DISENTANGLED REPRESENTATION LEARNING USING (β-)VAE AND GAN

Mohammad Haghir Ebrahimabadi

Submitted to the faculty of the University Graduate School

in partial fulfillment of the requirements

for the degree

Masters

in the Department of Computer Science,

Indiana University

July 2022

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements for the degree of Masters.

Masters Committee

Professor David Crandall, Chairperson, Computer Science

Professor Zoran Tiganj, Computer Science

July 29th, 2022

Acknowledgments

I wish to recognize the effort that my supervisor Professor David Crandall put in for training me to be a knowledgeable, thinker, caring, and communicative graduate. I am also grateful for the fundings and computing machines that I received from the Computer Science Department of the Indiana University Bloomington and National Institute of Health for my research.

Mohammad Haghir Ebrahimabadi

DISENTANGLED REPRESENTATION LEARNING USING (β-)VAE AND GAN

Given a dataset of images containing different objects with different features such as shape, size, rotation, and x-y position, and a Variational Autoencoder (VAE), creating a disentangled encoding of these features in the hidden vector space of the VAE was the task of interest in this thesis. The dSprite dataset provided the desired features for the required experiments in this research. After training the VAE with combinations of a Generative Adversarial Network (GAN), each dimension of the hidden vector was disrupted to explore the disentanglement in each dimension. Note that the GAN was used to improve the quality of output image reconstruction.

Professor David Crandall, Chairperson, Computer Science

Professor Zoran Tiganj, Computer Science

Contents

Chapter 1

Introduction

Truly understanding data might require identifying its generative factors — a concept that is more formally known as disentanglement. Classical approaches such as Principal Component Analysis (PCA) (Zietlow et al. 2021) have been developed for this purpose using linear algebra. In addition, Variational Autoencoder (VAE) (Kingma & Welling 2013) is a learning-based architecture that aims to represent the data in its disentangled latent space. In other words, VAEs were developed for learning a latent manifold in which the axes align with independent generative factors of the data. Zietlow et al. (2021) argued that VAEs recover the nonlinear principal components of the data. In addition β-VAEs (Higgins et al. 2016) are a modified version of VAEs that when $\beta > 1$, weigh in more for disentanglement by sacrificing reconstruction quality. In this project, the goal is to explore the capacities of β-VAEs for learning a disentangled representation, specifically to what degree the position of a moving object in the input frames can be encoded in the latent space.

Learning the posterior distribution of continuous latent variables in probabilistic models is intractable. Kingma & Welling (2013) proposed a Variational Bayesian (VB) approach for approximating this distribution that can be learned using stochastic gradient descent. This approach can be used in different settings where latent variables of a model required to be learned e.g. in supervised models with latent variable and learning complicated noise distributions. One example of using this approach for noise identification and removal is presented in (Wan et al. 2020). In this work VB is used in an encoder-decoder setting which is known for Variational AutoEncoder (VAE). In VAEs, the goal is learning the latent variables for input reconstruction. The loss in this setting is consisted of a reconstruction loss and a disentanglement loss. One variation of VAE is β-VAE (Higgins et al. 2016, Burgess et al. 2018) where the second term in the loss function of VAEs can be controlled using a parameter β. This parameter can be used to establish a trade-off between the reconstruction accuracy and disentanglement of the learned representations in the latent space. Thus, there are two measures that are of interest in this setting: (I) the amount of disentanglement (II) the reconstruction output accuracy. Factor-VAE Metric (FVM) (Kim & Mnih 2018) and Mutual Information Gap (MIG) (Chen et al. 2018) are developed for quantifying disentanglement. In addition, Frechet Inception Distance (FID) (Heusel et al. 2017) was developed for measuring the generated output quality. Note that, at some point in the project, I also used a GAN module. Details of relevant previous works on GAN are mentioned in Chapter 2.

Chapter 2

Methods

β-VAEs ($\beta > 1$) showed to have higher performance in disentanglement representation learning and generation quality compared to their peers such as VAEs ($\beta = 1$), InfoGAN (Chen et al. 2016), and DC-IGN (Kulkarni et al. 2015) on the dSprite dataset (Matthey et al. 2017). In this project, first I aimed to reproduce the output of β-VAEs for various learning configurations such as the number of latent dimension, β parameter value, and the learning rate. This phase of the project required understanding of β-VAEs loss function and auto-encoders structure, and their implementation. Essentially, the objective function in β-VAE is to optimize a modified lower bound of the marginal likelihood as follows:

$$\mathbb{E}_{x \sim p(x)}[\log p(x)] \geq \mathbb{E}_{z \sim q_\phi(z|x)}[\log p_\theta(x|z)] - \beta D_{KL}(q_\phi(z|x)||p(z)) \tag{2.1}$$

where x is a data point and the first term aims for a higher generation quality and the KL divergence term (Burgess et al. 2018) forces the posterior to be closer to the prior $p(z)$ which results in a more disentangled representation. Note that higher values of β sacrifices the generation quality in favor of a more disentangled representation in latent space.

Throughout the first phase of the project, I observed that even by setting the value of β to values < 1, the generation quality was far from the ground truth. This was observed even though the experiments were performed on a synthetic dataset in a controlled manner and without complications of a real-world dataset. This poor generation quality might arise from the fact that; (I) some factors of data might actually be at least partially dependent, so our simplifying assumption does not fully hold (II) the generator is usually a simple decoder and not capable of rendering complex patterns in output. To alleviate this problem, (Lee et al. 2020) proposed ID-GAN that feeds the latent space of VAE to a GAN (Goodfellow et al. 2014). The GAN module is employed in order to generate an output with high fidelity. This approach combines the strengths of the two modules; disentanglement representations form VAEs and high-fidelity synthesis of GANs. The disentangled factors acquired by the VAE module form the distilled information that will be the input to the GAN module.

