Student Checklist (1A)

This form is required for ALL projects.

1. a. Student/Team Leader: Eshaan Barkataki Grade: 10

   Email: esbarkataki@ctemc.org Phone: +1 (732) 487-0744

   b. Team Member: ______ c. Team Member: ______

2. Title of Project: VNN: A Variable Neural Network for Varied-Domain Problems.

3. School: High Technology High School School Phone: +1 (732) 842-8444

   School Address: 765 Newman Springs Rd Lincroft, NJ

4. Adult Sponsor: Craig Queenan Phone/Email: cqueenan@ctemc.org

5. Does this project need SRC/IRB/IACUC or other pre-approval? ☐ Yes (Tentative start date: ______) ☒ No

6. Is this a continuation/progression from a previous year? ☐ Yes ☒ No

   If Yes:

   a. Attach the previous year's ☐ Abstract **and** ☐ Research Plan/Project Summary

   b. Explain how this project is new and different from previous years on

   ☐ Continuation/Research Progression Form (7)

7. This year's experimentation/data collection:

   07/09/22 — Actual Start Date: (mm/dd/yy)

   01/08/23 — End Date: (mm/dd/yy)

8. Where will you conduct your experimentation? (check all that apply)

   ☐ Research Institution ☑ School ☐ Field ☑ Home ☐ Other: ______

9. Source of Data:

   ☑ Collected self/mentor ☐ Other Describe/url: ______

10. List the name and address of all non-home and non-school work site(s), whether you worked there virtually or on-site:

    Name ______ ______

    Address: ______ ______

    ______ ______

    Phone/email ______ ______

11. **Complete a Research Plan/Project Summary following the Research Plan/Project Summary instructions must accompany this form.**

12. **An abstract is required for all projects after experimentation.**

Research Plan/Project Summary Instructions

A complete Research Plan/Project Summary is required for ALL projects and must accompany Student Checklist (1A).

- All projects must have a Research Plan/Project Summary
  a. The Research Plan is to be written prior to experimentation following the instructions below to detail the rationale, research question(s), methodology, and risk assessment of the proposed research.
  b. If changes are made during the research, such changes can be added to the original research plan as an addendum, recognizing that some changes may require returning to the IRB or SRC for appropriate review and approvals. If no additional approvals are required, this addendum serves as a project summary to explain research that was conducted.
  c. If no changes are made from the original research plan, no project summary is required.
  d. Some studies, such as an engineering design or mathematics projects, will be less detailed in the initial project plan and will change through the course of research. If such changes occur, a project summary that explains what was done is required and can be appended to the original research plan.
- The Research Plan/Project Summary should include the following:
  a. **RATIONALE:** Include a brief synopsis of the background that supports your research problem and explain why this research is important and if applicable, explain any societal impact of your research.
  b. **RESEARCH QUESTION(S), HYPOTHESIS(ES), ENGINEERING GOAL(S), EXPECTED OUTCOMES:** How is this based on the rationale described above?
  c. Describe the following in detail:
     - **Procedures:** Detail all procedures and experimental design including methods for data collection, and when applicable, the source of data used. Describe only your project. Do not include work done by mentor or others. If you will use published surveys, questionnaires or tests, describe how you obtained these, including required permission if applicable.
     - **Risk and Safety:** Identify any potential risks and safety precautions needed.
     - **Data Analysis:** Describe the procedures you will use to analyze the data/results.
  d. **BIBLIOGRAPHY:** List major references (e.g. science journal articles, books, internet sites) from your literature review. If you plan to use vertebrate animals, one of these references must be an animal care reference.

Items 1–4 below are subject-specific guidelines for additional items to be included in your research plan/project summary as applicable.

