| p(\boldsymbol{z}))$$

where θ, ϕ are the parameters of the encoder and the decoder respectively. A well chosen β results in a more disentangled representation $\boldsymbol{z}$ by limiting the capacity of the latent information channel, and hence encouraging a more efficient factorized encoding.

3.2 Deep Recurrent Attentive Writer (DRAW)

In addition to producing a disentangled representation of the visual scene via β-VAE, we incorporate a recurrent attention mechanism into the VAE framework in the form of DRAW (1). DRAW allows the visual scene to be reconstructed over a sequence of steps such that each step focuses on just a portion of the scene and it's possible to go back and forth while reconstructing any particular region of the scene. By doing so, DRAW can give more focus to the relevant parts of the scene, that is foreground while abstracting out the background.

To allow for scene reconstruction in a sequence of steps, DRAW consists of an LSTM both as the encoder $LSTM^{enc}$ and decoder $LSTM^{dec}$ of the VAE. The attention mechanism is implemented as a grid of Gaussian filters, whose grid center, variances and stride are learned by the network (as in (1)). For every time step t, a glimpse r_t is read from the input frame and is fed to $LSTM^{enc}$ to learn the approximate posterior Q. z is then sampled from Q and is decoded using $LSTM^{dec}$. The output from $LSTM^{dec}$ is used to (1) generate a patch to be 'written' on to the canvas and (2) the attention parameters for the grid of Gaussian filters for localizing the patch on the canvas. The following equations explain these steps:

$$\begin{aligned}
r_{k-1_t} &= conv(read(Y_{k-1})) \\
r_t &= conv(read(Y_{k_{t-1}})) \\
s_{s_t} &= attention(h_{lang}, [r_{k-1_t}, h^{dec}_{t-1})] \\
h^{enc} &= LSTM^{enc}(r, s_s, s_l, h^{dec}_{t-1}) \\
z &\sim Q(z|h^{enc}) \\
h^{dec}_t &= LSTM^{dec}(z, s_s, s_l, h^{dec}_{t-1}) \\
Y_{k_t} &= Y_{k_{t-1}} + write(deconv(h^{dec}_t))
\end{aligned}$$

In the variational autoencoder, we take $P(z|U,V) = P(z) \sim \mathcal{N}(0, I)$. This simplifies the empirical variational bound to (note that we minimize the negative of the bound, as in VAEs (9)):

$$\mathcal{L} = -\left(\frac{1}{S}\sum_{s=1}^{S} log\ P(X|z^s) - KL(Q(z|X)||P(z))\right) \tag{1}$$

with

$$KL(Q(z|X)||P(z)) = \frac{1}{2}\left(\sum_{t=1}^{T}\mu_t^2 + \sigma_t^2 - \log\sigma_t^2\right) - T/2 \tag{2}$$

where t denotes the time-step over which the frames are generated as before, and z^s denotes the s^{th} sample taken from the z distribution among a set of total S samples which are used to compute the likelihood term.

We multiply the KL-divergence term with β to allow for a more disentangled representation z as discussed in Section 3.1 effectively making the VAE β-DRAW.

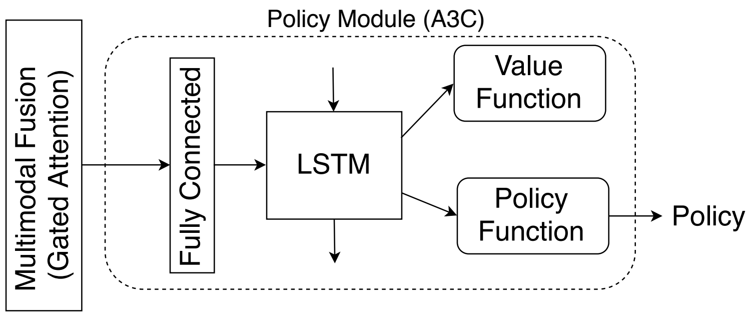


Figure 2: Architecture of the A3C policy module

3.3 Multimodal Fusion Output

We train the β-DRAW architecture as discussed above over a set of frames from the VizDoom environment. After training, the encoding generated by the autoencoder is used as the image representation. This representation is combined with an embedding generated for a given instruction using a Gated-Attention unit as used by (6). In the Gated-Attention unit, the instruction embedding is passed through a fully-connected linear layer with a sigmoid function. The output dimension of this linear layer is equal to the dimension of z from β-DRAW. The output of this linear layer is called attention vector a_L which is multiplied elementwise with z to generate what is called the Multimodal Fusion Output as follows,

