Language Pre-Training and Auxiliary Tasks for Vision and Language Navigation

by

SAUMYA BHATT

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2021

To

Knowledge,

Future,

And Life.

Acknowledgments

Firstly, I would like to thank my advisor, Prof. Manfred Huber, for all the support and guidance during my Master's. I can't imagine myself in this position without his support. I am grateful for all the feedback and the skills that I learned from him. He gave me the freedom and opportunity to explore and work on challenging problems. His wisdom and knowledge helped me pave my way towards a better future.

I would like to thank all my professors and the committee members, Prof. Farhad Kamangar and Prof. Deok Gun Park for being a part of my thesis journey and guiding me.

I would also like to thank my family members for their support and constant motivation in my master's journey.

December 31, 2021

Abstract

Language Pre-Training and Auxiliary Tasks for Vision and Language Navigation

SAUMYA BHATT, M.S. Computer Science
The University of Texas at Arlington, 2021

Supervising Professor: Dr. Manfred Huber

The Vision and Language Navigation task came to life from the idea that we can build a robot or an autonomous system that can be instructed in human language and that will navigate using the instructions given. For example, we tell the agent to "Go down past some room dividers toward a glass top desk and turn into the dining area. Wait next to the large glass dining table" and not only does it reach the goal state but it follows the instructions while navigating. With the current developments, this may not seem like a distant problem anymore and in recent years a number of systems have been developed that attempt to address this task.

To accomplish this task, the artificial agent must understand two modalities with which humans perceive the world, vision, and language, and then translate these into actions. While significant progress has been made in recent years to develop systems capable of performing this task, these systems still fail in a significant number of cases. To investigate reasons and potential ways to overcome

this, this thesis explores a few ways in which the navigation task with multiple modalities can be grounded and can be aligned temporally and visually.

This thesis analyzes the failures of the previously used Environment Drop method with Back translation and investigates what happens when pre-trained embeddings, as well as auxiliary tasks, are utilized with it. In particular, it proposes an augmentation to the architecture for the Vision and language Navigation task with pretrained language tokens and a navigator with reasoning to oversee the progress and to co-ground vision and language rather than to only use temporal attention mechanism. The underlying base architecture on which the modifications have been implemented has been a highly successful method and uses the Environment Drop method with Back translation. While results with the modified architecture and proposed improvements did not show a significant increase in the success rate of the chosen base architecture, the analysis of the results has provided valuable insights to help determine the direction of potential further research.

Table of Contents

List of Illustrations

List of Tables

Chapter 1

Introduction: The Vision and Language Navigation Problem

Artificial Intelligence has been used to understand and simulate human perceptions. After required training, the intelligent systems can perceive information based on patterns. The decision-making process is based to a large degree on the perceptions related to the input data patterns. In this thesis, I investigate one such application which connects vision and language. It has been in the spotlight recently for its fast developments and is commonly termed as "Vision and Language Navigation" or VLN.

Vision and language Navigation came to life from the idea that someday we might be able to give instructions to a robot to finish required tasks and it will be successfully done without any further human intervention. To make this possible we need the agent to understand the visual environment, understand human language, ground it in the real-world environment, and make the decisions to reach the required location. For a human, this task may not seem significant but for an artificial agent, there are a lot of nuances that are challenging and must be taken care of before this can be implemented successfully in the real world. There have been many implementations in this area, but it is still a while before it is perfected and can be implemented in previously unseen environments successfully.

In recent years, development in the field of machine translation has helped to produce increasingly better results that have translated into other domains. In this vein, improvements in machine translation models have been popularly used with vision and language to make decisions and found their entry into the Vision and Language Navigation domain. The real challenge in these models is grounding vision and language. However, this has proven to give good enough results in pre-explored environments but as we tend to move towards new environments the results get worse.

For simulating more than one human perception at once, the agent has to find a way to ground the different modalities so they can be correlated, usually requiring the agent to pre-learn using a large amount of data. This is the main reason why the current models for Vision and Language Navigation are data-hungry. Pre-training on large-scale data should thus be able to help the model to show better results on downstream tasks. However, pre-training is still a relatively unexplored area when it comes to Vision and Language Navigation.

One other but the related area which can potentially improve the results of Vision and Language Navigation is the introduction of auxiliary reasoning tasks. First applications of these have shown to give good results. The main reason to include the reasoning tasks is to obtain more continuous feedback from the agent

while it is navigating. This helps in grounding and overseeing the progress of the agent.

In this thesis, I propose to augment a common architecture for the Vision and Language Navigation task with pretrained language tokens and a navigator with reasoning to oversee the progress and to co-ground vision and language rather than only a temporal attention mechanism based on the [*Fengda Z. 2020*].

Chapter 2

Background and Related Work

Vision and Language Grounding

There has been a lot of work done to understand language in the environment, in synthetic as well as real environments. For example, image captioning, visual question answering, or generating images from the textual description, all serve as multimodal problems, where natural language is mapped to the objects in the images, or the images are mapped to the natural language. Here, I have focused on the VLN problem in a real-world environment which is a multimodal problem and requires the agent to continuously interact with the environment.

Vision and Language Navigation

There have been several early papers that talk about Navigating with Natural Language Instructions but only with the introduction of the Room-to-Room dataset [*Peter A. 2018*] has this navigation task become more popular and been made possible in real-world environments. They implemented a deep learning sequence-to-sequence architecture with an attention mechanism and used one of the popularly used Adam optimizers, [*Diederik P. 2015*]. In my work, I have used a different base architecture as a starting point but used the same dataset and Adam optimizer initially with weight decay.

With the introduction of the common VLN environment simulator, a number of approaches have emerged to address this task. Some of the works see this as a reinforcement learning problem like [*Xin W. 2018*] which applied model-based and model-free reinforcement learning techniques. An imitation learning aspect was added to training in works like [*Xin W. 2019*, *Khanh N. 2019*]. The imitation learning aspect helped the agent to learn in previously seen environments and reinforcement learning proved to work better in unseen environments. [*Daniel F. 2018*] applied the popularly known Speaker-Follower or the back translation method to the VLN task. It divides the architecture into two modules, Speaker and Follower. The Speaker learns to generate instructions for the routes and supports the follower while the follower learns to follow the instructions. There are other methods like [*Hao T. 2019*] that use similar base architecture but add additional components to try to address problems of overfitting to the set of environments in the training set. This particular architecture proposes an 'environmental dropout' method based on view and viewpoint-consistent masking of the visual features to increase visual diversity and make the results better in unseen environments. This architecture is better in terms of successes than most of the previous papers but still failed to co-ground vision and language while training the navigator. I work with this architecture and my proposed architecture differs from it in terms of training and pre-processing.

Some other works like [*Daniel F. 2018*, *Xin W. 2019*] have included pre-trained word embeddings but have not studied changes and analyzed them to get a better output or understand their impact on the dataset.

Most of the work tries to outgrow the bottleneck of the dataset and ground natural language in a real environment, but even after all the attempts, vision and language navigation have a lot of work to be done. Most of the previous work neglects the language semantics and works only on the given dataset without language pretraining. Most of the work also does not reason with the navigator during navigation to estimate the progress and the success. [*Fengda Z. 2020*] includes the reasoning aspect making the learning process better but neglects the language semantics. In contrast, in my work, I have included the pretrained language tokens and included a navigator with reasoning to oversee the progress and co-ground vision and Language rather than only a temporal attention mechanism.