1. Human participants research:

a. **Participants:** Describe age range, gender, racial/ethnic composition of participants. Identify vulnerable populations (minors, pregnant women, prisoners, mentally disabled or economically disadvantaged).
b. **Recruitment:** Where will you find your participants? How will they be invited to participate?
c. **Methods:** What will participants be asked to do? Will you use any surveys, questionnaires or tests? If yes and not your own, how did you obtain? Did it require permissions? If so, explain. What is the frequency and length of time involved for each subject?
d. **Risk Assessment:** What are the risks or potential discomforts (physical, psychological, time involved, social, legal, etc.) to participants? How will you minimize risks? List any benefits to society or participants.
e. **Protection of Privacy:** Will identifiable information (e.g., names, telephone numbers, birth dates, email addresses) be collected? Will data be confidential/anonymous? If anonymous, describe how the data will be collected. If not anonymous, what procedures are in place for safeguarding confidentiality? Where will data be stored? Who will have access to the data? What will you do with the data after the study?
f. **Informed Consent Process:** Describe how you will inform participants about the purpose of the study, what they will be asked to do, that their participation is voluntary and they have the right to stop at any time.

2. Vertebrate animal research:

a. Discuss potential ALTERNATIVES to vertebrate animal use and present justification for use of vertebrates.
b. Explain potential impact or contribution of this research.
c. Detail all procedures to be used, including methods used to minimize potential discomfort, distress, pain and injury to the animals and detailed chemical concentrations and drug dosages.
d. Detail animal numbers, species, strain, sex, age, source, etc., include justification of the numbers planned.
e. Describe housing and oversight of daily care.
f. Discuss disposition of the animals at the end of the study.

- **Potentially hazardous biological agents research:**
  a. Give source of the organism and describe BSL assessment process and BSL determination.
  b. Detail safety precautions and discuss methods of disposal.

4. Hazardous chemicals, activities & devices:

a. Describe Risk Assessment process, supervision, safety precautions and methods of disposal.
b. Material Safety Data Sheets are not necessary to submit with paperwork.

Approval Form (1B)

A completed form is required for each student, including all team members.

1. To Be Completed by Student and Parent

a. **Student Acknowledgment:**
- I understand the risks and possible dangers to me of the proposed research plan.
- I have read the ISEF Rules and Guidelines and will adhere to all International Rules when conducting this research.
- I have read and will abide by the science fair ethics statement.

Student researchers are expected to maintain the highest standards of honesty and integrity. Scientific fraud and misconduct are not condoned at any level of research or competition. Such practices include but are not limited to plagiarism, forgery, use or presentation of other researcher's work as one's own, and fabrication of data. Fraudulent projects will fail to qualify for competition in affiliated fairs and ISEF.

Eshaan Barkataki	Eshaan B	07/09/22
Student's Printed Name	Signature	Date Acknowledged (mm/dd/yy) (Must be prior to experimentation.)

b. **Parent/Guardian Approval:** I have read and understand the risks and possible dangers involved in the **Research Plan/Project Summary**. I consent to my child participating in this research.

Sanjeev Barkataki	Sanjeev	07/09/22
Parent/Guardian's Printed Name	Signature	Date Acknowledged (mm/dd/yy) (Must be prior to experimentation.)

2. To be completed by the local or affiliated Fair SRC

(Required for projects requiring prior SRC/IRB APPROVAL. Sign 2a or 2b as appropriate.)

a. **Required for projects that need prior SRC/IRB approval BEFORE experimentation** (humans, vertebrates or potentially hazardous biological agents).

The SRC/IRB has carefully studied this project's **Research Plan/ Project Summary** and all the required forms are included. My signature indicates approval of the **Research Plan/Project Summary** before the student begins experimentation.

SRC/IRB Chair's Printed Name

Signature

Date of Approval (mm/dd/yy) (Must be prior to experimentation.)

OR

b. **Required for research conducted at all Regulated Research Institutions with no prior fair SRC/IRB approval**.

This project was conducted at a regulated research institution (**not home or high school, etc.**), was reviewed and approved by the proper institutional board before experimentation and complies with the ISEF Rules. **Attach (1C) and any required institutional approvals (e.g. IACUC, IRB).**

SRC Chair's Printed Name

Signature

Date of Signature (mm/dd/yy) (May be after experimentation)

3. Final ISEF Affiliated Fair SRC Approval (Required for ALL Projects)

SRC Approval After Experimentation and Before Competition at Regional/State/National Fair

I certify that this project adheres to the approved **Research Plan/Project Summary** and complies with all ISEF Rules.