A GAN module consists of a generator G and a discriminator D. The input to the generator is a noise variable z, and it aims to generate a fake sample from z that maximizes the probability of the discriminator to make a mistake in identifying the true sample from the fake sample. The objective function V is as follows:

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{2.2}$$

where x is a data point as before (Goodfellow et al. 2014). Note that, here is no constraint on z in this formulation. In InfoGAN (Chen et al. 2016), this noise vector was decomposed to two parts; (I) a noise vector z (II) a latent code c that aims to represent the salient semantic features of the data distribution. Essentially, in InfoGAN a regularization term was added to the objective function for maximizing the mutual information between the latent code c and the generator distribution $G(z, c)$ (Chen et al. 2016). Thus, the InfoGAN objective function is:

$$\min_{G} \max_{D} V(D, G) - \lambda I(c, G(z, c)) \tag{2.3}$$

The mutual information term $I(c, G(z, c))$ includes a posterior $p(c|x)$ (similar to VAEs), which cannot be optimized directly. Therefore, a lower bound can be calculated for this term by introducing an approximate posterior $q_\phi(c|x)$ for $p(c|x)$. Using this assumption, the lower bound is as follows:

$$\mathbb{E}_{z \sim p(z)}[D_{KL}(p(c)||q_\phi(c|G(z, c)))] \tag{2.4}$$

Note that in this formulation, the latent code c is computed as part of the whole GAN module which can degrade the disentanglement performance. In this project, I used the formulation developed in ID-GAN (Lee et al. 2020) that learns the latent code separately using β-VAE. Using this formulation, I can use the latent space of the β-VAE models I trained for the first phase. Note that in the ID-GAN formulation, the regularization term is as follows:

$$\beta R_{VAE}(q) + \lambda R_{ID}(G) \tag{2.5}$$

$$R_{VAE}(q) = \mathbb{E}_{x\sim p(x)}[D_{KL}(q_\phi(c|x)||p(c))] \tag{2.6}$$

$$R_{ID}(q) = \mathbb{E}_{s\sim p(s)}[D_{KL}(q_\phi(c)||q_\phi(c|G(z,c)))] \tag{2.7}$$

The architecture of the ID-GAN network is shown in Figure 2.1. In Step1, the β-VAE model is trained. The latent code of the trained β-VAE is concatenated with the input noise vector to the generator for training in Step2. The variable s in Figure 2.1 is written as z in the equations above.

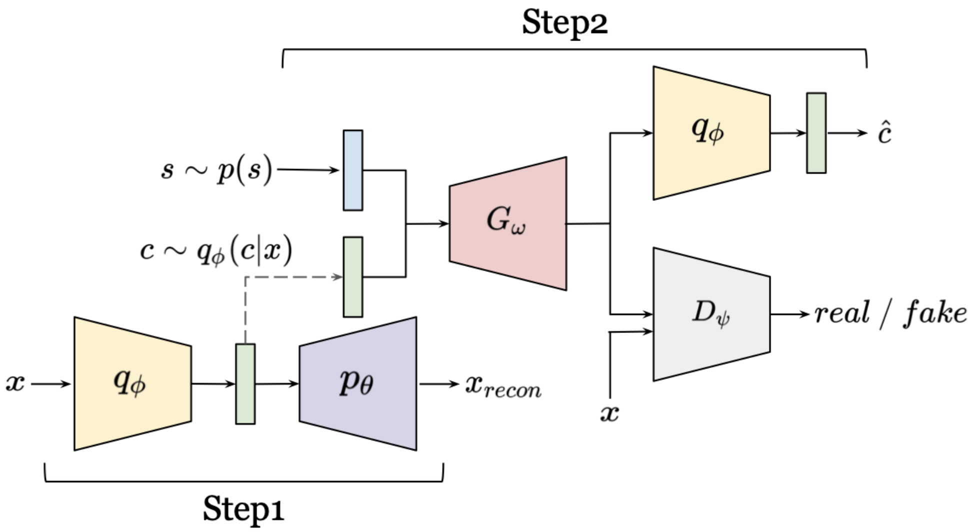

Figure 2.1: The ID-GAN network. The β-VAE model is trained in Step1. Then the latent space of the trained β-VAE model is concatenated with the noise vector as input to the generator, and the GAN module is trained. The variable s in this figure is written as z in the equations above. The schematic was taken from (Lee et al. 2020).

Chapter 3

Experiments

In this chapter, I explain the properties of the dSprite dataset. In addition, I describe the experiments that have been done and the obtained results. Note that the implementations have been done in PyTorch.

3.1 Dataset

For the experiments, dSprite dataset has been used. This synthetic dataset consist of $737,280$ binary 2D shapes. The dataset contains all combinations of 3 different shapes (oval, heart and square) with 4 other attributes: (i) 32 values for position X (ii) 32 values for position Y (iii) 6 values for scale (iv) 40 values for rotation. The images in this dataset are of 64×64 resolution. For each of these attributes we have equal number of labels as the number of distinct values. Using these labels a subset of the dataset can be selected.