Vision and Language Navigation Data Requirements

For a single modality task, we require a huge amount of data. For example, the popularly used [*ImageNet dataset*] for image recognition tasks contains hundreds and thousands of images for each label. This makes the image recognition task much easier as the dataset contains a lot of variety and there is very little chance of overfitting. The requirements for a multimodal task like Vision and Language

Navigation are far more complex. The data must be varied in terms of visual environments and text and must be aligned with each other. Moreover, the data here also has a temporal aspect as it must align with a corresponding action sequence, further increasing the complexity of the required data.

The dataset that I have used in my work is huge but still lacks in terms of variability and the amount of data needed to directly be used without any pretraining on images or texts. One of the major disadvantages is that even if we use pre-trained word embeddings or image features and let it be fine-tuned by the current dataset, it might overfit the training environment and it becomes harder to get good results in an unseen environment even after pretraining. This is the main reason that a lot of researchers are working towards pretraining the language and image tokens with vision and language navigation tasks along with the main decision-making architecture. We try to observe the change and evaluate our results by using pre-trained GloVe embeddings, taking one small step towards pretraining, and hopefully achieving better results.

Chapter 3

Base Architecture

The work presented in this thesis builds largely around the base speaker-follower architecture used in [*Hao T. 2019*], adding pre-training and auxiliary tasks to investigate their benefits. His chapter introduces the main components and methods used in this base architecture before the following chapters provide some analysis of this architecture's failure conditions and introduce and evaluate the proposed modifications.

Back Translation in Vision and Language Navigation

Overview

The back-translation method is popularly used in machine translation to find how accurate the translation is from the source to the target text. It translates the complete translation to the original language and then compares it with the source text. Back translation for vision and language navigation is termed by some of the previous works like [*Daniel F. 2018*] as the "speaker-follower model". Like machine translation, in this model, the speaker is the instruction generator that learns to generate alternate instructions for the given instruction-path pairs and the follower is the navigator who learns to follow the instructions as shown in Figure 1 and Figure 2. The overall architecture is shown in Figure 3.

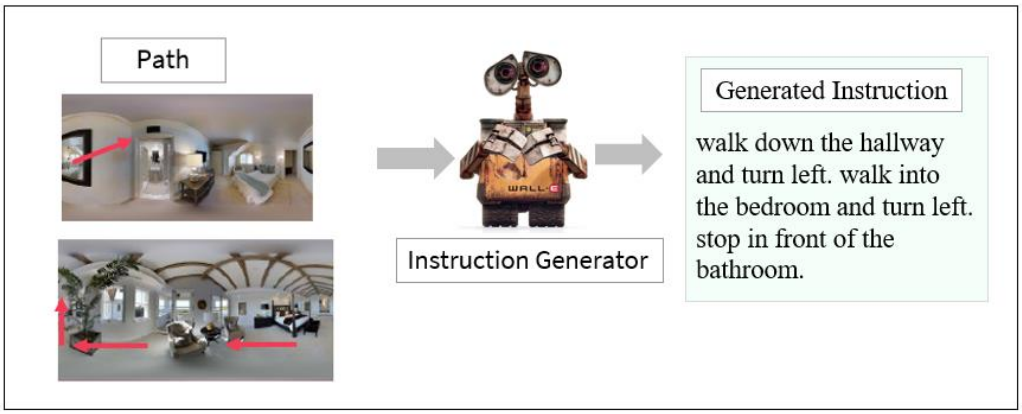

Figure 1: Instruction Generator

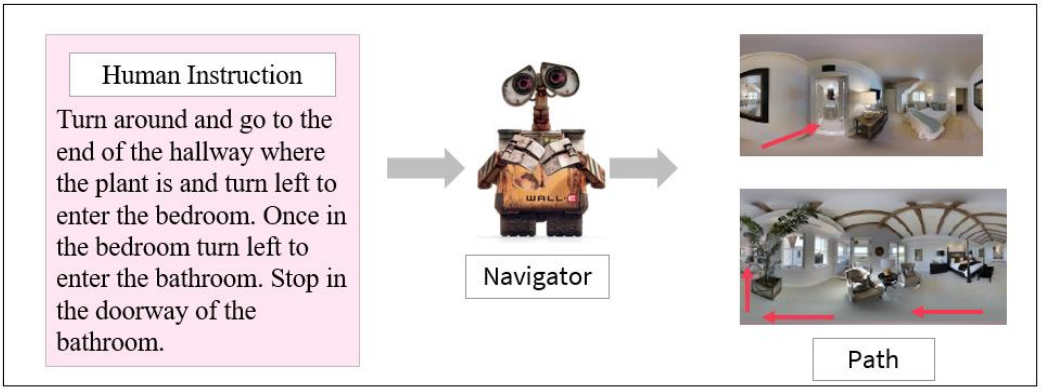

Figure 2: Navigator

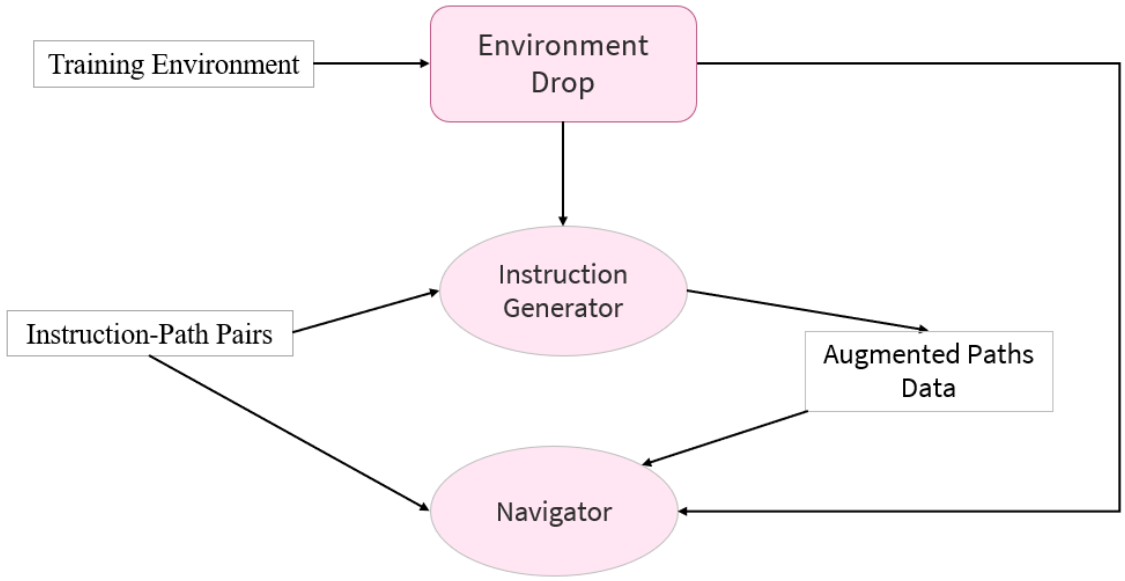

Figure 3: Architecture

Data Augmentation

There are only a limited number of training instruction-path pairs at present in the data, so generating more instruction-path pairs in the given environment should result in a better learning process. The instructions are generated by the instruction generator for a set of sampled paths and are then used in the architecture used in this thesis to train the Navigator separately.