Regional SRC Chair's Printed Name	Signature	Date of Approval (mm/dd/yy)
State/National SRC Chair's Printed Name *(where applicable)*	Signature	Date of Approval (mm/dd/yy)

Checklist for Adult Sponsor (1)

This completed form is required for ALL projects.

To be completed by the Adult Sponsor in collaboration with the student researcher(s):

Student's Name(s): Eshaan Barkataki

Project Title: VNN: A Variable Neural Network for Varied-Domain Problems.

1. ☐ I have reviewed the ISEF Rules and Guidelines, including the science fair ethics statement.
2. ☐ I have reviewed the student's completed Student Checklist (1A) and Research Plan/Project Summary.
3. ☐ I have worked with the student and we have discussed the possible risks involved in the project.
4. ☐ The project involves one or more of the following and requires prior approval by an SRC, IRB, IACUC or IBC:
   - ☐ Humans
   - ☐ Vertebrate Animals
   - ☐ Potentially Hazardous Biological Agents
     - ☐ Microorganisms ☐ rDNA ☐ Tissues
5. ☐ Items to be completed for **ALL PROJECTS**
   - ☐ Adult Sponsor Checklist (1)
   - ☐ Student Checklist (1A)
   - ☐ Research Plan/Project Summary
   - ☐ Approval Form (1B)
     - ☐ Regulated Research Institutional/Industrial Setting Form (1C) (when applicable; after completed experiment)
     - ☐ Continuation/Research Progression Form (7) (when applicable)

Additional forms required if the project includes the use of one or more of the following (check all that apply):

☐ **Humans,** including student designed inventions/prototypes. (Requires prior approval by an Institutional Review Board (IRB); see full text of the rules.)
- ☐ Human Participants Form (4) or appropriate Institutional IRB documentation
- ☐ Sample of Informed Consent Form (when applicable and/or required by the IRB)
- ☐ Qualified Scientist Form (2) (when applicable and/or required by the IRB)

☐ **Vertebrate Animals** (Requires prior approval, see full text of the rules.)
- ☐ Vertebrate Animal Form (5A) - for projects conducted in a school/home/field research site (SRC prior approval required
- ☐ Vertebrate Animal Form (5B) - for projects conducted at a Regulated Research Institution. (Institutional Animal Care and Use Committee (IACUC) approval required prior experimentation.)
- ☐ Qualified Scientist Form (2) (Required for all vertebrate animal projects at a regulated research site or when applicable)

☐ **Potentially Hazardous Biological Agents** (Requires prior approval by SRC, IACUC or IBC, see full text of the rules.)
- ☐ Potentially Hazardous Biological Agents Risk Assessment Form (6A)
- ☐ Human and Vertebrate Animal Tissue Form (6B) - to be completed in addition to Form 6A when project involves the use of fresh or frozen tissue, primary cell cultures, blood, blood products and body fluids.
- ☐ Qualified Scientist Form (2) (when applicable)
- ☐ The following are exempt from prior review but require a Risk Assessment Form 3: projects involving protists, archae and similar microorganisms; projects using manure for composting, fuel production or other non-culturing experiments; projects using color change coliform water test kits, microbial fuel cells; and projects involving decomposing vertebrate organisms.

☐ **Hazardous Chemicals, Activities and Devices** (No SRC prior approval required, see full text of the rules.)
- ☐ Risk Assessment Form (3)
- ☐ Qualified Scientist Form (2) (required for projects involving DEA-controlled substances or when applicable)

☐ **Other**
- ☐ Risk Assessment Form (3)

☐ **I attest to the information checked above and that I have read and agree to abide by the science fair ethics statement.**

Adult Sponsor's Printed Name	Signature	Date of Review (mm/dd/yy)
Phone	Email	

VNN: A Variable Neural Network for Varied-Domain Problems.