3.2 Results

In the first phase of the project I trained several β-VAE models with different settings for the number of dimension of latent space, value of β, learning rate, and the number of epochs. Figure 3.1 shows the decoder/generation output of a frame for models trained with different settings. The numbers in the title of each output are latent dimension, β value, learning rate, and threshold for excluding some of the x-axis positions from training data, respectively. I chose the settings to be all of the combinations of $|z| \in \{3, 5, 10\}$, $\beta \in \{0.5, 5, 100\}$, learning rate $\in \{1\text{e}{-4}, 1\text{e}{-5}\}$, and position threshold $\in \{5, 16, 32\}$. A position threshold t means that only the samples with x-axis position label $\leq t$ will be considered in training. I used this threshold parameter since I wanted to exclude some of the positions from training, and see if the learned latent space can be generalized to produce a sample in an unseen position during training. This hypothesis can be evaluated by traversing the latent space in a systematic manner (I have done this for ID-GAN that I talked about later in this chapter). As can be seen in Figure 3.1, all of the models could capture the position of the object in the frame, i.e. the circle with high intensity, however the generation quality is far from the input frame. Note that, all of the models were trained for 100 epochs and the batch size for all of them was 256. It is worth mentioning that training the same experiments for 1000 epochs did not change the output, so I included the ones for 100 epochs here.

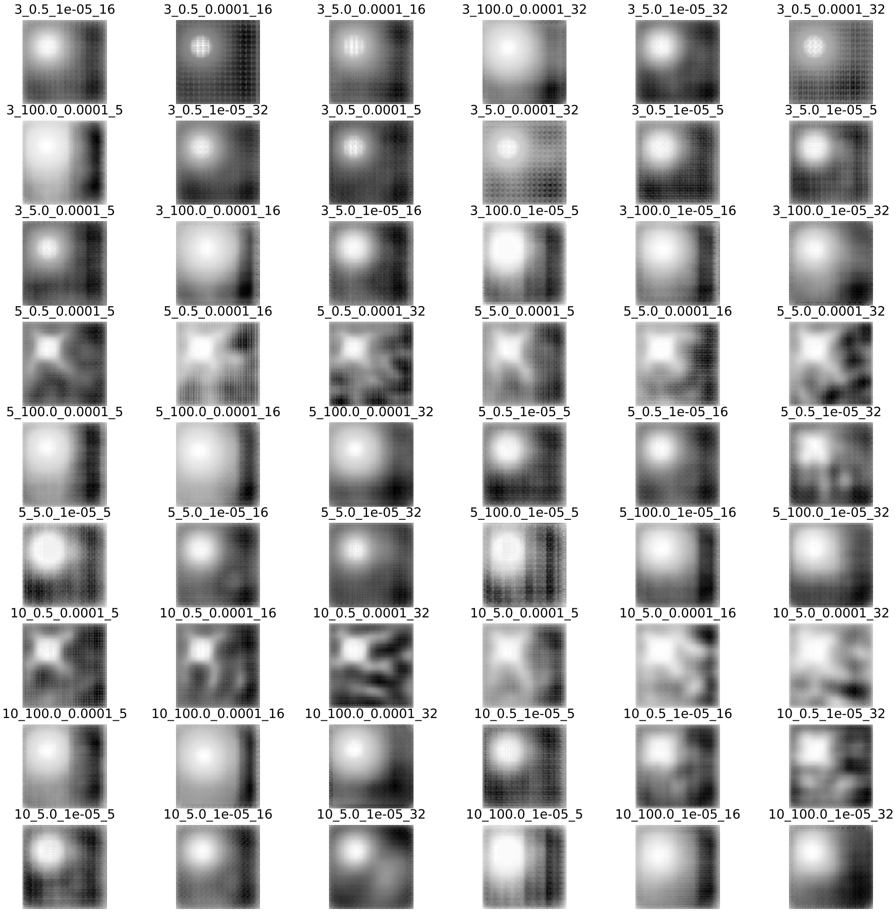

Figure 3.1: The reconstructed frames obtained by feeding a frame to a trained β-VAE with the setting in the title. The numbers in the title of each output are latent dimension, β value, learning rate, and threshold for excluding some of the x-axis positions from training data, respectively.

Notice that the generation quality in Figure 3.1 is better than others for some of the settings for example when $|z| = 3$, $\beta = 0.5$, learning rate (lr) is 0.0001 and the x-axis threshold (tr) is 16. However, still the shape boundaries are not sharp. To improve the generation quality of this model, I chose four settings from Figure 3.1 and used the latent code of their model as input c for Step2 of training ID-GAN. In the following I discuss the output of ID-GAN for each setting.

General note about all of the settings below. For all of the settings Step2 is trained for 100 epochs. In addition, one way for evaluating the disentanglement is by traversing the latent code and evaluate the output qualitatively (Kim & Mnih 2018). For example traversing one of the latent dimensions in a range and keeping the rest fixed might change the position of the object in different frames. This is a sign that this dimension is representing the position of the object. Furthermore, in all of the settings the generation quality is much higher than β-VAE alone, i.e. the boundaries of shapes are sharp, and the background is more clear.

The first setting. $|z| = 3$, $\beta = 0.5$, $lr = 0.0001$, $tr = 16$. I traversed the latent code in the range $[-2, 2]$ with steps 0.5. That means I tried all the combinations of nine numbers for different dimensions (the combinations are generated using three nested for loop). Figure 3.2 shows the output for these latent code values. There are 27 columns in this figure which means for the outputs in each row while two of the dimensions are fixed, the other one can take three consecutive values. Looking at the first column of this figure, it seems the first latent dimension is controlling the vertical position of the shape (this dimension is changing only vertically). Although, following the rows we see that the position, shape, scale, and rotation of the shapes are changing periodically and that can be a sign that not just one dimension is controlling one properties. Overall, looking at these output, different patterns can be discovered; for example looking at the few last rows I do not see any small scale shapes which can indicate that a combination of larger values for latent code can prevent from generating small shapes. Also, some shapes are generated at the second half of x-axis that the β-VAE network did not see during training.