Instruction Generator

I have used a sequence-to-sequence model which generates a distribution of instructions (i) given the path (p) for the speaker. Instead of a simple sequence-to-sequence model I have used a Bidirectional-LSTM encoder with attention over visual information, which encodes the visual observations, and a decoder with attention over encoded input route which generates instructions word-by-word. Figure 4 shows the final Instruction Generator Architecture similar to [*Hao T. 2019*].

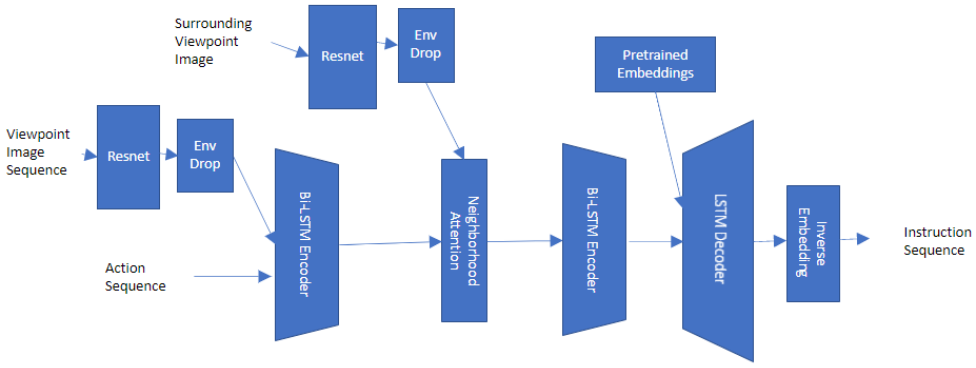

Figure 4: Instruction Generator Architecture

Navigator

The navigator or the agent uses a sequence-to-sequence model which generates a distribution of paths (p) given the instructions (i). A bidirectional encoder and LSTM decoder with attention over the features is used in the modified architecture sued here. To make sure that the agent knows the previous steps, we have added the instruction aware-hidden vector, instead of a hidden vector in the same way as [*Hao T. 2019*]. The final architecture for the baseline navigator is as shown in Figure 5.

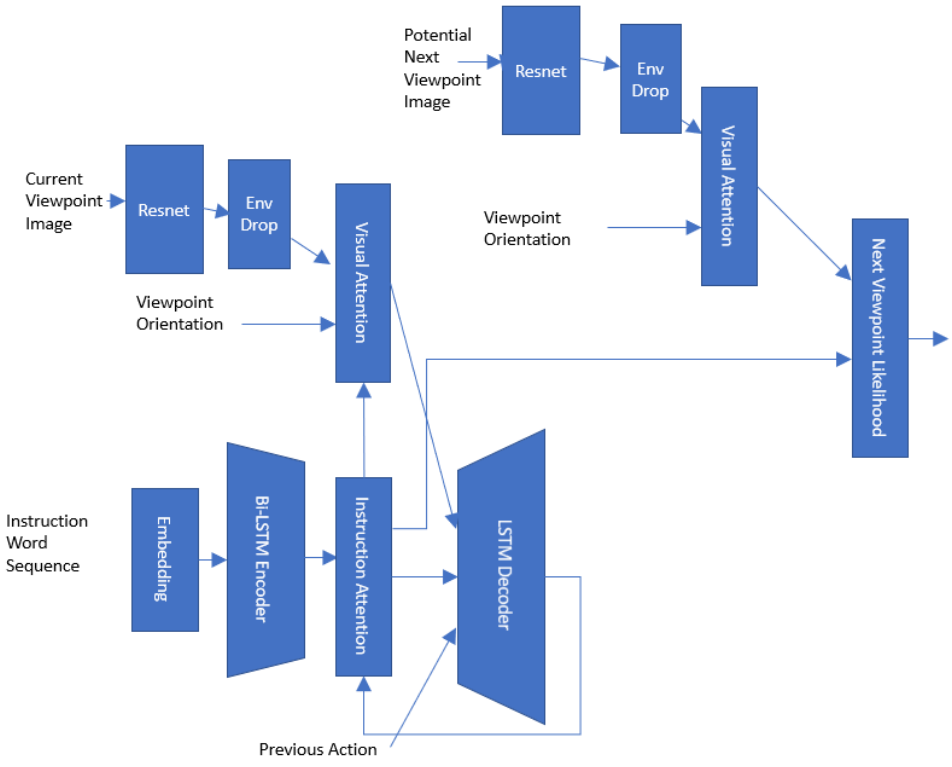

Figure 5: Navigator Architecture

Self-Supervised imitation Learning + Reinforcement Learning

Imitation learning is an off-policy learning method that utilizes given behavior sequences to train a policy that will reproduce them. The agent learns to imitate the behavior of the teacher. In this case, rather than using a pre-computed set of teacher examples, the next demonstrated navigation action is decided by calculating the shortest distance to the target, and the next navigable viewpoint on this path is selected. This is termed as teacher action. To achieve imitation, the actor is trained with a loss function that is the negative probability of the teacher's action.

To make sure that the teacher's action leads to the goal, an on-policy reinforcement learning agent is used along with the above. It adds a reward of +3 if the agent stops within 3m of the goal state and it adds -3 if the result is unsuccessful. This strategy used in [*Hao T. 2019*] works better than the previous works which largely relied on one learning approach but still leaves a lot of chance for improvement. The results show that this method has proven to be more effective than the Extrinsic and intrinsic Reward function used by [*Xin W. 2019*] but the Imitation Learning method only focuses on the nearest navigable viewpoint and the on-policy Reinforcement Learning only focuses on the results, rather than the progress in the trajectory or the relative position of the navigator with respect to natural language instructions.

Environmental Dropout method

[*Hao T. 2019*] attempted to solve the problem of overfitting to known environments and the resulting issues with navigating in an unseen environment with a simple dropout method. This method randomly drops out the raw features from the environment in the VLN method. It is also added to remove image features traditionally. The goal of this method is to increase the variety of the visual information to represent visually more diverse environments.

shows a broad overview of the base architecture with environment dropout where the dropout and Instruction Generator is used to increase the size and diversity of the instruction-path pair data available to train the Navigator to be better able to address previously unseen environments. In this architecture, both the environment dropout and the Instruction Generator are trained first using the environment data and the initial instruction-path pairs. Once trained, they are then used to help train the Navigator to address the VLN task.

Chapter 4

Analysis: Base Architecture

To investigate the potential for improvements to the existing base architecture, an analysis of failure conditions in both seen and previously unseen environments were performed. A failure as defined in the Room-to-Room dataset/simulator is a path execution that does not end within a predefined distance of the target location. Similarly, an oracle failure is any path that does not pass within that predefined distance of the target, thus corresponding to the situation where an oracle for stopping upon completion was available. As oracle success rates and success rates were virtually identical for this architecture, the analysis here focuses on path failures.

Failure Analysis

Overall failure rates for the base architecture are shown in Figure 6 for training, previously seen, and previously unseen environments. This data shows a strong tendency to overfit the training data as demonstrated by the failure rate of 4.69% vs a failure rate of 35.16% on evaluation paths even in the same environments contained in the training set. This failure rate further increases to 51.68% when applied to previously unseen environments.