Eshaan Barkataki
Mr. Queenan
High Technology High School

1 Introduction

A multitude of neural networks has been developed to support handling variable-sized sequences. For example, a recent development has been a Transformer (Vaswani et al.), that has proven to be successful in Natural Language tasks. An alternative option is the Long Short-Term Memory (Hochreiter and Schmidhuber). Inspired by the original recurrent neural network, it was proven to be successful in Natural Language tasks as well.

Recurrent Neural Networks work by processing every element in the sequence one by one. First, the hidden state of the recurrent neural network is initialized (usually by setting all values to 0). Then, the first input element (timestep 0) in the sequence will be processed. It does this by first multiplying the input element at timestep 0 by a weight matrix and a bias matrix. Furthermore, the hidden state will be processed by multiplying the hidden state by a separate weight matrix and bias matrix. The hidden state for the next time step will be computed by applying an activation function over the sum of the processed input and the processed hidden state. Then, the next input element of the sequence (timestep 1) will be processed, using the previous hidden state as input and using the same weight matrices (the weights and biases will be shared across time steps). However, there are improvements to this neural network. First, the recurrent neural network cannot remember information well. For example, if information was delivered to the recurrent neural network at time step 1, then the neural network would not remember the same information at timestep 100. Furthermore, when training the neural network, it suffers from the vanishing gradient problem. This is due to the fact that neural networks learn by using the Chain Rule, which involves multiplying numbers less than or equal to one when applied to a neural network. If the recurrent neural network is trained across multiple layers, the gradient will exponentially decrease to 0. Thus, training the neural network will no longer be effective.

LSTMs work to solve the vanishing gradient problem that recurrent neural networks have, as well as make improvements on remembering past information. First, the LSTM is initialized with a hidden state and a cell state. The hidden state and input at timestep 0 will work together to form the forget gate. The forget gate outputs a number between 0 to 1 that signifies whether each value of the cell gate is useful or not (0 means the value is not useful, and 1 signifies that the value is useful). The output of the forget gate is multiplied by the cell state using element-wise multiplication. The output of the forget gate is calculated by concatenating the hidden state and input and passing it through a linear layer (multiplying input with weights and adding bias) with a sigmoid activation function.

In order to update the cell state, we use the input gate layer. The output of the input gate layer is then added to the cell state. Calculating the input gate layer involves concatenating the hidden state and input at timestep 0 as input. Then the input is going to go through two separate layers (one will have Sigmoid activation function, the other will have Tanh activation function), and finally multiplying the output of those two layers. The producto will be the output of the input gate layer, which will be added to the cell state.

Next, the new hidden state and output is produced using the output gate. First, the output gate involves concatenating the hidden state and input at timestep 0 as input to an output layer with a sigmoid activation function. Then the new hidden state is computed by multiplying the output of the output layer and the Tanh activation function of the cell state. The new hidden state will act as the output as well. This will repeat for the next timesteps, using the new cell state and new hidden state as input to the next hidden state and cell state. The input will be the next element in sequence.

Another alternative to LSTMs are Transformers (Vaswani et al.). Unlike LSTMs, where each element is processed separately, Transformers are involved in taking every element in the sequence and multiplying by each other element, thus making Transformers much faster but more memory intensive. Transformers use multi-head attention blocks in order to process variable sized sequences. The multi-head attention block takes in 3 inputs, the values, keys, and queries. These 3 inputs are usually vectors that are the same (in some cases, sometimes the queries can differ, depending on the type of Transformer architecture). The Transformer then applies a separate linear layer for every value, key, and query for every head. Then the output from each of these linear layers for each head are the inputs to the scaled dot-product attention block. The scaled -dot product attention can be summarized in this formula:

$$Attention(Q, K, V) \;=\; softmax(\frac{QK^T}{\sqrt{d_k}}M)V$$

Where Q represents the queries, K^T represents the transpose of the keys, d_k represents the query and keys dimension, which is calculated by dividing $\frac{d_{model}}{heads}$ (d_{model} is a hyperparameter adjusted, and $heads$ is the number of heads that multi-head attention has). V represents the values and M stands for the masking. Note that masking is optional, but is used when we want to ensure that the neural network is only comparing with the elements that it has seen (it cannot compare with future elements that it has not seen). Usually the masking is an upper triangular matrix consisting of either 1s or 0s, where 0 represents to not compare the value, and 1 represents to compare the value.

