SeqVista: Sequence to Sequence Model for Vision-Language Navigation

Zeynep Hanife Akgül
Bilkent University
Computer Science (CS)
Ankara, Türkiye
hanife.akgul@ug.bilkent.edu.tr

Arshia Bakhshayesh
Bilkent University
Computer Science (CS)
Ankara, Türkiye
arshia@ug.bilkent.edu.tr

Huzaifa Huzaifa
Bilkent University
Computer Science (CS)
Ankara, Türkiye
huzaifa@bilkent.edu.tr

Emre Karataş
Bilkent University
Computer Science (CS)
Ankara, Türkiye
emre.karatas@ug.bilkent.edu.tr

Faaiz Khan
Bilkent University
Computer Science (CS)
Ankara, Türkiye
faaiz.khan@ug.bilkent.edu.tr

***Abstract*—The SeqVista project represents a significant undertaking within the field of Artificial Intelligence, with a particular emphasis on advancing Vision-Language Navigation (VLN). The core objective is to develop a robust Sequence-to-Sequence (Seq2Seq) model capable of training an autonomous agent for effective VLN. This requires equipping the agent with the proficiency to interpret and execute natural language instructions within unseen environments by integrating advanced natural language processing and computer vision techniques. A key aspect of this project involves taking advantage of the Matterport 3D simulator in conjunction with the Room-to-Room (R2R) dataset, which provides a graph-based environment rich in real-world imagery and diverse instruction-trajectory pairs.**

***Index Terms*—Artificial Intelligence, Vision-Language Navigation, Sequence-to-Sequence Model, Attention Component, Hyperparameters, Natural Language Processing, Computer Vision, Matterport 3D Simulator, Room-to-Room Dataset, LSTM, RESNET-152, Autonomous Agents.**

I. Introduction

SeqVista is an attempt to explore the realm of VLN within Artificial Intelligence. With our understanding of previously available resources, we aim to re-implement a Seq2Seq model that can train agents proficient in interpreting natural language instructions to achieve tasks in the Matterport3D Simulator and beyond. Once the training is completed, agents are tested on unseen data, and the created trajectories are obtained to view how the agent works in real-time. The main objective of the project is to study the effect of removing the attention mechanism component in a Seq2Seq model. A secondary objective of this project is to understand the effect of hyperparameters on the end results of training a model.

II. Literature

A detailed literature review is quite important to help understand what is already known in the area of research. By identifying gaps, the literature review reasons the research. Additionally, gathering together the key ideas from existing studies helps to build a strong foundation. Learning from past studies allows us to make smart choices in the project's design and methodology.

This part is divided into five subsections. First, key techniques will be discussed. Then, a comparative analysis will be made, discussing the future directions.

A. Vision-Language Navigation

Research on the natural language command of robots in unstructured environments has been ongoing for decades. Previous approaches often simplified the problem by either pre-defining navigation goals and objects or operating in limited perception environments [5]. The development of new benchmark datasets has driven advances in image captioning, visual question answering, and visual dialog. These have enabled end-to-end training on raw image data but have not allowed for agent movement or camera control, a gap the R2R benchmark aims to fill [6, 7, 8, 9].

1) Existing Approaches in Vision-Language Navigation:

- Natural Language Processing (NLP) Techniques: Natural language processing techniques are commonly used in order to understand the navigation instructions, as seen in papers. For example, Mei, Bansal and Walter use a neural mapping approach to map instructions to a sequence of actions. Their purpose is to train a neural network to be able to predict the sequence of actions which an agent is supposed to take based on the input instruction and visual features of the environment. The network is trained on a large dataset of the pairs of instruction-action, and the network learns to generalize to new instructions and environments [10]. Similarly, Tellex et al. [11] and Fasola et al. [12] focus on mapping verbs and spatial relations of verbs to certain actions in the environment. For example, the verb "go" might

be associated with the action of moving forward, while the spatial relation "left of" might be associated with the action of turning left. Agents can navigate based on natural language instructions by mapping these natural language instructions into actions in the real world-like environment. One other research, proposed by [10], introduces an end-to-end, sequence-to-sequence approach to mapping natural language instructions to actions where the local and the the observable world state is given using a bidirectional LSTM-RNN model with a multi-level aligner. However, it is also mentioned that free-form instructions in unknown environments are problematic due to their ambiguity and complexity, which is a known issue in NLP techniques. To deal with the problem, the paper proposed a model that uses a multi-level aligner to focus on certain parts of a sentence that are relevant to the current world state. The paper then analyzes its performance through a series of decomposition experiments. Overall, NLP techniques are used to generate a sequence of actions that an agent can take to navigate in the environment based on natural language instructions, even in complex and ambiguous environments.