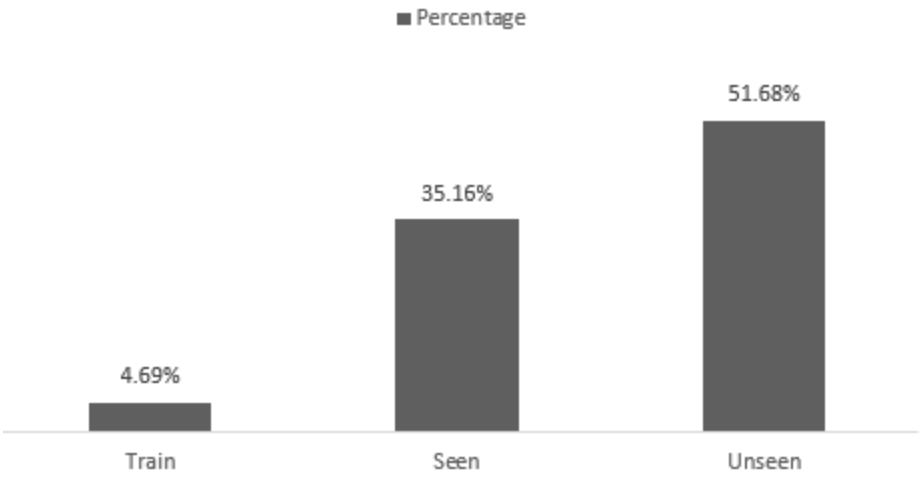

Figure 6: Failures in the Training, Seen Test, and Unseen Test Cases

This overfitting suggests that the system over-emphasizes the vocabulary and sentences in the training set over the broader language and visual feature set. Moreover, the difference between seen and unseen environments suggests some degree of attenuation to the specifics of the trained environments despite the environment dropout components. All this suggests that training the language and image feature used in a broader context could be useful.

To obtain more information regarding the reasons for failure, in particular, whether recognizing the intermediate or goal locations or looping represent major challenges for this architecture, a more detailed analysis of navigation errors in terms of distance from the goal reached and path length and path type in the failed cases was performed.

It was found that continuous looping does not occur for any trajectories. Some trajectories do form loops, but they do not continue looping as they move forward.

Navigation Errors and Path Lengths

Below is the histogram of the navigation errors whose path lengths are more than the shortest length and for the ones which are less than the shortest length.

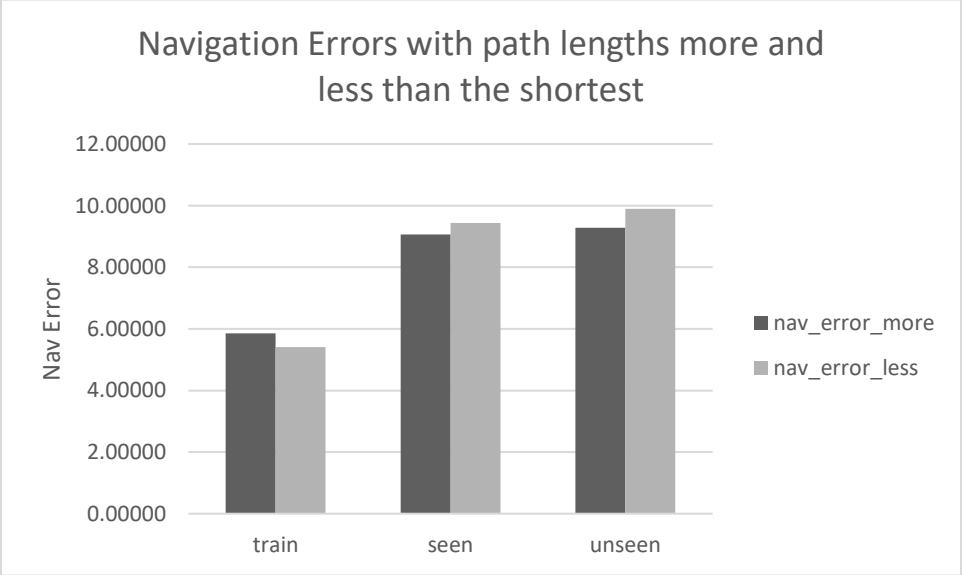

Figure 7: Navigation Errors whose paths are more and less than the shortest path

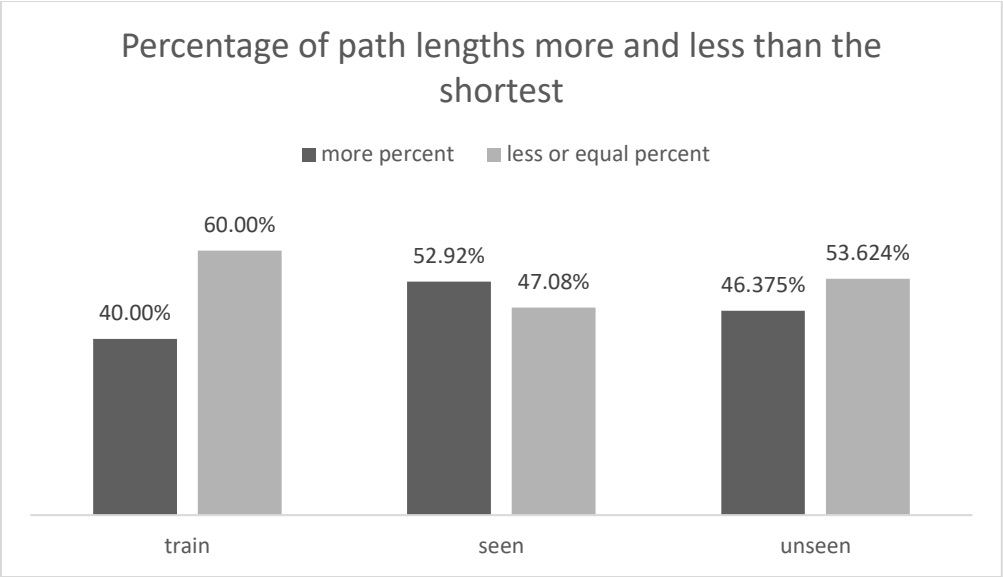

Figure 8: Percentage of Path more and less than the shortest path

From the above graphs, we can see that there is no trend for the path lengths of the failed cases when compared with the shortest path length.

Training Data

9 shows a histogram of the path errors, i.e. the distance to the target location reached, of failed navigation tasks in the training data.

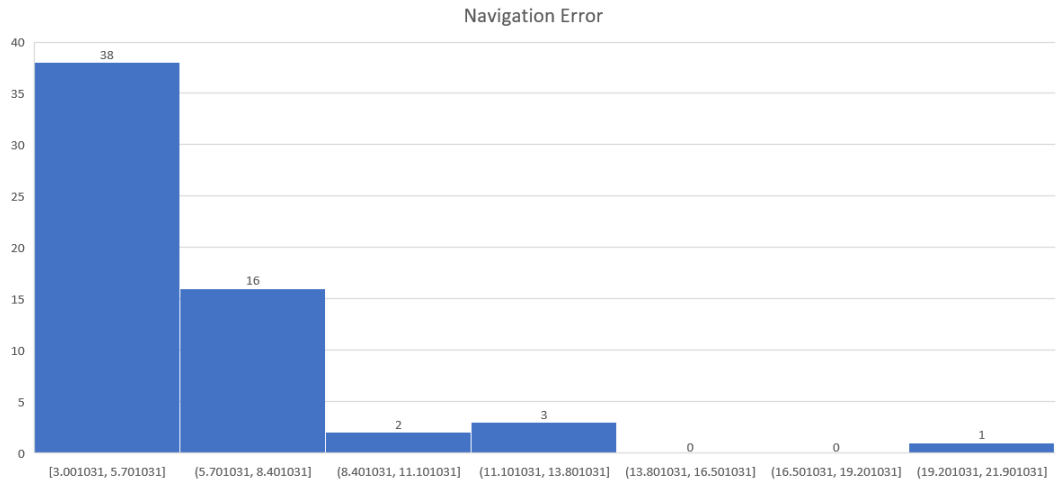

Figure 9: Navigation Errors in the Training Set

In the training data, most of the path errors are between 3 to 5.7 m. This shows that even while training the vision-language grounding is a problem. The navigator comes close to the goal state even in most failed paths but does not identify the goal. One possible reason could there be that the navigator is not able to correlate the word vectors with the features in the environments.