Then, the output of the Scale Dot-Product Attention for every head is calculated and concatenated. The multi-head attention block is used with every Transformer architecture, with additional layers and residual connections embedded into the Transformer model architecture.

This research aims to create a new type of neural network called VNN, a **V**ariable-Sized **N**eural **N**etwork. This is a neural network that generates weights using another neural network. This makes it so it could calculate new weights for variable-sized input sequences. Furthermore, given the desired output size, the same neural network weight generation can output variable-sized sequences as well. VNN will be tested on a variety of benchmarks to evaluate its capabilities and limitations. VNN will be first tested with Image Classification, Sentimental Analysis, Autoencoding, and Robot Design and Control with Reinforcement Learning.

For the experiment, the independent variable is the type of benchmark that the neural network is in (Image Classification, Sentimental Analysis, Autoencoding, or Robot Design and Control) and the type of neural network: VNN, Transformers, or LSTMs. The Transformer and LSTM will serve as the control, while the VNN will serve as the independent variable. The dependent variable depends on the environment. For Image Classification and Sentimental Analysis, the dependent variable will be the test validation accuracy, and the time needed to train for 5 epochs (this measures the speed of the neural network during training). For the Robot Design and Control Reinforcement Learning benchmark, the dependent variable will be the average reward over the last 100 timesteps and the time needed to reach 10,000,000 timesteps. For the Reinforcement Learning benchmark, training an LSTM and VNN will not be used but instead compare the results from the EvoGym paper (Bhatia et al.)

VNN performs with higher accuracy and less time than the Transformer and the LSTM on all of the benchmarks because VNN takes all the elements of the sequence at once (unlike LSTMs) as well as being less resource intensive as Transformers.

2 Materials

The hardware needed for this research is a computer. The computer can be equipped with an NVIDIA GPU since it provides a massive speed increase in training and testing time. However, it does support running on any CPU. The computer used for training and testing will be on an NVIDIA 2060 GTX Mobile GPU and an Intel Core i7 CPU. It is imperative that the computer has efficient cooling systems to prevent overheating. It is also being considered that the tests will be conducted on a cloud computer provided by Google Colab, a website that offers free GPU computing services (a subscription is also available in which Google Colab will offer even faster GPUs and more RAM). The GPU provided by Google Colab without any subscription is an NVIDIA Tesla K80 GPU with a limit of 12 hours per week.

The computer must be equipped with many frameworks for this research project. The programming language Python (version 3.8.10) will be used for building and testing VNN. Python can be installed using the instructions provided on their website: https://www.python.org/downloads/. Pytorch is a machine learning framework that automates many functions such as back-propagation. The specific version of Pytorch is 1.10.2+cu102 (this version supports GPU acceleration). Pytorch can be installed by using “pip”, a Python package manager (built-in after installing Python). In order to test the Robot Design and Control environments, the research project will be using Evo Gym, which provides soft-body environments used during training and testing that have to achieve various tasks (such as throwing a block or climbing). Instructions to install Evogym are available in their github repository: https://github.com/EvolutionGym/evogym/blob/main/README.md. Furthermore, wandb 0.12.16, available through pip, will be used for logging and tracking the progress of training and testing. Tqdm 4.62.4, available through pip, is going to be used simply for a progress bar that can be displayed on a terminal window. An internet connection will be needed to install all the dependencies. However, after installing all the dependencies, an internet connection is not needed (unless the model is trained/tested in Google Colab or a similar GPU service).

No potential risk or safety precautions needed.

2 Methods

The following section will explain how Virtual Neural Networks work, along with an explanation of the experiment setup of each of the benchmarks used to evaluate Virtual Neural Networks against other types of neural networks.