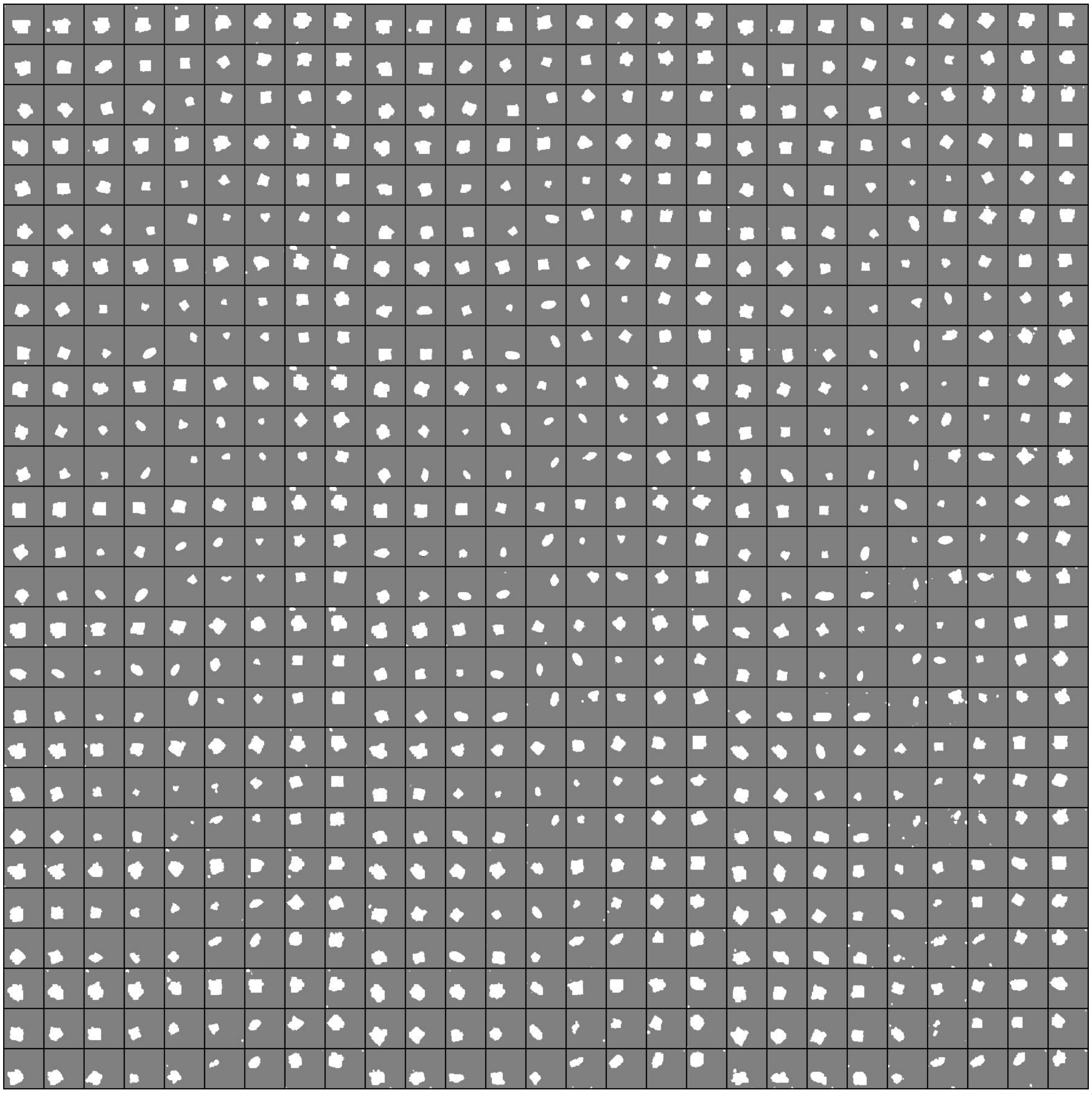

Figure 3.2: ID-GAN output by traversing the latent code in the range $[-2, 2]$. The settings for β-VAE: $|z| = 3$, $\beta = 0.5$, $lr = 0.0001$, $tr = 16$.

The second setting. $|z| = 3$, $\beta = 5.0$, $lr = 0.0001$, $tr = 16$. In this setting, the β value was higher than the first setting (also > 1) with everything else unchanged. I see more disentanglement at least for scale. However, higher value of β degraded the generation quality. Similar to the first setting, it seems still a combination of latent dimensions can change the properties of generated output rather than one dimension.