$$M_{GA}(z, x_L) = z \odot x_L \tag{3}$$

The purpose of having a Gated-Attention unit over simple concatenation is to allow different aspects over the instruction embedding to react to different dimensions of the image representation separately and simultaneously. Moreover, by allowing for the image representation to be disentangled we can enable the visual and textual modalities to fuse better across dimensions providing a better output.

3.4 Policy Learning

The fusion output from above will serve at the state representation for our RL model to perform task-based language ground. As the RL model, we use the Asynchronous Advantage Actor-Critic (A3C) algorithm (14), which uses a deep neural network to learn the policy and the value function. In order to reduce the variance of the policy gradient we use Generalized Advantage Estimator (15).

As suggested by (6), we also include an LSTM layer in the A3C model architecture followed by the fully connected layers as shown in Figure 2. The LSTM layer is introduced so that the agent can have some memory of previous states. This could help the agent since the environment is partially observable and it might explore states where all objects are not visible and need to remember the objects seen previously.

4 Experimental Setup

4.1 Environment

To test our model, we use the environment built by (6) on top of the ViZDoom API (16), that is based on Doom, a classic first person shooting game. The environment allows the agent to execute a natural language instruction and obtain a positive reward on successful completion of the task. Each scenario in the environment comprises of an agent along with a list of objects (a subset of ViZDoom objects) of which one is the correct object to reach while rest are incorrect objects. The agent is allowed to perform the following actions – turn right, turn left and move forward. For instance, given an instruction "Go to the red armor", the task is considered successful if the agent is able to reach the *red armor*

An *instruction* is a combination of (action, attribute(s), object) triple. The objects can have various visual attributes such as color, shape and size. Each time an instruction is selected, the environment generates a random combination of incorrect objects and the correct object in randomized locations. The environment thus allows to create multiple episodes for a given instruction. This introduces an additional difficulty in the task, that is to understand that the same instruction can refer to different objects in different episodes. For instance, "Go to the green object" can refer to a green pillar in one episode and a green torch in another episode. Similarly, an instruction like "Go to the pillar" can refer to multiple correct objects in the same episode. Moreover, there are challenges like occlusion which necessitates efficient exploration in the environment.

The environment provides different modes with varying difficulty - Easy, Medium and Hard. Given the time and compute resources, we show our experiments on Easy and Medium scenario and compare our method with (6) on the same. In the **Easy** setting, the agent is spawned at a fixed location. The candidate objects are spawned at five fixed locations along a single horizontal line along the field of view of the agent. In the **Medium** setting, the candidate objects are spawned in random locations, but the environment ensures that they are in the field of view of the agent.

4.2 Hyperparameters

In order to compare our method with (6), we intend to an experimental setup that is as close as possible to their method. So we restrict the number of objects to 5 for each episode (one correct and four incorrect). Out of a total of 70 instruction, 55 are used as training set and rest are held out to test for zero-shot task generalization. The episode length is restricted to 30 steps. We evaluate the accuracy of the agent at the metric which corresponds to the success rate of reaching the correct object before the episode terminates.