- Computer Vision Techniques: In VLN, several computer vision techniques are commonly used to make agents analyze and respond to natural language instructions in a visual environment. For example, CNN is used for feature extraction to understand the content of the environment for decision-making. For example, in [13], pre-trained CNN Resnet-152 on ImageNet to extract feature vectors of images implemented. Other than CNN, attention mechanisms are also used in studies. In [2], attention mechanisms are used to focus on relevant parts of an image, allowing the model to concentrate on the specific details in the natural language instructions. Several obstacles occur with these techniques. For example, one of the problems is the generalization to previously unseen environments. In [14], it is mentioned that the Multimodal Indoor Simulator for Navigation in Complex Environments (MINOS) evaluates the agents in environments that they were not trained in, allowing for the study of generalization. This is important for VLN systems to be able to navigate effectively in new and unfamiliar environments.

- Integration of NLP and Computer Vision: Combining language understanding and visual perception can cause several challenges, especially if the agent only has access to visual content without the ability to interact with the environment. To deal with the issue, in [15], Interactive Question Answering (IQA), which requires the agent to interact with a dynamic environment is proposed. IQA involves tasks such as navigation, acquiring an understanding of the environment, interacting with objects, and planning and executing actions based on questions asked. Furthermore, Multimodal Attention Mechanisms are also used to operate across both language and vision modalities. In [2], attention mechanisms that function across both language and vision domains enhance the model's ability to concentrate on relevant elements within textual instructions and visual environments, improving navigation precision.

2) Benchmark Datasets for VLN: Benchmark datasets are quite significant in the development of VLN systems. The R2R benchmark is one of those, providing a diverse set of instruction-trajectory pairs associated with Matterport 3D environments. This dataset, when associated with Matterport 3D, offers a robust testing environment for VLN agents, allowing evaluations on realistic, building-scale environments. The availability of such benchmark datasets is significant for training models and assessing their performance in unseen scenarios in VLN research. Similar to research using R2R Benchmark, as in [2], this dataset is also used in SeqVista because of its ability to be associated with the Matterport 3D Simulator.

B. Sequence-to-sequence Models

Agents in various papers are implemented by the Sequence-to-Sequence (Seq-2-seq) model. Seq-2-Seq models are recurrent neural network (RNN) models based on LSTM-based architecture [16]. It is worth mentioning that most of the papers implement the Seq-2-Seq model with an attention mechanism [16]. However, the Seqvista project re-implemented the Seq-2-Seq model **without an attention mechanism** to obverse the importance of the attention mechanism in the related Seq-2-Seq models.

Detailed information related to the attention mechanism should be mentioned to highlight the difference between SeqVista and other papers implementing the Seq-2-Seq model.

An attention mechanism functions as a crucial component within a neural network framework[17]. It operates by determining the significance of various parts of the source data at each step of the decoder's process. This mechanism allows for a more nuanced approach compared to traditional methods where the encoder had to encapsulate the entire source information into a single, compact vector. Instead, it provides detailed representations for each token in the source - for instance, it can offer representations for every state in a Recurrent Neural Network (RNN) rather than just the final state[17]. This enables the model to focus selectively on different parts of the input data, enhancing its ability to understand and process complex patterns and dependencies, especially in tasks such as language translation or image recognition, where context and specific details are crucial[17].

1) Seq-2-Seq Models in Navigation: Matterport 3D simulator action space is state-dependent; agents are allowed to make pre-defined actions based on choices[13]. Since the Matterport 3D camera takes visuals every 30 degrees, the model and navigation of Seq-2-Seq models are defined to move every 30 degrees.

For image observation, the papers that have been read implemented pre-trained CNN Resnet-152 on ImageNet to extract feature vectors of images [13]. That is, an embedding is learned from every action in the action space and then takes relative actions in the navigation.

Most of the papers predict future actions based on the attention mechanism explained in detail above. It computes an attentional hidden state and calculates the predictive distribution over the next states as softmax[18]. Although it is an enormous advantage to increase the model's performance, Seqvista is based on the Seq-2-Seq model without an attention mechanism to highlight the importance of an attention mechanism in these models.

C. Challenges and Open Problems

While VLN has developed significantly in recent years, several challenges persist in making these systems robust and adaptable. One crucial problem is the generalization of models to previously unseen environments. Addressing this challenge is critical for VLN systems to navigate effectively in new and unfamiliar settings [14]. This issue also appears in SeqVista; however, it is not analyzed further to avoid misdirecting the project scope. Another significant challenge involves integrating natural language processing (NLP) and computer vision, particularly when the agent has limited interaction with the environment [15]. Interactive Question Answering tasks, as proposed in [15], show the complexity of tasks requiring dynamic interactions in VLN systems. Overcoming these challenges will contribute to developing more versatile and capable VLN agents.

D. Comparative Analysis

A comparative analysis of existing approaches in Vision-Language Navigation with SeqVista allows for the analysis of diverse techniques. Natural Language Processing techniques, as studied by Mei, Bansal, Walter, Tellex, et al., and Fasola and Mataric, focus on mapping the natural language instructions to sequences of actions, grounding verbs, and spatial relations to actions in the environment [10, 11, 12]. SeqVista also uses NLP techniques to map the instructions to specific actions like these studies. Computer Vision techniques, such as those discussed in [14], emphasize the evaluation of agents in previously unseen environments. Integrating NLP and Computer Vision introduces challenges associated with dynamic environment interaction. SeqVista, with the advantage of Matterport 3D simulator, experiences a dynamic environment interaction, differentiating from studies such as [14]. The Seq-2-Seq model, a recurrent neural network architecture, is widely adopted [16], with attention mechanisms enhancing model performance in various papers [17]. However, SeqVista intentionally excludes an attention mechanism to investigate its impact on Seq-2-Seq models in the VLN context. This comparative analysis guides the SeqVista project in understanding the strengths and limitations of different methodologies.