Seen Data

10 shows the same path error analysis for failed paths in new examples in the same environments as used in the training set (i.e. in the same room arrangements with the same visual features and objects).

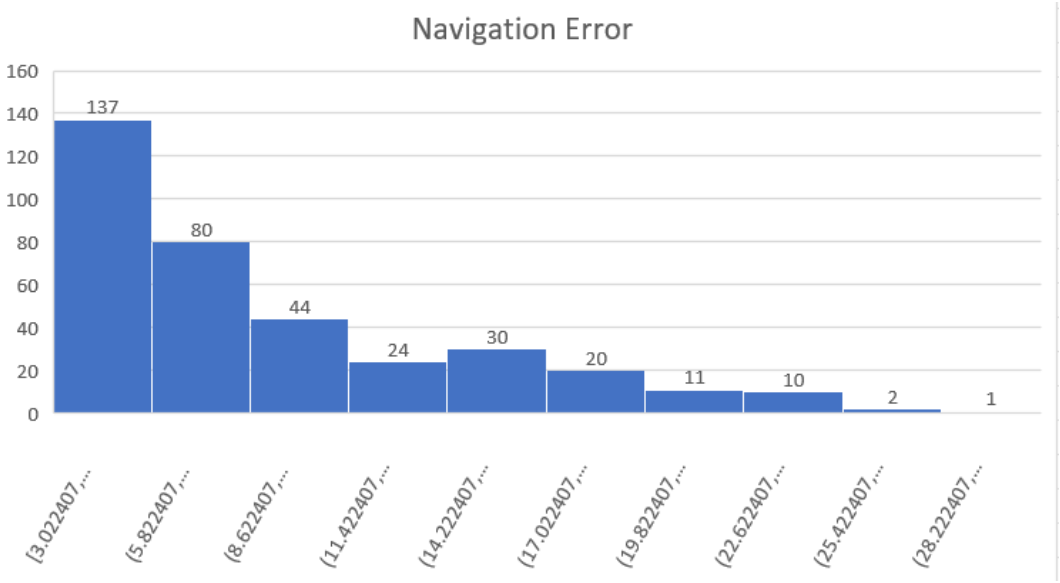

Figure 10: Nav Error Seen

Almost, 40% of the data has a higher navigation error than 8.5 meters, which is significant. This means the agent is not able to match the language to the visual environment correctly. This could be due to a number of reasons like Vision-Language grounding and the progress of the navigation instruction is not taken into consideration with respect to the environment.

Most of the instructions have low navigation errors. The number of paths that do not get to within a few meters, however, is significantly larger than for the training data. That means that while the agent again comes close to the goal state but is not able to identify the goal frequently, there are also conditions where it does not get close enough to be near the relevant features.

Unseen Data

11 shows the path error analysis for failed paths in the Unseen test set, i.e. in new paths that are executed in previously unseen environments.

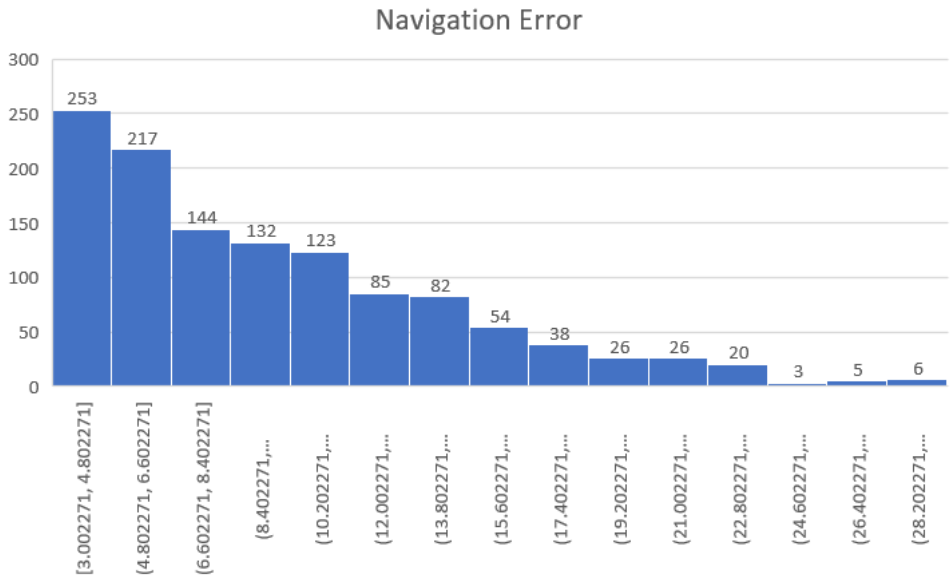

Figure 11: Navigation Errors in the Unseen Test Set

This has the highest number of failure cases and again, the highest number of failure conditions have low navigation errors. However, the distribution has a much longer tail, also containing a reasonable percentage of example paths where the system does not get close to the target and often is more than a room away. That means about half the time the agent still comes close to the goal state but is not able to identify the goal, while the other half of the time it is not able to get close to the goal.

Comparison

Figure 12 shows the development of navigation error through the training process for training, seen, and unseen data sets.

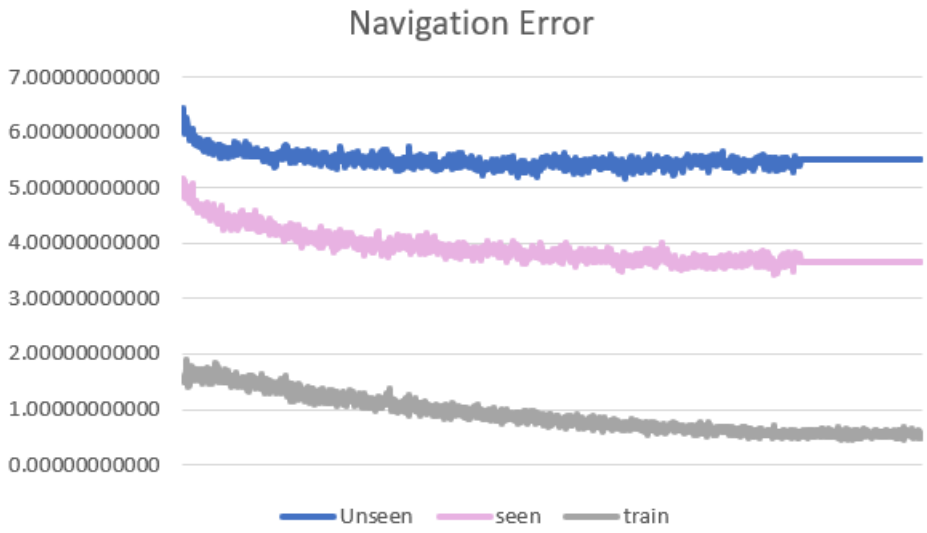

Figure 12: Nav Error Comparison

These graphs show a significant difference between the three conditions throughout the entire training process with the largest gap between training and test data and an increasing gap between seen and unseen test sets. The former indicates significant overfitting to the language used in the training set while the latter suggests a growing adaptation to the features in the training environment. From Figure 12 we can see that after a given point the model stops improving. Considering the complexity of the Vision and Language Navigation problem we can say that for an improved dataset this problem may occur at a much later point.