As the first stage of the methodology, we train β-DRAW to reconstruct the frames of the ViZDoom environment by learning a disentangled low-dimensional representation over the frames. We ran the ViZDoom environment in the Medium setting for 1000 episodes to generate a dataset of roughly 20,000 frames. We chose Medium setting since it encompasses both Medium and Easy setting in terms of frames. Each frame was passed as an RGB image of size $3 \times 64 \times 64$ to β-DRAW. The encoder and decoder LSTMs have a hidden layer size of 512. The grid of 15 Gaussian filters is used for reading/writing glimpses. The reconstruction happens in a total of 10 timesteps. The latent representation z is chosen to be 50-dimensional.The model is trained using Adam optimizer with a learning rate of 0.001 in PyTorch for 1000 epochs with batch size of 64 frames.

After training β-DRAW, the next stage is to retrieve a state representation for training the policy for task-based language grounding. Every time a frame is returned by the ViZDoom environment, it is resized to $3 \times 64 \times 64$ and is fed to trained model of β-DRAW to generate a sequence of latent representations $Z = z_1, z_2, , z_{10}$. This sequence is concatenated to form the image representation for size $50 \times 10 = 500$. The input instruction is simultaneously encoded to be of the same size (500) via Gated Recurrent Unit (GRU) and is then combined with the image representation via Gated Attention mechanism to form the Multi-modal fusion output as discussed in Section 3.3.

The A3C policy learning module has a linear layer of size 256 followed by an LSTM layer of size 256 which encodes the history of state observations. The LSTM layer's output is fully-connected to a single neuron to predict the value function as well as three other neurons to predict the policy function. The model is trained using Stochastic Gradient Descent with a learning rate of 0.001. Discount factor of 0.99 is used for expected reward. Mean squared loss between the estimated value function and discounted sum of rewards is used for training the value function while policy gradient loss is used for training the policy function. The model is trained in PyTorch using 12 GPU threads.

5 Results

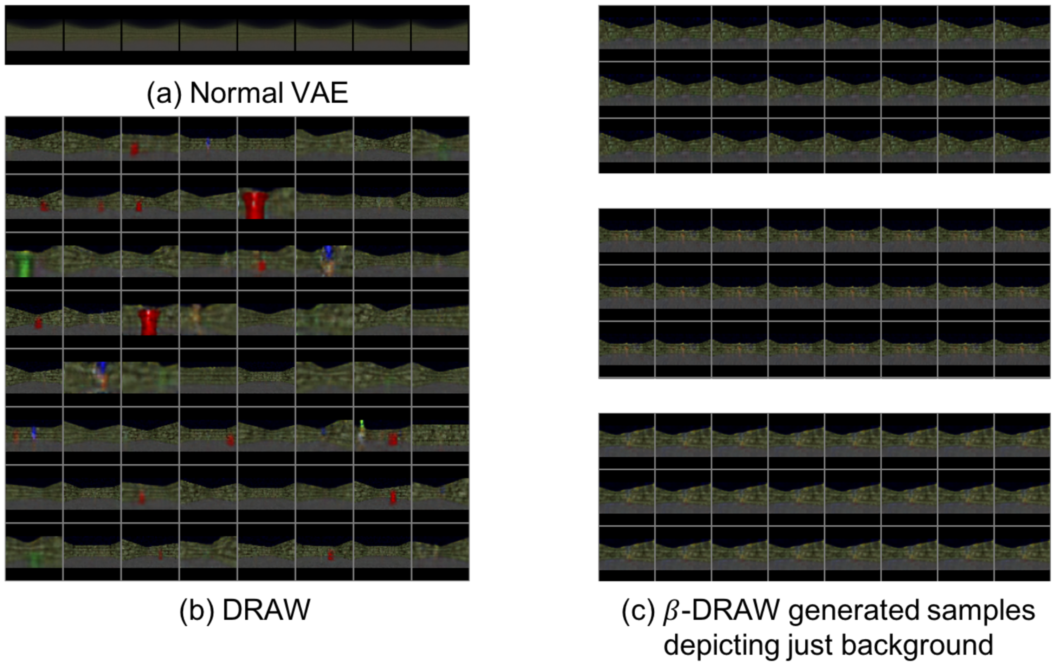
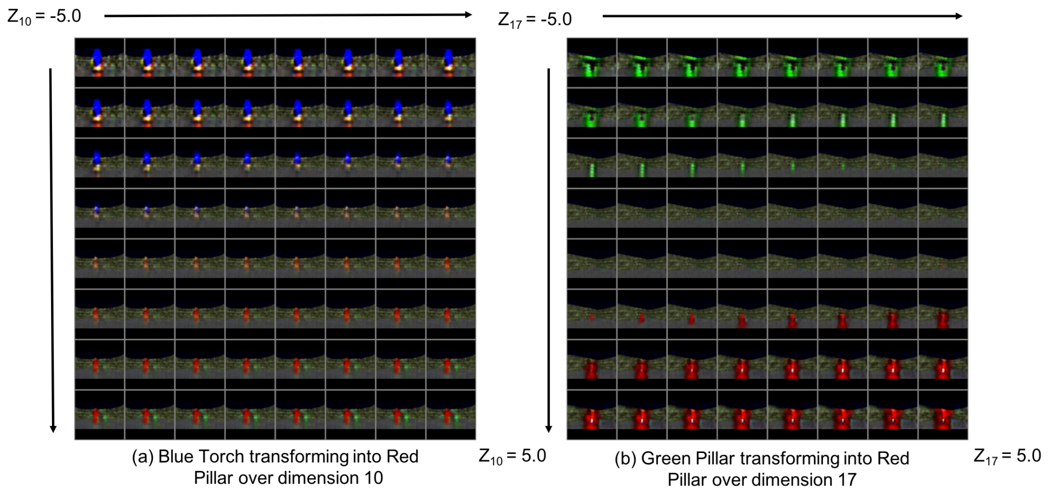