E. Conclusion and Future Directions

In conclusion, the literature review provides valuable insights into the existing state of Vision-Language Navigation. Essential techniques have been explored, including NLP and Computer Vision integration, Seq-2-Seq models, and attention mechanisms. It has been observed that the lack of attention mechanisms reduces the agent's overall performance, highlighting the importance of such mechanisms. The challenges, such as generalization to unseen environments and dynamic environment interaction, show the complexity of VLN systems. The SeqVista project aims to contribute to this massive field by intentionally excluding an attention mechanism in the Seq-2-Seq model, emphasizing the importance of this component in VLN models.

III. Description

The project aims to re-develop a **Sequence-Sequence (Seq2Seq) model** for training an autonomous vision-language navigation (VLN) agent. An important difference is that the attention mechanism component is removed from the model, and then a detailed comparison is made between the existing model with the attention component and the one created in this project. VLN refers to the navigation of embodied agents in an unseen environment based on natural language instructions. The agent must understand the natural language instructions using the natural language processing and then use computer vision to identify potential landmarks mentioned in the instructions to initiate its navigation actions. As an example, the agent might be required to execute the instruction, *"Walk down the stairs to the bottom of the staircase. Continue down the next small flight of stairs towards the bathroom at the lower level."*

A. Simulator

The Matterport 3D simulator was used. The Matterport 3D simulator is a graph-based environment based on the Matterport 3D dataset [1], which consists of 90 building-scale environments. The Matterport 3D simulator was used in conjunction with the Room-to-Room (R2R) dataset [2]. This dataset consists of a large number of diverse instruction-trajectory pairs associated with Matterport 3D environments. The action space in this simulator consists of the following actions:

1) **Forward**: This action moves the agent to the navigable node, which is closest to the center of the agent's field of view (FOV).
2) **Right, Left**: These actions would change the camera heading by 30 degrees.
3) **Up, Down**: These actions would change the camera elevation by 30 degrees.
4) **Stop**: This special action is initiated for the termination of the navigation episode.

At each time step, the output of the Matterport 3D simulator consists of:

- First-person RGB observation of the environment.
- The set of next navigable nodes from each node.

B. Seq-to-Seq Model

The VLN problem can be solved using a Sequence-to-Sequence model, also called the encoder-decoder model, which takes in a sequence of natural language instructions to produce the sequence of navigational actions, as shown in Fig 1. The input to the encoder-decoder model is the sequential instruction $\mathbf{w} = \{w_1, w_2, w_3, \ldots, w_n\}$ where w_i represents a learned embedding vector for each of the n-word tokens present in the instruction. Similarly, the image features for RGB observations were also computed beforehand using RESNET-152 architecture, which is pre-trained on ImageNet architecture and makes use of convolutional neural networks for the extraction of image features. We represent the extracted image features by $\mathbf{f}_t$.

Also, the embeddings for each of the actions were learned separately. The encoder LSTM takes in the instruction sequence $\mathbf{w}$ to compute the final hidden state $\mathbf{h}_n$ from the encoder context $\mathbf{h} = \{h_1, h_2, h_3, \ldots, h_n\}$ as:

$$h_n = \text{LSTM}_{\text{enc}}(w_n, h_{n-1})$$

This final hidden state of the encoder LSTM is passed to the decoder LSTM for outputting the sequence of navigational actions. At each time step t, the decoder observes the hidden state $\mathbf{h}_n$, the embeddings of the previous action a_{t-1}, and the representation of current RGB observation $\mathbf{f}_t$ to compute the raw probabilities for the next action. The action embeddings and the image representations are concatenated into a single vector $\mathbf{q}_t$. So, the decoder LSTM operates as:

$$\mathbf{h}'_t = \text{LSTM}_{\text{dec}}(\mathbf{q}_t, \mathbf{h}'_{t-1})$$

where $\mathbf{h}'_t$ gives us the raw probabilities for the next action. These raw probabilities are passed through the softmax function to obtain the final output in terms of navigational action in the simulation environment as

$$a_t = \text{softmax}(\mathbf{h}'_t)$$

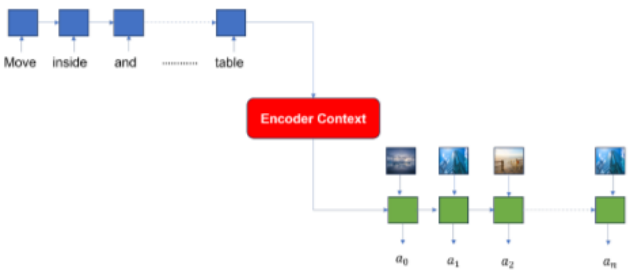

Fig. 1. Seq-Seq Model

IV. METHODOLOGY

This section is spared for a detailed explanation of the total work done for this project.