Success Rate

The plot of success rates during training in similarly shows a significant gap between training and test conditions and an increasing gap between unseen and seen test cases.

Figure 13: Success Rate Comparison

Above, when we observe Figure 12 and Figure 13, there is a huge gap between the success rate of seen data and training data. This gap is wider than the one between validation seen and unseen data. This means even after training in the same environment the agent is not able to match all the features with the word embeddings.

Chapter 5

Proposed Improvements

Overview

Based on the failure analysis in the previous chapter, two main architecture modifications are proposed here and evaluated in the next chapter to attempt to address the overfitting of the language components through a broader word embedding task and to achieve a broader relationship between language and vision through the inclusion of auxiliary reasoning tasks. The goal of these proposed modifications in the present architecture is to allow pretraining or co-training with additional tasks and datasets to make sure that the vision-language task is aligned temporally and spatially and is grounded.

In the previous architecture, the language semantics are not used while training. Here, we include language semantics along with the visual features through the use of pre-trained word embeddings that capture broader language semantics. The second modification, the inclusion of auxiliary tasks during Navigator training by contrast is aimed at providing more continuous feedback to the learner to allow for a closer correlation of language, vision, and action throughout the training process.

Pre-trained Word Embeddings

We can observe that most of the previous work disregard the language semantics when training the data. One component of the language semantics in terms of word similarities is encoded in the word embedding used as part of the Instruction Generator component of the base architecture. Hence, to understand the effect of language semantics better on the Instruction Generator as well as on the Navigator, we use pre-trained GloVe embeddings [*Jeffrey P. 2014*]. I have used GloVe embeddings with 300 dimensions trained from the Common Crawl dataset (42B tokens, 1.9M vocab, uncased, 300d vectors). I have fine-tuned the word vectors later in the training environment.

Auxiliary Reasoning Tasks

To take the feedback from the navigator continuously and to make the training process more grounded I have adapted the Auxiliary Reasoning training method from [*Fengda Z. 2020*]. The main difference is that I have focused on the temporal and spatial features rather than orientation. The goal of this method is to achieve a tighter coupling between language, vision, and action by incorporating additional training tasks that are related to the overall task but can be evaluated at each time step rather than only at the end of a navigation trajectory. Inclusion of these tasks should ideally accelerate training and prevent overfitting by requiring feature spaces to be sufficient not only for the navigation task but also for the

auxiliary tasks uses. Here I have used the Progress Estimation and Cross-modal matching tasks along with the teacher forcing method to train the Instruction Generator. The main difference between my method and the previously mentioned is that my method uses pre-trained embeddings and the Environment Dropout method along with the previously mentioned task, and I do not include the Angle Predictor here as this gives us a better idea of training with the image and word features rather than the orientation. Figure 14 shows the architecture of the Navigator after adding the Auxiliary Tasks which are similar to [*Fengda Z. 2020*].

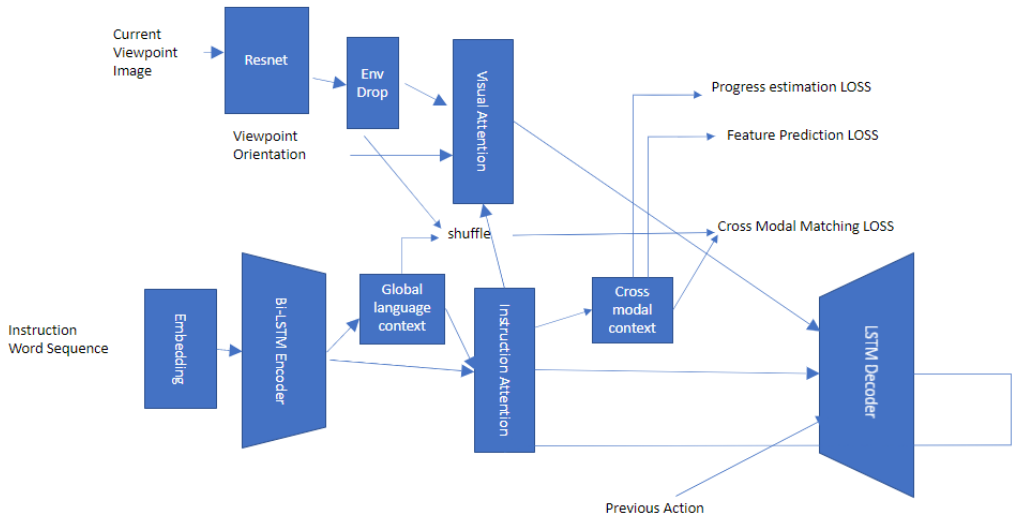

Figure 14: Proposed Architecture with Pretrained Embeddings and Auxiliary Tasks

Progress Estimator

This task is used to train the navigator instead of just using imitation learning. This auxiliary task aims at predicting the amount of progress made in completing the navigation task at each point in the trajectory. Here, Cross Modal context and the percentage of steps is used to represent the progress and converted into an auxiliary loss function that is added to the overall loss during training. Cross modal context is nothing but the attended language feature similar to the one used in [*Fengda Z. 2020*]. This task helps in vision language grounding by more tightly correlating a part of the language description to the visual input at the particular stage in the trajectory.

Cross-Modal Matching

This task uses the hidden language feature vector of the navigator encoder and the attended vision-language features to calculate the loss. It tries to make sure that the trajectory and the instruction match each other by trying to align the historical vision-language features. It is calculated by shuffling the feature vector and then supervising the prediction results similar to [*Fengda Z. 2020*].

Feature Predictor

A third auxiliary task added here is aimed at aligning a broader set of visual features with the state of the navigation task by utilizing predictions of visual features from multiple viewpoints in an additional loss function term. This takes

into consideration the angle of the teacher action in the imitation learning and reinforcement learning task and encoded feature of the navigator. This function gives information about the semantics of the room and the relationship of the visual features between different viewpoints.

Chapter 6

Experiment and Implementation Detail

Dataset

I have evaluated the proposed model using the Room-to-Room (R2R) environment dataset based on the [*Matterport3D*] simulator. It contains 90 different housing environments. Using Matterport3D region annotations 7,189 paths are sampled, capturing most of the diverse surroundings in the dataset. The navigation instructions have been collected via Amazon Mechanical Turk. It is split into train set, validation seen set, and validation unseen set. The training set contains 61 environments with 14,025 instructions, in the same environment Validation seen data contains 1,020 instructions, and validation unseen data is set in 11 new environments with 2,349 instructions.

Implementation Details

I have used popularly used pretrained visual feature vectors which are generated from the final convolutional layer of ResNet [*Kaiming H. 2015*] trained on the ImageNet [*Olga R. 2015*] classification dataset. I have used GloVe vectors for pre-trained word embeddings for text to be fine-tuned after freezing them for a number of initial iterations. We have used the results of a single run for evaluation and analysis and we do not use pre-exploration for our experiments.