(a) Normal VAE
(b) DRAW
(c) β-DRAW generated samples depicting just background

Figure 3: Comparison of frame reconstruction between (a) Normal VAE and (b) DRAW. (c) shows samples generated by β-DRAW for specific dimensions that completely omits foreground and shows just the background.

Figure 3 shows a comparison of frame reconstruction between a normal Variational Autoencoder and β-DRAW as discussed above. The normal VAE serves as the baseline for reconstruction. Clearly, the reconstruction is significantly better for β-DRAW than normal VAE. In case of normal VAE, the model seems to have averaged out visual scene. In doing so, it completely disregards the relevant foreground objects and reconstruct a highly blurry background. On the other hand, the samples generated by DRAW are much more clear with a diverse set of foreground objects in the visual scene. Due to the recurrent attention mechanism implemented by DRAW, it allows DRAW to focus on only specific parts of the frame at any particular step. This in turn enables it to perform a semantic separation of the visual scene into foreground and background. By doing so, DRAW is able to consider the objects in the frame to be relevant to the generation, thus avoiding the averaging effect.



Z_{10} = -5.0
Z_{10} = 5.0
(a) Blue Torch transforming into Red Pillar over dimension 10
Z_{17} = -5.0
Z_{17} = 5.0
(b) Green Pillar transforming into Red Pillar over dimension 17

Figure 4: Samples generated by β-DRAW as value of z is regressed over a particular dimension. This shows how β-DRAW has learned to associate a particular concept(object) with a particular dimension, thus disentangling them over different dimensions. Here β=20.

We now qualitatively analyze the ability of β-DRAW to learn a low-dimensional factorized representation of the visual scene. To this end, we allow samples to be randomly generated by β-DRAW such that we manually specify the value of one of the dimensions of z keeping values of all other dimensions fixed. Figure 4 shows the samples generated when the value of z is regressed uniformly in $[-5, 5]$ for the 10^{th} and 17^{th} dimension. The samples are shown in the form of a grid where they are arranged row-wise. From the figure, we can clearly observe that the model is able to associate a particular concept (here a foreground object) with a particular dimension such that when we change the value of only one particular dimension, we can observe a smooth transformation taking place from one kind of object to another such that all the attributes of the visual scene remain the same. We saw this effect in at least 10 dimensions out of the 50 total dimensions of z. Moreover, Figure 3.1 (c) shows samples generated by β-DRAW for some other dimensions which seems to have completely omitted the foreground and just depicts a different variant of the background in the scene. This kind of disentanglement between background and foreground is due to DRAW being used for reconstruction in tandem with β. DRAW allows the model to perceive the foreground separate from the background due to the recurrent attention mechanism. This enables the disentanglement to focus on the two separately giving an improved disentanglement in addition to better reconstruction.