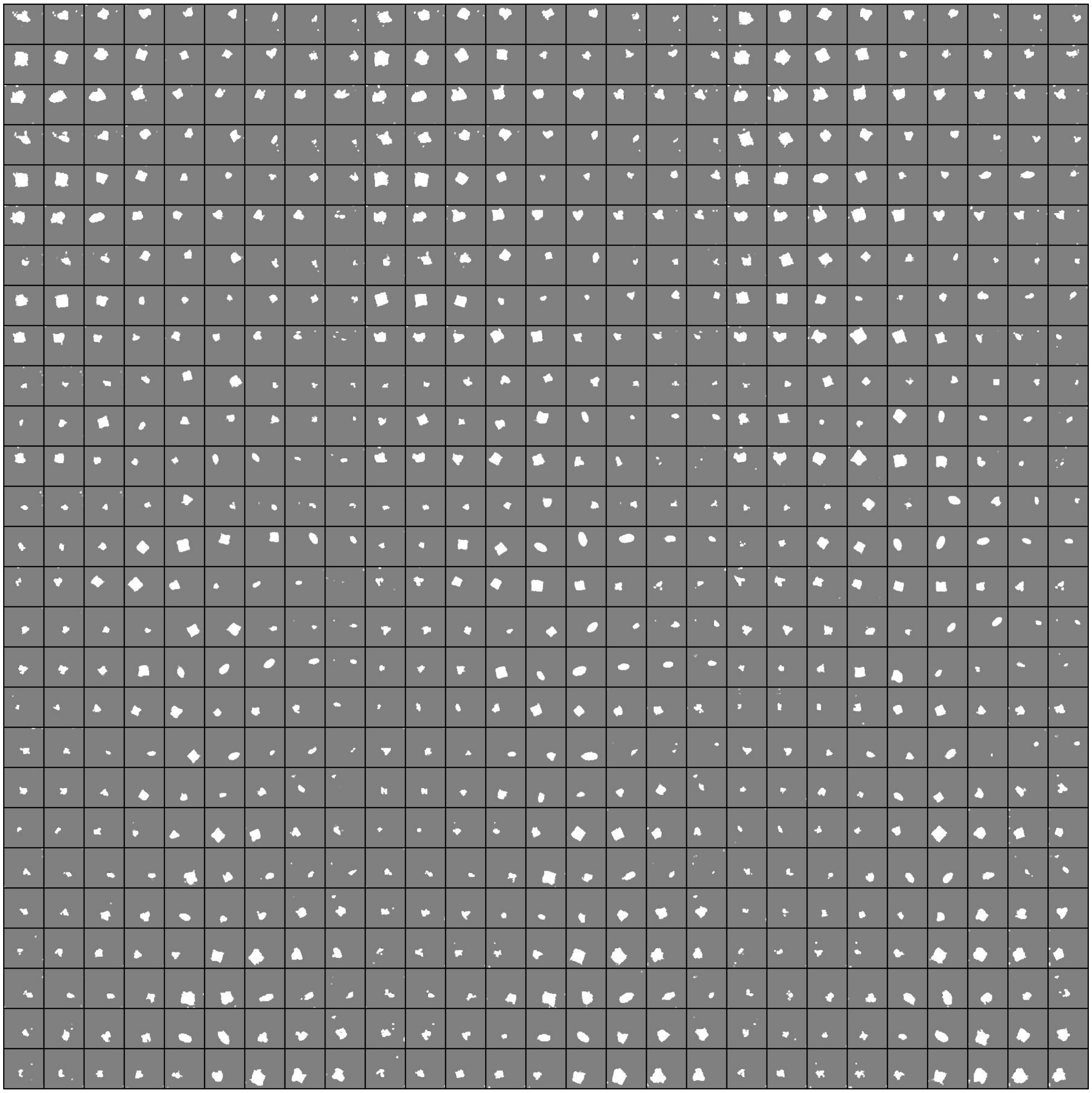

Figure 3.3: ID-GAN output by traversing the latent code in the range $[-2, 2]$. The settings for β-VAE: $|z| = 3$, $\beta = 5.0$, $lr = 0.0001$, $tr = 16$.

The thrid setting. $|z| = 5$, $\beta = 0.5$, $lr = 0.0001$, $tr = 16$. I traversed the latent code in the range $[-2, 2)$ with steps 1. So, then latent code can take any combination of 5 numbers. In this setting, I see that the scale is changing less than previous settings that can be a sign of higher disentanglement. Also, it seems the y-axis position of shapes is only changing on certain values for a specific dimension (the shapes were generated at the top of the frame and periodically their position were changed to the bottom of the frame).