A. Simulator

In the pursuit of comprehending the MatterSim simulator and its API, significant milestones were achieved through systematic exploration and practical application. This section also outlines the milestones reached during the understanding of the simulator, emphasizing a hands-on approach.

Technical Exploration:

- **Configuration Mastery:** The initial phase involved mastering the configuration parameters of the MatterSim simulator. Functions like `setCameraResolution`, `setPreloadingEnabled`, `setDepthEnabled`, `setBatchSize`, and `setCacheSize` were dissected for their role in the simulator setup.
- **Simulator Initialization:** The simulator was initialized using the `initialize` function, setting the stage for subsequent interactions. A detailed understanding of parameters like scan IDs, viewpoint IDs, headings, and camera elevations was crucial for accurately defining the simulator's starting state.
- **Action Execution:** The core of agent interaction was explored through the `makeAction` function. The program executed forward movements with varying heading and elevation changes. This hands-on experience facilitated a deep understanding of how the agent navigates within the simulator.
- **State Inspection:** Regular checks of the simulator state using the `getState` function provided insights into the dynamic changes during and after agent actions. Extracting information such as `scanId`, step count, RGB and depth images, location details, and heading facilitated a comprehensive understanding of the simulator's inner workings.

Programming Accomplishments:

- **Code Implementation:** A simple Python program was developed to showcase the practical application of the acquired knowledge. The program executed a sequence of actions, moving the agent forward in different elevations and headings for a specified number of steps (see Appendix 1 for the code).
- **Iterative Adjustment:** The program's iterative nature allowed for continuous adjustments based on real-time observations of simulator behavior. This iterative process contributed to a more nuanced understanding of the simulator's response to different inputs.

B. R2R Cleanup

For the development of the SeqVista project, an open-source Matterport 3D Simulator Dataset is utilized [1]. Since the key aspect of this project relies on the graph-based Matterport 3D Simulator, the R2R Dataset is employed in conjunction with the Matterport 3D Simulator.

Initially, the original Matterport 3D R2R Dataset consisted of 90 houses composed of 1.3 TB of data [3]. Due to hardware (especially RAM, GPU & storage capacity) and time

constraints of this project, it is decided to use 10 randomly selected houses, which are partitioned into 3 different categories:

- **Training:** 7 Houses - 70% of data
- **Testing:** 2 Houses - 20% of data
- **Validation (Unseen):** 1 House - 10% of data

Randomly selected 10 houses composed of 150 GB of data, which is decided to be enough in order to get meaningful results, taking account of the limitations mentioned above favoring optimality.

Houses in the Matterport 3D R2R Dataset are labeled with their scan codes. The dataset consists of 4 different JSON files, which are named as follows:

- `R2R_test.json`
- `R2R_train.json`
- `R2R_val_seen.json`
- `R2R_val_unseen.json`

These JSON files give insight into the scan (house), the path IDs related to these scans, their headings (in radians), and instructions (in the natural English language) to accomplish these moves.

C. Seq-Seq Agent Development & Training

For the development of the Seq-Seq agent using LSTMs, an open-source PyTorch library v1.5.0 was used. The sentences were tokenized in white spaces. We follow the standard hyperparameter values used by [2]. The hidden size of neural network layers was set to 512. The size of action and word embeddings was set to 32 and 256, respectively. The image features were extracted from the second-last layer of RESNET-152 architecture, which has a size of 2048. We set the maximum episode length to 20. The loss criterion used in this case was a cross-entropy loss. We use the Adam optimizer for the optimization of encoder and decoder parameters. We use a learning rate of 1×10^{-4} and a weight decay of 5×10^{-4} in order to train our agent. The dropout ratio to avoid overfitting is set to 0.5. The agent was trained using two different feedback approaches, namely "sample" and "teacher" feedback. In sample feedback, the agent uses the predicted previous action a_t based on the learned model for getting the probability distribution over the next action in the output stage of the encoder-decoder model. However, in the "teacher" feedback approach, the agent always uses the correct ground truth in the previous action a_t^* instead of the predicted previous action to get the probability distribution for the next action. The agent was trained on a core i9 machine with 125 GB RAM and incorporating an NVIDIA GeForce RTX 3090 Ti/PCIe/SSE2 GPU. We trained the agent on a batch size of 100 with a fixed number of iterations, which was 20000 for the "sample" feedback approach and 5000 for the "teacher" feedback approach. It took approximately 3 hours to train the agent using the "sample" feedback approach, while the "teacher" feedback approach took about 45 minutes. The model weights were saved in a separate directory to be used for evaluation on the test split of the R2R dataset. The training & validation loss curves, along with the success rate, are shown in Fig 2.