Figure 5 compares the performance (accuracy) of A3C policy using state representations obtained from β-DRAW for different values of β (that is for different degrees of disentanglement). From the plot, we can observe that in both experimental settings, model with $\beta > 1$ performs significantly better over time in comparison to model with $\beta = 1$ which essentially is normal DRAW. We also compared our results with A3C-GA from (6) which we used as baseline for our experiments. In Figure 5, the accuracy trend for A3C-GA is shown in orange. We can infer from the plot that for some values of β, the model is able to solve the task quicker than A3C-GA. As discussed in (12), the optimal value of β is highly subjective to the environmental setting. Even in our case, we notice that though $\beta > 1$ is in general better performing, there is no clear winner in terms of performance over both experimental settings. Some values of β perform better in the Easy setting than A3C-GA while performing worse in the Medium setting and vice versa.

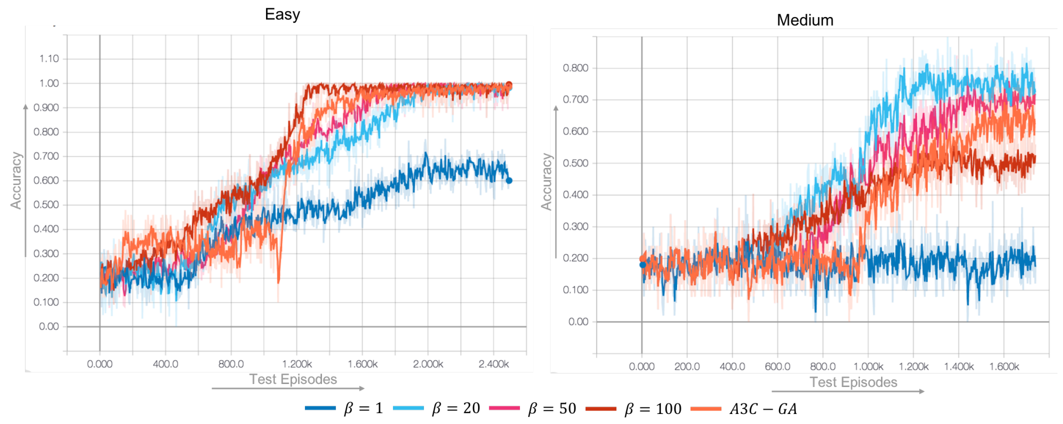


Figure 5: Performance comparison of A3C-GA with β-DRAW for different values of β with baseline A3C-GA

Model	Easy		Medium	
	MT	ZSL	MT	ZSL
A3C-GA	1.0	0.81	0.89	0.75
$\beta = 1$	0.76	0.43	0.25	0.13
$\beta = 20$	1.0	0.75	**0.90**	**0.77**
$\beta = 50$	1.0	0.78	0.86	0.70
$\beta = 100$	1.0	**0.82**	0.88	0.75

Table 1: Accuracy comparison of normal A3C-GA with A3C-GA using disentangled representations on Multi-Task Generalization and Zero-shot Generalization.

The accuracy of the different methods are evaluated on two scenarios - (1) Multi-Task Generalization (MT) where agent is evaluated on unseen maps using instructions from the training set and (2) Zero-shot Task Generalization (ZSL) where agent is evaluation on unseen test instructions. Table 1 shows the summary of the accuracy evaluations for different models. We can observe that $\beta > 1$ performs significantly better overall than $\beta = 1$. Moreover, we can also observe that our method for $\beta > 1$ performs comparably with the baseline A3C-GA. In fact, in some cases our method is performing slightly better than the baseline which shows that using disentangled representations holds potential in enabling the RL policy to better perform the task.