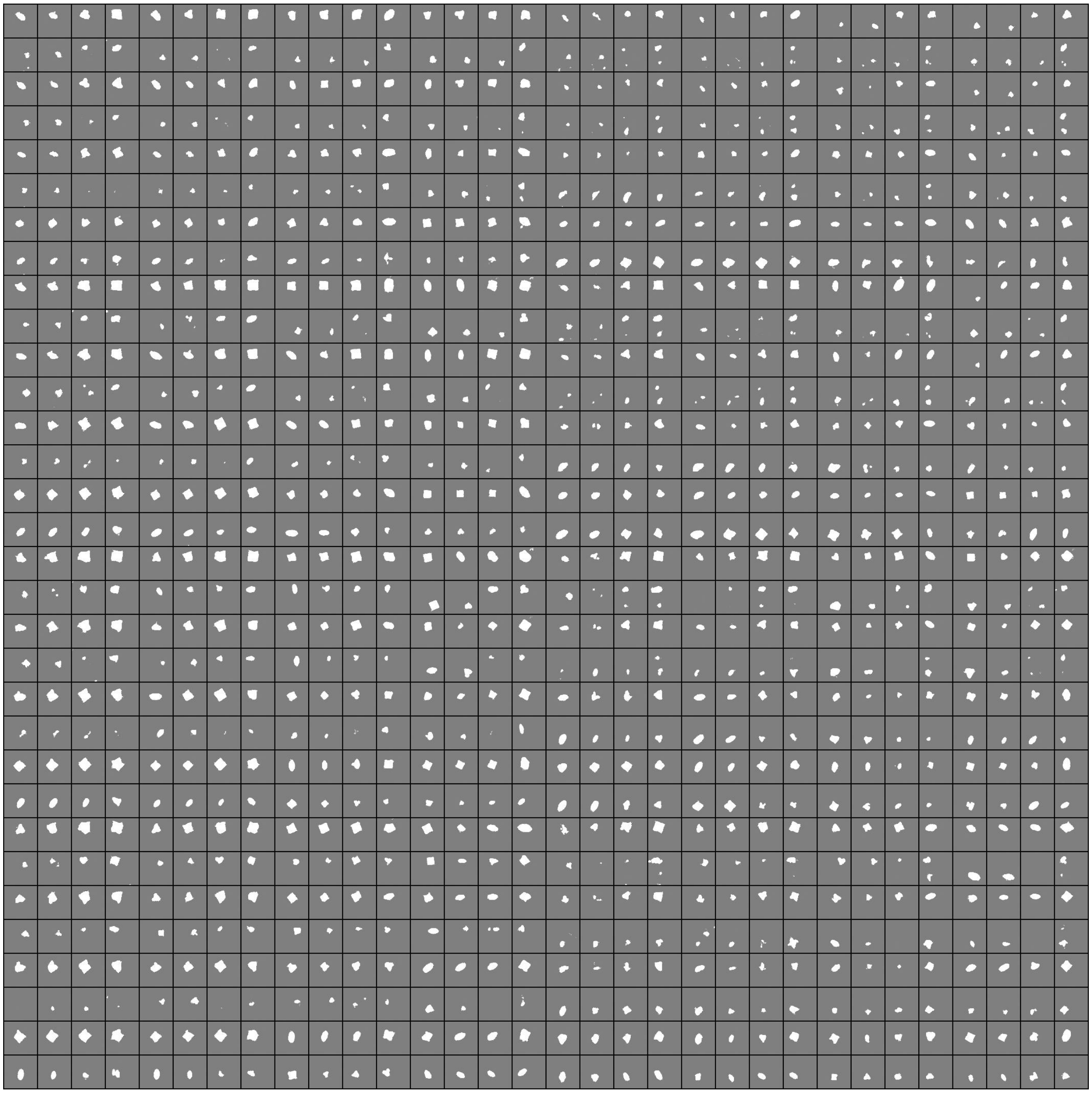

Figure 3.4: ID-GAN output by traversing the latent code in the range $[-2, 2)$. The settings for β-VAE: $|z| = 5$, $\beta = 0.5$, $lr = 0.0001$, $tr = 16$.

The fourth setting. $|z| = 5$, $\beta = 5.0$, $lr = 0.0001$, $tr = 16$. Similar to the previous setting, I see scales are limited in Figure 3.5 which can be a sign of disentanglement. Also, the generation quality is higher in this case. Also, the shapes and y-axis positions are periodically changing.