I have included Back translation, Environment Drop, Data Augmentation, and Imitation Learning, and Reinforcement Learning for my initial experiments. I use this as the Base Architecture of the Experiments to compare against the modified architecture proposed here. As these modifications affect both the Instruction Generator (through the word embeddings) and the Navigator, both of these are trained, and evaluation is performed on both of them in order to determine the effects of the different modifications and their interactions. To evaluate the use of pre-trained word embeddings to incorporate more language semantics, a two-stage experiment without Auxiliary Tasks was performed as follows:

1. I have first pre-trained the Instruction Generator which also allows us the gather the Augmented data for paths. I have allowed the pre-trained GloVe embeddings to be frozen for the first 25,000 iterations and then let it be fine-tuned in the Matterport3D environment till 80,000 iterations. After this, the Instruction Generator is evaluated in terms of the language properties of the generated instructions.
2. Then the Navigator is trained with the augmented dataset from the Instruction Generator using the mixture of imitation and reinforcement learning with Environment Dropout. The same model is then fine-tuned by the augmented data with environment dropout.

To investigate the effect of auxiliary tasks, a second experiment is performed. Here, I have added Auxiliary Reasoning Tasks for training the Navigator and observed the results. Again this involves a two-stage training process as follows:

1. I have first pre-trained the Agent with the Instruction Generator which also allows us the gather the Augmented data for paths. As before, I have allowed the pre-trained GloVe embeddings to be frozen for the first 25,000 iterations and then let it be fine-tuned in the Matterport3D environment till 80,000 iterations.
2. Then the Navigator is trained using the auxiliary reasoning tasks, along with a mixture of imitation and reinforcement learning with Environment Dropout. The same model is then fine-tuned by the augmented data with environment dropout.

Evaluation Metrics

To evaluate the Instruction Generator, we use Bleu1, Bleu2, Bleu3, Bleu4, and Bleu Score for the evaluation [*Kishore P. 2002*]. These metrics are aimed at capturing semantic similarity between the generated instructions and the original ground truth instructions stored with the original data.

For the evaluation of our final data, the primary evaluation metrics are the Success Rate (SR): it measures the rate at which the agent successfully reaches the

goal, the trajectory is considered to be a success if the navigation error is less than 3m.; and the Navigation Error (NE): it measures the average distance of the shortest path from the last position in the selected path to the goal position. We also consider Path Length (PL) while analyzing the data. In addition to these metrics, we also try to observe differences based on the different loss values.

Chapter 7

Analysis: Pre-trained Word Embeddings

To investigate the influence of broader language semantics through pre-trained word embeddings on the language descriptions generated the Bleu scores of generated language instructions of the Instruction Generator are evaluated using different Bleu scores. After adding pre-trained word embeddings, I have fine-tuned it with the given dataset.

5 shows the scores for the Bleu 1 – Bleu 4 metrics with and without pre-training on generated instructions for new paths in environments from the training set. Similarly, 6 shows the scores for unseen environments.

Figure 15: Bleu Values of Generated Instructions Before and After Use of GloVe Pre-Trained Word Embeddings on Previously Seen Environment Data

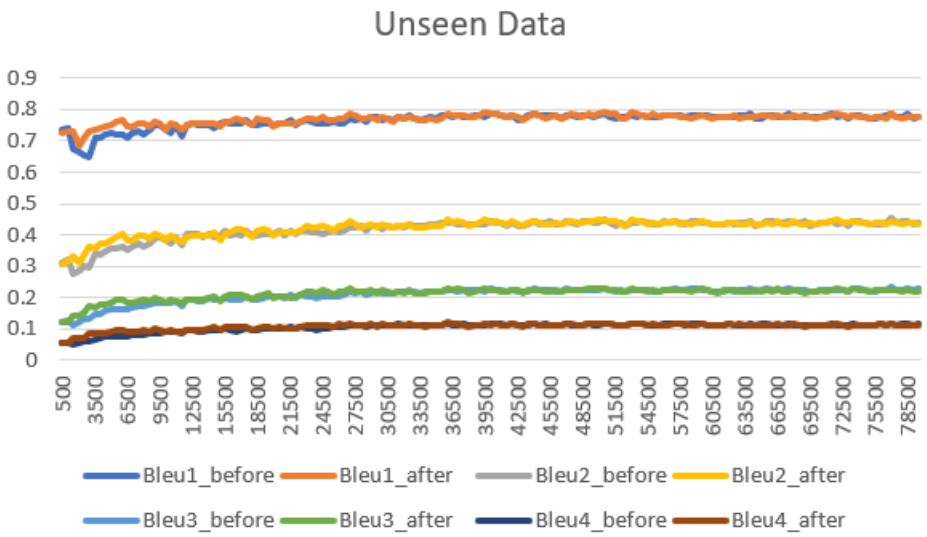

Figure 16: Bleu Values of Generated Instructions Before and After Use of GloVe Pre-Trained Word Embeddings on Previously Unseen Environment Data

With these graphs, we observe that while the pre-trained models initially achieve higher Bleu scores, there is not much difference in the output of the Instruction Generator once completely trained even after using the pretrained graphs. Similar can be observed in a more detailed view of the combined Bleu value for the case of previously unseen environments in 7.

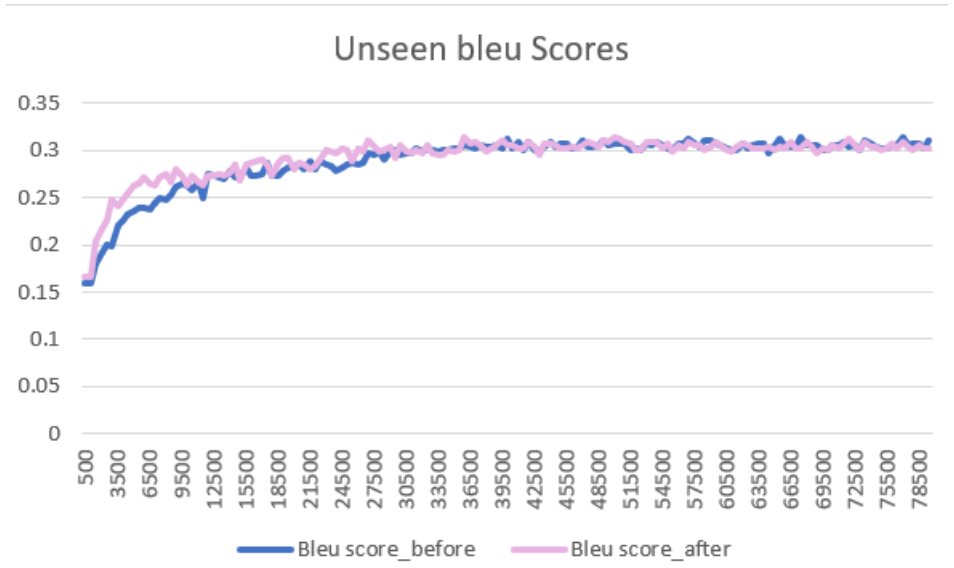

Figure 17: Bleu Values for Generated Instructions Before and After Use of Pre-Trained GloVe Word Embeddings for Previously Unseen Environment Data

This figure shows more clearly the initial benefit of the pre-trained embedding in terms of the quality of the generated instruction semantics. However, this benefit seems to reduce even before fine-tuning after the first 25,000 iterations and seems to disappear completely during fine-tuning. Note, however, that this does not necessarily imply that the generated instructions are not more diverse but rather only that they are not more semantically similar to the ground truth instructions than the ones without pre-training. To fully evaluate this, its effect on the Navigator has to be evaluated in the next section.