D. Experimentation on Hyperparameters

As a secondary objective, the project also observed the effects of certain hyperparameters on the training of the model and, subsequently, the end results. From the list of hyperparameters [4], the following were selected:

- **Learning Rate:** This parameter determines the step size during the optimization of the model's weights, influencing how quickly or slowly the model learns from the training data. A proper learning rate is essential for stable convergence, ensuring that the model effectively adapts to the patterns in the input sequences.
- **Weight Decay:** Weight decay, also known as L2 regularization, is one of the regularization techniques used in machine learning, including sequence-to-sequence models. Within seq-to-seq models, weight decay is used to prevent over-fitting by using penalty terms in the loss function. This is ensured by penalizing larger weights more during the training process. Weight decay also improves the generalization of the model to unseen data. In the literature review, it has been seen that one of the challenges that researchers faced was the generalization of models to previously unseen environments [14]. With this in mind, research is done to see if increasing the weight decay would improve the generalization of the unseen data.
- **Batch Size:** In the context of this project, the hyperparameter of batch size is a significant focus. Batch size, which determines the number of training samples processed before the model's internal parameters are updated, has a crucial impact on the training dynamics of the sequence-to-sequence models. A smaller batch size often results in more frequent updates with higher variance in the gradients, which can lead to better generalization by preventing the model from settling too quickly into local minima. On the other hand, larger batch sizes provide computational efficiencies through parallel processing and smoother gradient estimations, but they may lead to challenges in generalization, especially when dealing with complex tasks such as Vision-Language Navigation (VLN). As a result, selecting the appropriate batch size is a balancing act between training efficiency, model accuracy, and its ability to generalize to new, unseen data.
- **Hidden Size:** This hyperparameter refers to the dimensionality of the hidden state vectors in the model's encoder and decoder. In practice, the choice of the hidden size is often based on experience, and it depends on various factors like the complexity of the task, the size of the dataset, computational resources available, and the desired balance between model capacity and generalization. Experimenting with different hidden sizes and evaluating the model's performance on validation data can help determine the optimal size for a particular task. For the task at hand, looking at previous implementations of the model [4], it was apparent that the hidden size was almost always a power of 2. The suspected reasons

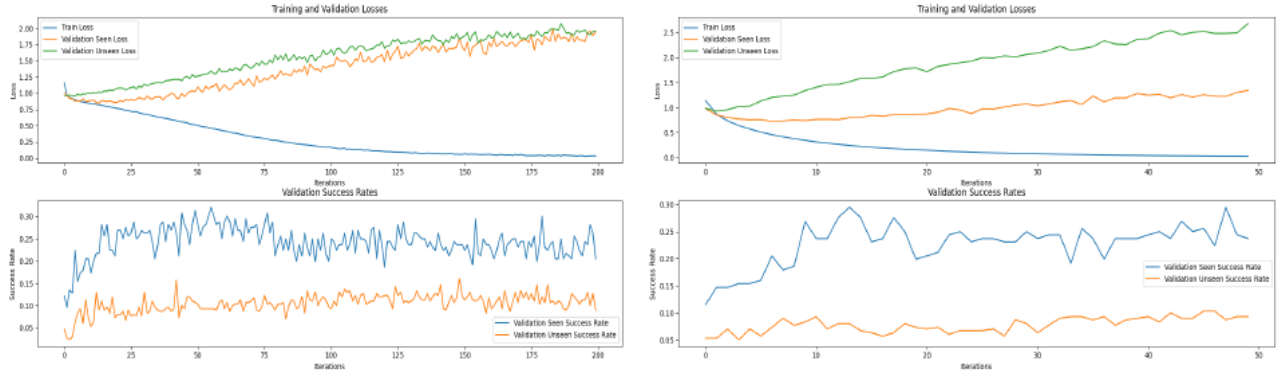

Fig. 2. Training, Validation Losses & Success Rate for Sample (Left) and Teacher Feedback (Right)

for this are that it is efficient for hardware, is easier for memory alignment, and is also just the conventional practice. This is to say that the value may deviate from powers of 2, but usually it is kept at those values for the aforementioned reasons.

V. RESULTS

This section highlights the results that were observed for the duration of this project.

A. Seq2Seq Model with and without Attention

Table I below provides context about the results observed in the training and testing of this project's model, without the attention mechanism, comparing them to the results seen by Anderson et al. and their model containing the attention mechanism. The results were based on the following metrics:

- **Navigation Error:** This metric provides the distance between the agent and the target position.
- **Path Length:** This metric provides the distance traveled by the agent until the end of the task.
- **Success Rate:** This metric provides the rate at which the agent managed to guide itself to an ending position that was within the acceptable threshold of the navigation error.
- **Oracle Success Rate:** This metric provides the rate at which the agent managed to guide itself to an ending position that was within the acceptable threshold of the navigation error, aligning with multiple valid reference positions in the training data.