6 Discussion and Conclusion

In this paper, we presented a novel model to improve the performance on task-based language grounding by using low-dimensional disentangled representations and recurrent attention mechanism to encode the visual scene. Through the experimental analysis, we found that β-DRAW not only learns a factorized representation by associating different concepts to different dimensions of the latent representation, but it also learns a semantic separation of the foreground and background by focusing on only certain portions of the scene at any given time. By doing so, the model is able to distill out the relevant components of the visual scene and later allow an improved fusion with the instruction embedding to generate a robust state representation. We found that state representations with certain degree of disentanglement improves the training of the RL policy and also performs comparably (and sometime slightly better) than the baseline. Although not experimented directly, we also believe that using β-DRAW can help in circumventing the use of auxiliary approaches like perceptual similar loss via Denoising Autoencoder to improve the sample reconstruction as done in (13). Given the potential of disentangled representations and recurrent attention mechanism in generating better state representations, it should be possible to improve the performance of tasks that lies at the cusp of vision and reinforcement learning.

References

[1] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra, “Draw: A recurrent neural network for image generation,” in *International Conference on Machine Learning*, pp. 1462–1471, 2015.

[2] V. Mnih, N. Heess, A. Graves, *et al.*, “Recurrent models of visual attention,” in *Advances in neural information processing systems*, pp. 2204–2212, 2014.

[3] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, “Mastering the game of go without human knowledge,” *Nature*, vol. 550, no. 7676, p. 354, 2017.

[4] T. D. Kulkarni, A. Saeedi, S. Gautam, and S. J. Gershman, “Deep successor reinforcement learning,” *arXiv preprint arXiv:1606.02396*, 2016.

[5] S. Guadarrama, E. Rodner, K. Saenko, N. Zhang, R. Farrell, J. Donahue, and T. Darrell, “Open-vocabulary object retrieval.,” in *Robotics: science and systems*, vol. 2, p. 6, Citeseer, 2014.

[6] D. S. Chaplot, K. M. Sathyendra, R. K. Pasumarthi, D. Rajagopal, and R. Salakhutdinov, “Gated-attention architectures for task-oriented language grounding,” *arXiv preprint arXiv:1706.07230*, 2017.

[7] S. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, G. E. Hinton, *et al.*, “Attend, infer, repeat: Fast scene understanding with generative models,” in *Advances in Neural Information Processing Systems*, pp. 3225–3233, 2016.

[8] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum, “Deep convolutional inverse graphics network,” in *Advances in Neural Information Processing Systems*, pp. 2539–2547, 2015.

[9] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.

[10] Y. Bengio, “The consciousness prior,” *arXiv preprint arXiv:1709.08568*, 2017.

[11] K. Ridgeway, “A survey of inductive biases for factorial representation-learning,” *arXiv preprint arXiv:1612.05299*, 2016.

[12] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, “beta-vae: Learning basic visual concepts with a constrained variational framework,” 2016.

[13] I. Higgins, A. Pal, A. A. Rusu, L. Matthey, C. P. Burgess, A. Pritzel, M. Botvinick, C. Blundell, and A. Lerchner, “Darla: Improving zero-shot transfer in reinforcement learning,” *arXiv preprint arXiv:1707.08475*, 2017.

[14] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, “Asynchronous methods for deep reinforcement learning,” in *International Conference on Machine Learning*, pp. 1928–1937, 2016.

[15] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, “High-dimensional continuous control using generalized advantage estimation,” *arXiv preprint arXiv:1506.02438*, 2015.

[16] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaśkowski, “Vizdoom: A doom-based ai research platform for visual reinforcement learning,” in *Computational Intelligence and Games (CIG), 2016 IEEE Conference on*, pp. 1–8, IEEE, 2016.