Chapter 8

Results and Conclusions

To evaluate the ultimate benefit of the modifications, the overall performance of the Navigator was evaluated with and without the modifications. The results show that in our original baseline model if we use pre-trained word embeddings, they behave almost the same way if we let it fine-tune on the model even on the later stage.

The base model has Back Translation (BT), Environment Drop (ED), Imitation Learning (IL), Reinforcement Learning (RL). After adding Auxiliary Reasoning Tasks (AuxR) and pretrained embeddings to our previous model we do not see many improvements in the final results. The final results can be seen in Table 1.

	Train		Seen		Unseen	
Matrices	SR	NE	SR	NE	SR	NE
Base Model	96.4%	0.20	64.8%	5.02	50%	9.92
Base +Pre	96%	0.3	64%	5.0	50%	9.9
Base + AuxR	94.7%	0.37	60.6%	6.4	48.6%	10.6

Table 1: Results in Terms of Performance of the Navigator

The architecture with Environment Drop overfits the limited available feature at a much later stage as it was initially included as a solution for overfitting problem and improving with the same base architecture is harder. At the same time, the architecture fails to identify the features in seen data even though it has been trained in the same environment. It is not able to identify the visual features with the textual information resulting in a huge gap between training and seen results.

Environment drop, even though it by itself has benefits for unseen environments, makes it more difficult to apply modifications to improve the architecture with Environment Drop as it generates visual features that were not present in any actual environment and might yield misleading training data with inconsistent language and vision feature relations.

To find a solution to the overfitting problem, Data Augmentation and increasing the actual data with a better variation in it will surely help the results. I have analyzed the Instructor Generator in detail and have found that using pre-trained embeddings which are allowed to be fine-tuned even at a later stage of training do not change the outputs of the Instructor Generator or the Navigator in a significant manner. Pre-training the textual embeddings and the image features with relevant datasets and making sure that they do not overfit the data by early stopping or keeping the weights constant may help the results.

Adding Auxiliary Reasoning Tasks does not make the success rate of our Base Architecture better. Potential improvements can be made by removing Environment Dropout from the base architecture when using the Auxiliary Reasoning Tasks or by utilizing ony trials without environment drop in the learning trials for the Auxiliary Reasoning loss functions. There is still a lot of work to be done with Auxiliary Reasoning Tasks. It calls for further study and analysis of the results.

Chapter 9

Conclusion and Future Work

This thesis studied failure conditions for the commonly used speaker-follower architecture in the Vision and Language Navigation domain and proposes two modifications to attempt to achieve a stronger integration of language semantics through pre-trained word embeddings and coupling between language, vision, and action spaces through the use of auxiliary tasks. While the results using the modifications on the base architecture do not show significant improvements and auxiliary tasks even incurred a performance drop, they give some insight into the benefits and suggest a number of additional modifications that might provide the benefit sought. Here, we can conclude that for increased improvement in the results, removing the Environment Drop from the architecture will potentially be beneficial when using the loss with Auxiliary Reasoning Tasks.

There is still a lot of work to be done with Auxiliary Reasoning Tasks. It calls for further study and analysis.

The Vision and Language Navigation problem can be approached with more datasets in the future and increased pretraining with relevant data so that the grounding of vision-language is better possible.

References

[Peter A. 2018] Vision-and-Language Navigation: Interpreting Visually Grounded Navigation Instructions in Real Environments.
Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, Anton van den Hengel CVPR, 2018

[Xin W. 2018] Look Before You Leap: Bridging Model-Free and Model-Based Reinforcement Learning for Planned-Ahead Vision-and-Language Navigation
Xin Wang, Wenhan Xiong, Hongmin Wang, William Yang Wang ECCV, 2018

[Daniel F. 2018] Speaker-Follower Models for Vision-and-Language Navigation
Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, Trevor Darrell NeurIPS, 2018

[Xin W. 2019] Reinforced Cross-Modal Matching and Self-Supervised Imitation Learning for Vision-Language Navigation
Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, Lei Zhang CVPR, 2019

[Hao T. 2019] Learning to Navigate Unseen Environments: Back Translation with Environmental Dropout
Hao Tan, Licheng Yu, Mohit Bansal NAACL, 2019

[Chih-Yao M. 2019] The Regretful Agent: Heuristic-Aided Navigation through Progress Estimation
Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, Zsolt Kira CVPR, 2019

[Liyiming K. 2019] Tactical Rewind: Self-Correction via Backtracking in Vision-and-Language Navigation

Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, Siddhartha Srinivasa CVPR, 2019

[Fengda Z. 2020] Vision-Language Navigation with Self-Supervised Auxiliary Reasoning Tasks

Fengda Zhu, Yi Zhu, Xiaojun Chang, Xiaodan Liang CVPR, 2020

[Vihan J. 2019] Stay on the Path: Instruction Fidelity in Vision-and-Language Navigation

Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, Jason Baldridge ACL, 2019

[Diederik P. 2015] ADAM: A method for Stochastic Optimization

Diederik P. Kingma, Jimmy Lei Ba ICLR, 2015

[Khanh N. 2019] Vision-based Navigation with Language-based Assistance via Imitation Learning with Indirect Intervention

Khanh Nguyen, Debadeepta Dey, Chris Brockett, Bill Dolan CVPR, 2019

[Nitish S. 2014] Dropout: A Simple Way to Prevent Neural Networks from Overfitting

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, 2014

[Jeffrey P. 2014] GloVe: Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, Christopher D. Manning 2014

[Kaiming H. 2016] Deep Residual Learning for Image Recognition

Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun CVPR, 2016

[Olga R. 2015] ImageNet Large Scale Visual Recognition Challenge

Olga Russakovsky · Jia Deng · Hao Su · Jonathan Krause · Sanjeev Satheesh · Sean Ma · Zhiheng Huang · Andrej Karpathy · Aditya Khosla · Michael Bernstein · Alexander C. Berg · Li Fei-Fei IJCV, 2015

[Kishore P. 2002] BLEU: A Method for Automatic Evaluation of Machine Translation

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, 2002

[Angel C. 2017] Matterport3D: Learning from RGB-D Data in Indoor Environments

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, Yinda Zhang, 2017

[ImageNet dataset] https://www.image-net.org/

Biography

Saumya has done her Master's in Computer Science at the University of Texas at Arlington. Her research work is around Natural Language Processing, Deep Learning, Reinforcement Learning, and Machine Translation. Her thesis explores a Multi-modal problem termed Vision and Language Navigation.

During her master's journey along with knowledge, she gained experience working with applications like prediction problems i.e. weather prediction, recommender systems like movie recommendation, Machine Translation problems, Entity Recognition Problems, Multi-modal Problems like Vision and Language Navigation, and worked with a huge amount of data.

She is an innovator and a researcher at heart and believes in learning every day. She loves to work with challenging problems in different domains.

Her area of interest is wide and includes Deep Learning, Natural Language Processing, Machine Learning, Data Science, and Data Analysis. Along with a passion for Artificial Intelligence and Machine Learning applications, she loves to work with data and generate insights. She is not just her knowledge and experience, but she is her ideas, positivity, relentlessness, and much more.