TABLE I
COMPARING RESULTS

Result Metrics	Anderson et al.	SeqVista
Naivgation Error	7.85 meters	9.3869 meters
Path Length	8.13 meters	6.51973 meters
Success Rate	20.4%	8.017%
Oracle Success Rate	26.6%	12.236%

B. The Effects of Hyperparameters

- **Learning Rate:** Experimentation with the learning rate revealed a trade-off between convergence speed and success rate in the Seq2Seq model. Increasing the learning rate facilitated faster convergence. However, this expedited learning process came at the cost of reduced success rates. The heightened learning rate led to increased oscillations during training. (see Appendix Fig. 8.)
- **Weight Decay:** Although increasing weight decay improves the success rate up to a certain point, after a maximum of success rate, a minimum of navigation error, and a minimum of loss, the plot changes behavior to grow oppositely. To represent the expected changes, Anderson's analysis of validation loss, navigation error, and success rate is taken and modified hypothetically. The manipulated graph can be seen in the Appendix, Fig. 10. Hyperparameter: Weight Decay graph.
- **Batch Size:** In this project, we systematically evaluated various batch sizes to determine their impact on the computational efficiency and performance of the Sequence-to-Sequence (Seq2Seq) model. Our empirical results indicate that a medium-sized batch offers a pragmatic compromise, enhancing the model's training stability and generalization capabilities while maintaining reasonable computational demands. Particularly, batch sizes in the range of 150-175 have yielded optimal outcomes, striking a judicious balance between training speed and model accuracy on our dataset.
- **Hidden Size:** Experimentation with the hidden size largely remained inconclusive. The initial idea was to increase the hidden size as the Seq2Seq model without attention performed quite poorly (with an 8% success rate). This, however, did not end up fruitful as the project was severely limited by its computational capabilities. Doubling the hidden size from 512 to 1024 caused the training time to increase as well, so much so that it was unable to converge within a suitable time.

VI. DISCUSSION ON RESULTS

This section aims to understand and conclude, in detail, the results observed for the duration of this project.

A. Understanding the Effects of the Attention Mechanism

In the previous section, the results between SeqVista's model and the model created by Anderson et al. were displayed and compared. To infer a reasonable conclusion from these results, it is important to look at what is different between the two models: the attention mechanism. The attention mechanism, amongst other things, allows the model to do two things that are important to the context of VLN:

- The attention mechanism allows the model to "focus" on different parts of the input. It was previously explained that the instructions are tokenized by whitespaces: this means that the attention mechanism is vital to understand the context of a token within an instruction
- Due to this selective focus, it is able to decide which part of the input is more important which further allows it to understand the instruction in a more "human" way as opposed to just simply segmenting an instruction token-by-token.

It is therefore concluded that the **increased navigation error** in the model without the attention mechanism is because of this selective focus and especially the ability to handle long-range dependencies. Building on this, the **decreased success rate and oracle success rate** in SeqVista's model can also be explained as the attention mechanism leads to more complex and better decision-making. However, there was an anomaly observed in the path lengths. Seqvista's model showed that the addition of the attention mechanism resulted in undesirable **longer path lengths**. At the end, it was hypothesized that the more complex decision-making due to the attention mechanism encourages the VLN agent to explore more, and thus provides a trade-off: decreased path efficiency for increased path precision.

B. Understanding the Effects of Hyperparameters

- **Learning Rate:** An essential aspect of the observed phenomena lies in the increased aggressiveness of weight updates facilitated by a higher learning rate driven by more responsive gradient information. However, this heightened adaptability introduces a notable challenge—the model exhibits a higher likelihood of overshooting optimal weight configurations. The delicate equilibrium between rapid adaptation and accurate navigation tasks is disrupted as the model grapples with the risk of veering away from the desired trajectory. The intricate interplay between the learning rate and the model's capacity to strike a balance between swift adjustments and precision underscores the nuanced challenges in hyperparameter tuning for vision-and-language navigation tasks. These insights shed light on the intricate dynamics influencing model behavior and guide a more thoughtful approach to parameter selection for optimal task performance.
- **Weight Decay:** Increasing weight decay, especially increasing it too much, can have several effects on the training process and the model's performance. The graphs examine the model's performance in three different categories, so three different effects of high weight decay can be discussed.
  - Validation Loss: A high weight decay can cause an increased validation loss during training. This result is observed because weight decay overly penalizes weights in the model as it is a regularization technique that penalizes large weights. That's why if the weight decay is too high, a decrease in the model's ability to fit the training data can be observed, resulting in higher validation loss.
  - Navigation Error: In the context of navigation tasks, in order to reach the goal location, the model learns mapping natural language instructions to a sequence of actions. A high weight decay affects the model's ability to learn complex relationships between the natural instructions and the expected actions, which causes an increased rate of errors in navigation predictions. This happens because of the model's ability to learn and to generalize from the training data being compromised, which leads to the prevention of capturing the nuances and complexities of the task.
  - Success Rate: Since the validation loss and the navigation error rates increase, the success rate naturally decreases. Decreased learning capacity, as penalties are high and impaired generalization, causes success rates to decrease. The model struggles to accurately deploy natural language instructions to a corresponding sequence of actions.
- **Batch Size:** The batch size hyperparameter significantly affects the training process of deep learning models. A smaller batch size may enhance generalization and enable training with less memory, though it requires more iterations to converge. Larger batch sizes can quicken computation and offer more stable convergence, but they may generalize less effectively and demand more computational resources. In our project, we tested various batch sizes to strike a balance between computational efficiency and the Seq-Seq model's performance, aiming for optimal results on our dataset.
- **Hidden Size:** The effect of the hidden size parameter is not completely unknown, even though the model was unable to converge within a reasonable time. It is suspected that the potential advantages of increasing the hidden size is an increased model capacity and expressiveness due to the fact that a higher dimensionality allows for more detailed and nuanced information. This, in turn, increases the performance of the model. However, on the other hand, increasing the hidden size hyperparameter increases the computational time as is apparent from the project, but it may also risk overfitting, where the model becomes too specialized in learning the training data so it is unable