2.1 Virtual Neural Networks

Virtual Neural Networks (VNN) are very similar to dense neural networks, except the input and output layers have their weights generated by the parameter dense model. As input, the parameter dense model will take the index of the input node, the index of the output node, and the input value corresponding with the input node. The parameter dense model will then output a weight number corresponded between the input node and the output node. Thus, the parameter dense model will be expected to output $N \times M$ vector, where N represents the input space (can be variable) and M represents the output space (fixed, this depends on the input of the next dense layer). With this weight matrix, the input of the VNN will be multiplied by this weight matrix to get the desired output size. Note that in a normal layer, a bias would be included. However the VNN will not compute the bias and instead multiply the weights with the input vector.

After the output of the input layer is computed, then it will be passed through a series of normal neural networks with fixed input and action space. Then, the final output layer can either be a weight generation layer (like the input layer) or can be a normal dense layer with fixed output size. The latter setting can be used for classification, and will be used for benchmarking Sentiment Analysis and Image Classification.

Regularization Techniques such as Dropout Layers and Batch Normalization Layers will be used to ease training.

2.2 Image Classification Benchmark

The image classification benchmark will test whether VNN can recognize images with different sizes. Namely, the sizes will be either 125 by 125 pixel, 100 by 100 pixel, or 75 by 75 pixel images. The dataset will be from the Intel Image dataset. The Intel Image Dataset will contain 14,034 training images, and 3,000 test validation dataset. The goal of this dataset is to process the image and output whether the image is a building, forest, glacier, mountain, sea, or a street.

In order to evaluate the different models, we are going to test the percentage of correctness on the test validation images, and compare that with a normal convolution neural network, trained on only fixed sized images (this will act as our control). However, the VNN will receive 3 times the amount of training images than the control group, because each image is resized into 3 different sizes. To account for this, the control group will be trained for 5 epochs while the VNN will be trained for 3 epochs.

The VNN will contain a convolution neural network that will be able to process variable sized images before the VNN. However, the control group will not have a VNN after the convolution neural network and instead have a normal dense layer.

2.2 Sentimental Analysis Benchmark

The sentimental analysis benchmark will test whether a VNN can understand text. Sentimental Analysis is where the neural network is exposed to a sentence. First, the sentence will be processed by a tokenizer. This embedding not only separates the sentence with words or similar ending (ex: -ing), but it will also assign a unique number to each of the words for processing. This function will be implemented using HuggingFace's Tokenizer function. We then embed each unique id into a high-dimensional vector (usually a N by 512 vector, where N is the number of elements from the tokenizer). This embedding function will also be implemented using HuggingFace's Embedding function. Finally, we could use the output of this to feed it through the VNN. The dataset used will contain 1,600,000 twitter tweets and whether each tweet is happy, sad, or neutral. This benchmark will remove the tweeks that are neutral, which only prompts the neural network to check whether a sentence is happy or sad. The test validation dataset will be 800 sentences from the 1,600,000 twitter tweets. When training the VNN, it will only use 600,000 sentences for training. None of these sentences will exist in the test validation dataset.

In order to test the control, HuggingFace provides a sentimental analysis "pipeline", where you could enter any sentence and it will try to predict whether it is happy or sad. The test model will be HuggingFace's Pretrained BERT model, which is trained on the GLUE Text Classification database. The BERT model is a variant of the original Transformer model that is used for text classification.

2.3 Autoencoding Benchmark

Autoencoding involves taking a vector of numbers, compressing it down to a smaller size, and then decompressing it to its original size. The VNN will aim to input and output the same input. The encoder will consist of the VNN input parameter layer, a 512-node dense layer, and a 256-node dense layer. The decoder will consist of a 256-node input layer, a 512 dense layer, and finally a VNN output parameter layer. The VNN output parameter layer will expect to output the same size as the input sequence itself.

The control group for the Autoencoding Benchmark will be a LSTM model trained from scratch, using a similar procedure as described above. The encoder of the LSTM model will be a many-to-one LSTM model. First the encoder will cycle through each embedding of each word. Then once the final word is processed, the LSTM will produce a vector with size 512. Then the vector will go through a single neural network layer with an output size of 512, which then is passed through the decoder.