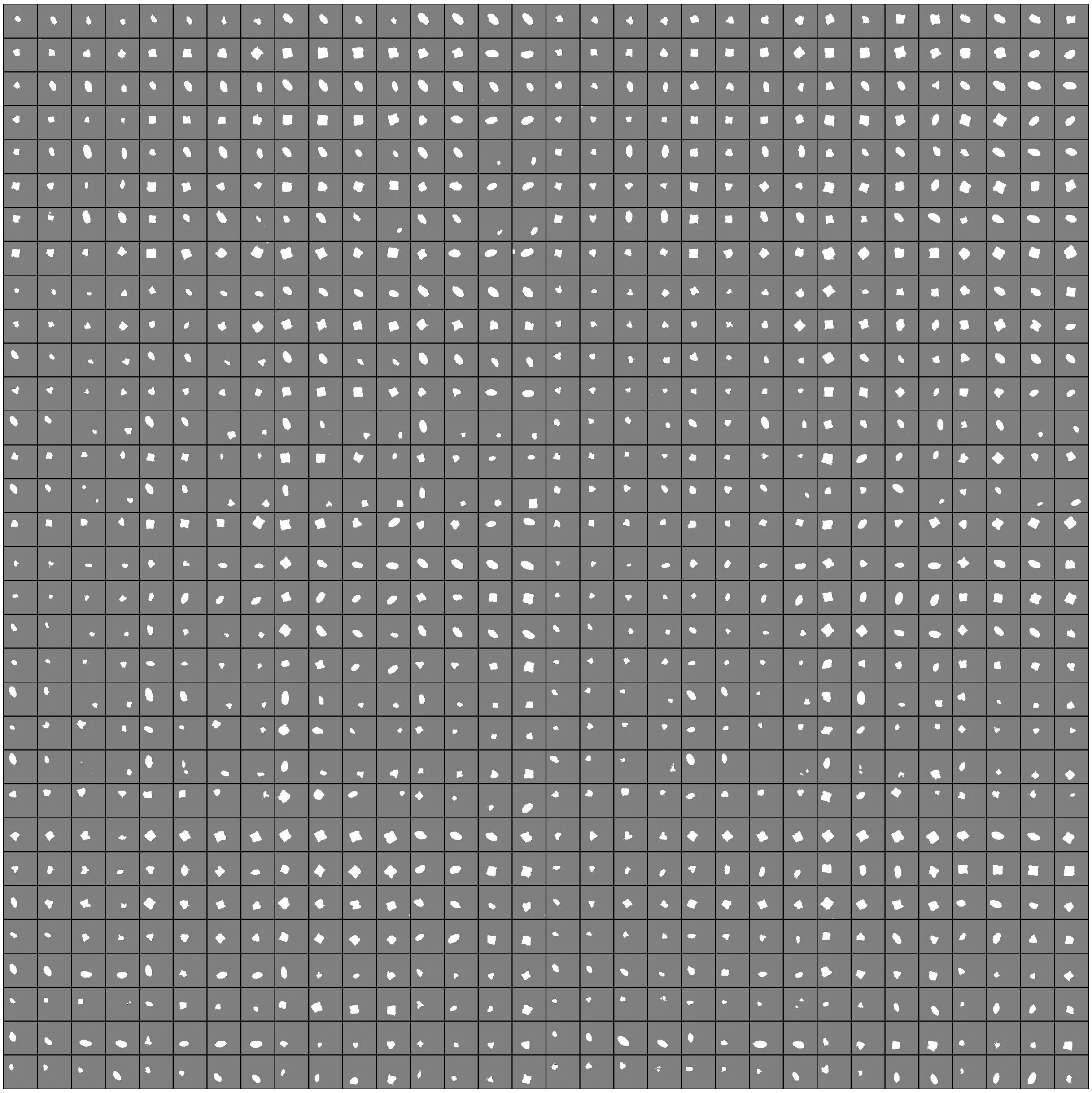

Figure 3.5: ID-GAN output by traversing the latent code in the range $[-2, 2)$. The settings for β-VAE: $|z| = 5$, $\beta = 5.0$, $lr = 0.0001$, $tr = 16$.

Chapter 4

Discussion and Conclusion

4.1 Discussion

In this paper I explored the disentanglement and generation performance of β-VAEs. At this stage, I observed that generation quality is not as good that I can even investigate the disentanglement performance. So, I used an architecture called ID-GAN to improve the generation quality. The output from ID-GAN has a much higher generation quality, also I observed some degrees of disentanglement. I evaluated the performance using latent code traversal which can be subjective. There are some metrics developed for quantifying both generation quality and disentanglement that I mentioned some of them in Chapter 1. A caveat about these metrics is that the ground truth disentangled representation of the dataset is needed for being able to calculate them.

4.2 Conclusion

I evaluated the performance of β-VAEs for disentanglement and generation. Furthermore, I looked for ways to improve the β-VAE performance which led me to some works that discuss the correspondence of β-VAE with PCA (Zietlow et al. 2021), how to alleviate the generation output of VAEs (Lee et al. 2020), also, is β-VAE somehow prioritizing in retaining information (or any information is equally good) (Fertig et al. 2018). I also realized it is possible to improve the generation quality of β-VAEs by using a GAN module.

Quantifying the disentanglement and understanding the number of required dimensions for encoding a feature could be a task of importance as a topic of interest for future research. Although, using a GAN helped with having a closer image reconstruction to the input, that might have an effect on obtaining disentangled features. In this regard, simplifying the network for the main task could result in a precise answer.

Bibliography

Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G. & Lerchner, A. (2018), 'Understanding disentangling in β-vae', *arXiv preprint arXiv:1804.03599* .

Chen, R. T., Li, X., Grosse, R. & Duvenaud, D. (2018), 'Isolating sources of disentanglement in variational autoencoders', *arXiv preprint arXiv:1802.04942* .

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I. & Abbeel, P. (2016), 'Infogan: Interpretable representation learning by information maximizing generative adversarial nets', *arXiv preprint arXiv:1606.03657* .

Fertig, E., Arbabi, A. & Alemi, A. A. (2018), 'β-vaes can retain label information even at high compression', *arXiv preprint arXiv:1812.02682* .

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. (2014), 'Generative adversarial networks', *arXiv preprint arXiv:1406.2661* .

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B. & Hochreiter, S. (2017), 'Gans trained by a two time-scale update rule converge to a local nash equilibrium', *arXiv preprint arXiv:1706.08500* .

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S. & Lerchner, A. (2016), 'beta-vae: Learning basic visual concepts with a constrained variational framework'.

Kim, H. & Mnih, A. (2018), Disentangling by factorising, *in* 'International Conference on Machine Learning', PMLR, pp. 2649–2658.

Kingma, D. P. & Welling, M. (2013), 'Auto-encoding variational bayes', *arXiv preprint arXiv:1312.6114* .

Kulkarni, T. D., Whitney, W., Kohli, P. & Tenenbaum, J. B. (2015), 'Deep convolutional inverse graphics network', *arXiv preprint arXiv:1503.03167* .

Lee, W., Kim, D., Hong, S. & Lee, H. (2020), High-fidelity synthesis with disentangled representation, *in* 'European Conference on Computer Vision', Springer, pp. 157–174.

Matthey, L., Higgins, I., Hassabis, D. & Lerchner, A. (2017), 'dsprites: Disentanglement testing sprites dataset', https://github.com/deepmind/dsprites-dataset/.

Wan, Z., Zhang, B., Chen, D., Zhang, P., Chen, D., Liao, J. & Wen, F. (2020), 'Old photo restoration via deep latent space translation', *arXiv preprint arXiv:2009.07047* .

Zietlow, D., Rolinek, M. & Martius, G. (2021), 'Demystifying inductive biases for β-vae based architectures', *arXiv preprint arXiv:2102.06822* .