to generalize to unseen data.

VII. WORK DISTRIBUTION

- **Zeynep Hanife Akgül**: Literature Review, Challenges and Open Problems, Comparative Analysis to other research, Exploration of the weight decay hyperparameter, Matterport 3D Simulator set up trial run using AWS Linux instance.
- **Arshia Bakhshayesh**: Simulator installation, Model training(original anderson model on 10 houses + different learning rate), Learnt agent evaluation, Transformer methodology experimentation, Trajectory visualization, Learning rate hyperparameter experimentation and analysis.
- **Huzaifa Huzaifa**: Development of Seq-Seq model, Training & Evaluation of Learnt Agent.
- **Emre Karataş**: Code Debugging Seq-2-Seq Model, Exploration of Hyperparameter batch size, Comparing Initial SeqVista results with Anderson [2], Matterport 3D R2R Dataset preprocessing, Literature Review.
- **Faaiz Khan**: Understanding and mastery of the attention component in a Seq2Seq model, Inference of results, Comparison between Anderson et al's model, Exploration of the hidden size hyperparameter.

APPENDIX

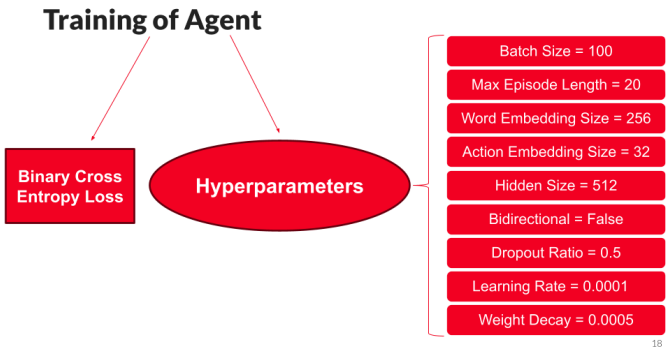

Fig. 3. Training Agent Schema [2]

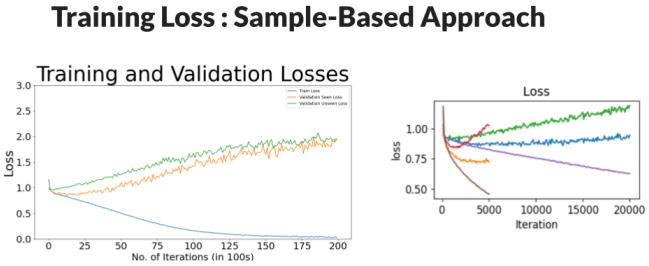

Fig. 4. Training Loss Comparison: Sample Based Approach [2]

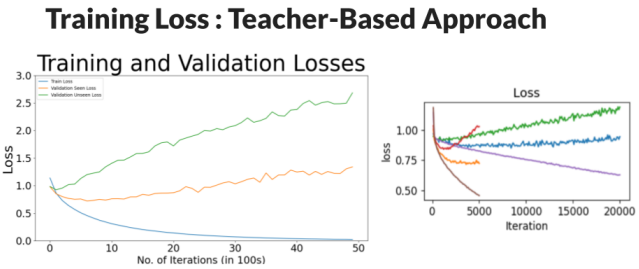

Fig. 5. Training Loss Comparison: Teacher Based Approach [2]

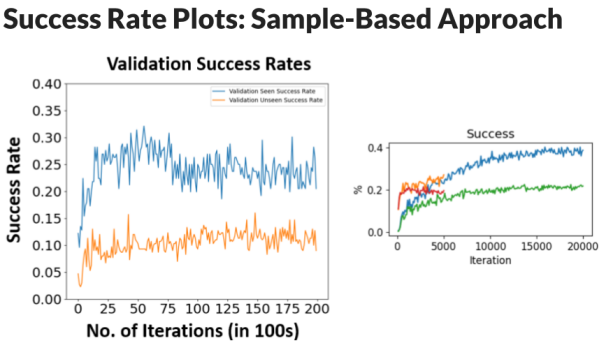

Fig. 6. Success Rate Comparison: Sample Based Approach [2]