The LSTM decoder model will take the input in timestep 0. Then for the next N timesteps (where N represents the length of the input sequence), the LSTM will output a scalar which is then concatenated across all timesteps to form the original vector. The input of the LSTM when it is from timesteps 1 to N+1 will be a positional encoded vector with size 512. Usually the positional encoded vector will be added to the word embedding, however in this case it will simply add to a vector of zeros.

During training, the encoder and decoder will combine to become a single neural network, and then the random generated sequences will be from 0 (inclusive) to 1 (inclusive). The length of the sequences will vary with length 256 to 512.

2.4 Robot Design and Control Benchmark

The Robot Design and Control benchmark will test if a VNN can function in a variable domain and output reinforcement learning environment. More specifically, it will test whether the VNN can not only control the model, but also design a soft body robot itself.

The environment will be built using EvoGym (Bhatia et al.). The EvoGym environment will allow a policy to not only control a soft body robot, but also design a 2d soft body robot. The soft body robot will be represented by connected voxels, where a voxel can be a vertical actuator, horizontal actuator, a soft voxel, or rigid voxel. If the voxel is a vertical actuator or horizontal actuator, then an action can be applied to the actuator in order to move the soft body robot. For each actuator, the neural network should output a value between 0.6 to 1.6, where 0.6 would compress the soft-body actuator and 1.6 would expand the soft-body actuator.

For every timestep, the VNN model will input the state of the environment, the current timestep the environment is on (this will be encoded using positional encoding), and the design of the robot that the model generated at time step 0. The expected output will be dependent on the number of actuators that the design has from the VNN model. If the VNN model outputs a design with no actuators, then the environment will automatically end the episode with final reward -10. During timestep 0 where the task of the VNN is to generate a robot design, the input will consist of a vector of ones with length 256 as input, and the expected output will a N x M x 4, where N represents the max height (for every environment, this parameter will be set to 5), M represents the max width (for every environment, this parameter will be set to 5), and 4 represents the types of voxels that the soft body can contain (horizontal actuator, vertical actuator, rigid voxel, or soft voxel).

The control of the environment will be the model proposed by the EvoGym paper (Bhatia et al.). Training this model and the VNN will use the PPO control algorithm (Schulman et al.) to optimize the policy. The implementation of the PPO control algorithm is from Ilya Kostrikov. The model proposed by the EvoGym paper involves not only a control neural network, but also a genetic algorithm co-design algorithm, which will generate the designs. However, with using VNN, the control and design will be done under a single neural network. The data will be extracted from Figure 3 of the EvoGym paper (Bhatia et al.).

EvoGym offers a number of environments. However, the benchmark will only test 3 Evo Gyms environments: Climber-v0, UpStepper-v0, and Catcher-v0. The description for each test environment is provided:

- UpStepper-v0 is an environment implemented from Evogym where the policy designs and controls a 2-dimensional representation of a soft-body robot to walk up a staircase of varying lengths.
- Climber-v0 is an environment implemented from Evogym where the policy designs and controls a 2-dimension representation of a soft-body robot to climb as high as possible. The robot will climb on poles that are 5 blocks wide and flat (no obstacle between robot and goal).
- Catcher-v0 involves a policy to create and design a 2-dimensional representation of a soft-body robot that catches a box initialized 40 units above the soft body robot.

Works Cited

Bhatia, Jagdeep Singh, et al. "Evolution Gym: A Large-Scale Benchmark for Evolving Soft Robots." *Openreview.net*, 9 Nov. 2021, openreview.net/forum?id=eoTy4ihL0W. Accessed 8 Sept. 2022.

Hochreiter, Sepp, and Jürgen Schmidhuber. "Long Short-Term Memory." *Neural Computation*, vol. 9, no. 8, Nov. 1997, pp. 1735–1780, 10.1162/neco.1997.9.8.1735.

Schulman, John, et al. "Proximal Policy Optimization Algorithms." *ArXiv.org*, 2017, arxiv.org/abs/1707.06347.

Vaswani, Ashish, et al. "Attention Is All You Need." *ArXiv.org*, 2017, arxiv.org/abs/1706.03762.