Mohammad Haghir Ebrahimabadi
mhaghir@iu.edu | (404) 953 - 2708 | Linkedin | GitHub | Google Scholar

EDUCATION

Indiana University, Bloomington, IN, United States July 2022
Master of Science in Computer Science GPA: 3.90 / 4.0

Sharif University of Technology, Tehran, Iran December 2017
Master of Science in Bioelectrics GPA: 3.74 / 4.0

Amirkabir University of Technology, Tehran, Iran December 2015
Bachelor of Science in Electronics GPA: 3.82 / 4.0

Amirkabir University of Technology, Tehran, Iran August 2015
Bachelor of Science in Bioelectrics GPA: 3.83 / 4.0

TECHNICAL SKILLS

Languages: Python, MATLAB, C/C++, BASH, Java, LaTeX, VHDL, R, HTML
Platforms: Linux, Mac OS, Microsoft Windows
Databases: SQL
Devices: STM32F407VG, STM32F103C8T6, LPC2148, L3G4200D, ATmega32, ATmega8, XC3S400
Miscellaneous: Google Workspace, Adobe Photoshop, Adobe After Effects, Altium Designer
Spoken Languages: English, Farsi

WORK EXPERIENCE

TReNDS Center, Atlanta, GA 2023 - 2023
Research Engineer

- Worked on uncovering brain networks using linear and explicitly nonlinear functional interaction.

Indiana University, Bloomington, IN
Research Assistant in Computer Vision Lab at IU, and Computational Biology Lab at IU 2018 - 2022

- Understood a problem and broke it down into fundamental questions for providing a solution.
- Explained a problem, ideas, and solutions.
- Practiced effective coding in Python, PyTorch, TensorFlow, and terminal.

Teaching Assistant for Applied Machine Learning, Machine Learning for Bioinformatics, Data Mining 2020 - 2022

- Presented science materials.
- Communicated with more than 100 students and answered their questions about homeworks and quizzes.

National Institute of Health (NIH), Washington, D.C.
Predoctoral Visiting Fellow at National Cancer Institute (NCI) 2019 - 2020

- Became familiar with the work environment.
- Worked on driving tumor phylogeny trees using deep learning.

PROJECTS

Indiana University, Bloomington, IN
Implementing RANSAC and Hough Transform Algorithms for Hyperplane Detecting Summer 2021

- Given a set of 3D points, the mentioned algorithms can detect an existing hyperplane. Implemented a code to synthesize a hyperplane with adjustable noise and the mentioned algorithms for detecting the hyperplane.

Generating Pseudo-LiDAR Data from a Single Satellite Image Spring 2021

- Employed Transformers for obtaining depth maps from a single image and producing point cloud data by mapping each pixel in the depth map to a 3D location.

Implementing Totally Ordered Multicasting Algorithm Spring 2019

- Used the ZeroMQ package to implement a publisher-subscriber pattern where subscribers as clients can multicast their messages to other clients using a publisher. In addition, each client can send messages to other clients privately.

Sharif University of Technology, Tehran, Iran

Implementing a Registration Algorithm Summer 2016

- Implemented a registration algorithm for aligning two images and added to the Medical Image Registration Toolbox (MIRT) package in MATLAB.

Amirkabir University of Technology, Tehran, Iran

Designing and Implementing a Wireless Response Box Summer 2015

- Used STM32F103C8T6 microcontroller to run an infrared LED for message passing between a sender and a receiver. The messages are coded for sending and decoded at the receiver and shown using a GUI on a computer. MATLAB was used for writing the GUI.

Saving Data on a µSD Card Summer 2014

- Programmed STM32F407VG microcontroller for saving data with FAT32 format on a µSD card using SDIO interface.

Reading Gyroscope Data Spring 2015

- Programmed LPC2148 microcontroller for reading data from L3G4200D IC (gyroscope) using I2C interface and displaying angles on an LCD.

Programming and Simulating a Simple Processor using VHDL Language Fall 2014

- Utilized Altium Designer to design and implement a simple processor and run it on XC3S400 AMD Xilinx FPGA.

PUBLICATIONS

Ebrahimabadi MH**. Disentangled Representation Learning Using (β-)VAE and GAN. arXiv preprint arXiv:2208.04549. 2022 Aug 9. **Citations: 0

*Malikić S, Mehrabadi FR, Azer ES, **Ebrahimabadi MH**, Sahinalp SC. Studying the history of tumor evolution from single-cell sequencing data by exploring the space of binary matrices. Journal of Computational Biology. 2021 Sep 1;28(9):857-79. **Citations: 5***

*Qi YA, Maity TK, Cultraro CM, Misra V, Zhang X, Ade C, Gao S, Milewski D, Nguyen KD, **Ebrahimabadi MH**, Hanada KI. Proteogenomic Analysis Unveils the HLA Class I-Presented Immunopeptidome in Melanoma and EGFR-Mutant Lung Adenocarcinoma. Molecular & Cellular Proteomics. 2021 Jan 1;20. **Citations: 4***

*Azer ES, **Ebrahimabadi MH**, Malikić S, Khardon R, Sahinalp SC. Tumor phylogeny topology inference via deep learning. Iscience. 2020 Nov 20;23(11):101655. **Citations: 12***

*Eybposh MH, **Ebrahim-Abadi MH**, Jalilpour-Monesi M, Saboksayr SS. Segmentation and classification of cine-mr images using fully convolutional networks and handcrafted features. arXiv preprint arXiv:1709.02565. 2017 Sep 8. **Citations: 4***

Ebrahim-Abadi MH**, Fatemizadeh E. A Clustering-Based Algorithm for De Novo Motif Discovery in DNA Sequences. In 2017 24th National and 2nd International Iranian Conference on Biomedical Engineering (ICBME) (pp. 1-6). IEEE. **Citations: 0