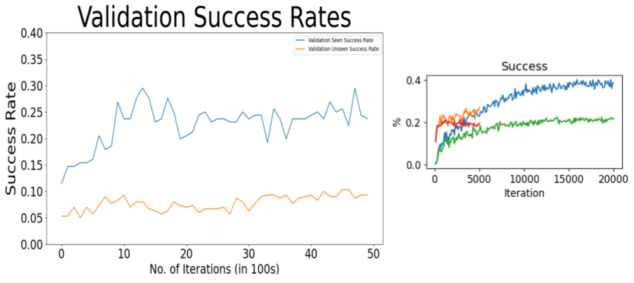

Fig. 7. Success Rate Comparison: Teacher Based Approach [2]

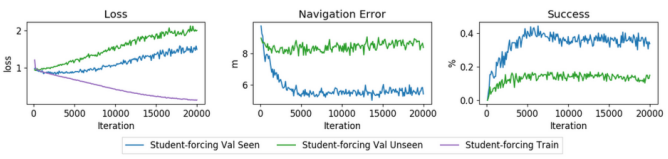

Fig. 8. Hyperparameter: Learning Rate

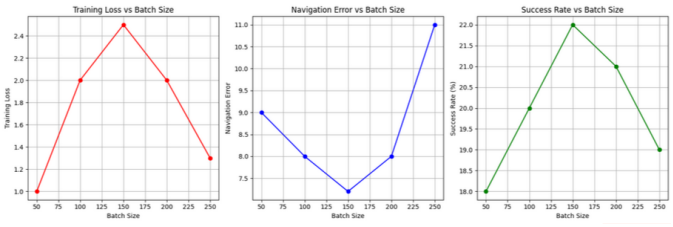

Fig. 9. Hyperparameter: Batch Size

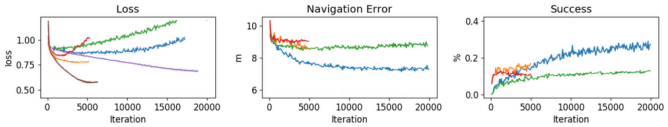

Fig. 10. Hyperparameter: Weight Decay

REFERENCES

[1] Matterport3D. "Learning from RGB-D Data in Indoor Environments." Accessed Oct. 23, 2023. [Online]. Available: https://niessner.github.io/Matterport/.

[2] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. V. D. Hengel, "Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments," ArXiv, 2017. [Online]. Available: https://arxiv.org/abs/1711.07280.

[3] "Matterport3D Scans." Accessed at: https://kaldir.vc.in.tum.de/matterport/v1/scans.txt.

[4] P. Anderson, "MatterPort3D Simulator," Github Repository, 2021. [Online]. Available: https://github.com/peteanderson80/Matterport3DSimulator.

[5] T. Winograd, "Procedures as a Representation for Data in a Computer Program for Understanding Natural Language," Technical Report, Massachusetts Institute of Technology, 1971.

[6] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "VQA: Visual Question Answering," in ICCV, 2015, pp. 2, 4, 5.

[7] X. Chen, T.-Y. L. Hao Fang, R. Vedantam, S. Gupta, P. Dollar, and C. L. Zitnick, "Microsoft COCO Captions: Data Collection and Evaluation Server," arXiv preprint arXiv:1504.00325, 2015, pp. 2, 4.

[8] A. Das, S. Kottur, K. Gupta, A. Singh, D. Yadav, J. M. F. Moura, D. Parikh, and D. Batra, "Visual Dialog," in CVPR, 2017, pp. 2.

[9] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh, "Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering," in CVPR, 2017, pp. 2.

[10] H. Mei, M. Bansal, and M. R. Walter, "Listen, Attend, and Walk: Neural Mapping of Navigational Instructions to Action Sequences," arXiv preprint arXiv:1506.04089, 2015.

[11] S. A. Tellex, T. F. Kollar, S. R. Dickerson, M. R. Walter, A. Banerjee, S. Teller, and N. Roy, "Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation," 2011.

[12] J. Fasola and M. J. Mataric, "Using Semantic Fields to Model Dynamic Spatial Relations in a Robot Architecture for Natural Language Instruction of Service Robots," in Intelligent Robots and Systems (IROS), 2013, pp. 143–150.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in CVPR, 2016, pp. 6.

[14] M. Savva, A. X. Chang, A. Dosovitskiy, T. Funkhouser, and V. Koltun, "MINOS: Multimodal Indoor Simulator for Navigation in Complex Environments," arXiv:1712.03931, 2017.

[15] D. Gordon, A. Kembhavi, M. Rastegari, J. Redmon, D. Fox, and A. Farhadi, "IQA: Visual Question Answering in Interactive Environments," in CVPR, 2018.

[16] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, 1997, pp. 6.

[17] "Seq2Seq and Attention," Lena Voita's NLP Course, Accessed at: https://lena-voita.github.io/nlp$_course/seq2seq_and_attention.html$.

[18] M.-T. Luong, H. Pham, and C. D. Manning, "Effective Approaches to Attention-Based Neural Machine Translation," in EMNLP, 2014, pp. 6.