MineObserver 2.0: A Deep Learning & In-Game Framework for Assessing Natural Language Descriptions of Minecraft Imagery

Jay Mahajan, Samuel Hum, Jack Henhapl, Diya Yunus, Matthew Gadbury, Emi Brown, Jeff Ginger, H. Chad Lane

University of Illinois - Urbana Champaign
{ jaym2, hum3, jackah2, dyunus2, gadbury2, ecbrown2, ginger, hclane}@illinois.edu

Abstract

MineObserver 2.0 is an AI framework that uses Computer Vision and Natural Language Processing for assessing the accuracy of learner-generated descriptions of Minecraft images that include some scientifically relevant content. The system automatically assesses the accuracy of participant observations, written in natural language, made during science learning activities that take place in Minecraft. We demonstrate our system working in real-time and describe a teacher support dashboard to showcase observations, both of which advance our previous work. We present the results of a study showing that MineObserver 2.0 improves over its predecessor both in perceived accuracy of the system's generated descriptions as well as in usefulness of the system's feedback. In future work we intend improve system-generated descriptions, give teachers more control and upgrade the system to perform continuous learning to more effectively and rapidly respond to novel observations made by learners.

Introduction

There are few challenges more important to educators and parents than helping children come to a better understanding of the world. Central to this goal is to promote their abilities to articulate what they see and observe about the natural world and to explore those observations through the lens of science (Arias and Davis 2016).

In previous work we presented a prototype AI system called *MineObserver* (Mahajan et al. 2022) that sought to assess learner observations in the popular game Minecraft. In this paper, we report on several significant updates to that system (MineObserver 2.0) that improve it along several critical aspects. In the context of video game research, MineObserver 2.0 is a framework that uses machine learning image captioning and a Minecraft plugin that assists users in Minecraft environments designed to promote science learning. The system seeks to both assess learner observations of these worlds and deliver feedback intended to improve their ability to make accurate and productive observations primarily in the context of Astronomy learning.

Background

Minecraft

Minecraft is an exceptionally popular game. Since its release in 2009, the user base has exploded with over 140M players and 241M logins per month and consistently ranks in the top 5 most popular games for children.[1] It spans many platforms and its players have a range of interests, ages and experience. It is referred to as a "sandbox" game because it can be used in several different modes and contexts and often participants come up with their own challenges and meanings when playing alone or with others. The Java Edition of the game has an enormous community following and is very modifiable, which makes it an ideal candidate to create more complex teaching and learning simulations like the one exhibited in this paper.

Minecraft as a Learning Environment

As an extremely modifiable game, Minecraft can be used as a learning environment. Several studies (Lane and Yi 2017; Bos et al. 2014; Lim 2020; de Andrade, Poplin, and Sousa de Sena 2020) have used Minecraft to teach a variety of topics including (but not limited to) basic mathematics, civic engagement, and even music. Nearly each study found that Minecraft's flexibility allowed the students to be more creative and have more freedom than a traditional classroom setting. A key component of learning environments is the ability to collaborate with other learners. This is easily achieved since Minecraft can be played with multiple users which can encourages collaboration including collecting difficult achievements, building large structures, and even exploring worlds that are treacherous together. Each of these tasks can be done alone, but would require a great deal of time and effort on one single player. Moreover, modifying the game via a Java plugin, programmers can create new tools or achievements that require multiple users in order to achieve a rich collaborative experience.

Minecraft for Science Learning

The research shared in this paper is part of the NSF-funded project WHIMC (What-If Hypothetical Implementations in

[1] https://news.xbox.com/en-us/wp-content/uploads/sites/2/2021/04/Minecraft-Franchise-Fact-Sheet_April-2021.pdf

Minecraft). WHIMC investigates the use of Minecraft as an educational tool for science learning, with an emphasis on Astronomy content that engages children and promotes interest in STEM (Lane et al. 2022). WHIMC is implemented as a Minecraft (Java Edition) server consisting of a space station hub and a collection of worlds to visit. On these worlds, learners interactively explore the scientific consequences of alternative versions of Earth via "what if?" questions (e.g., "What if the earth had no moon?") as well as feasible representations of several known exoplanets. It is hoped that such experiences will act as *triggers* of interest (Yi, Gadbury, and Lane 2021), which are required in order for interest to be sustained over time (Renninger and Hidi 2015).

A key component of the project is to analyze how learners interact with the environment and assess their engagement with and understanding of science content. Players are invited to participate in exploration challenges where they can learn about and measure pertinent science characteristics of simulated worlds (such as temperature and radiation). In addition to measurements, learners also make written observations about things they think are noteworthy or interesting in some way, which appear as floating text. For example, without a moon and its gravitational pull, the Earth's rotation would be much faster than it is now. This would cause fierce winds on the surface of Earth. To withstand the force of such wind trees would need to be shorter, wider, and stronger (Comins 1993). An example of an observation for this phenomena is shown in Figure 1. We note that the combination of a screenshot and a description forms the basis for our data set described in the next section.

Figure 1: A sample observation of tree variation on a version of Earth with no moon.

WHIMC provides a framework for making observations like a scientist. In particular, based on our prior work to assess learner observations (Yi, Gadbury, and Lane 2020), we identified five key categories for observations:

1. *Factual*: observations are comprised of nouns without any elaboration.
2. *Descriptive*: observations related to color, temperature, quantity, and other physical attributes such as weight or size.
3. *Comparative*: observations comparing one natural phenomenon to another.
4. *Analogies*: observations comparing natural phenomena with another similar structure or object (an advanced form of comparative).
5. *Inferences*: observations where a hypothesis or explanation is proposed (the most advanced form, rare for middle school students to do spontaneously).

Observations are also visible to all players on the server, so they might prompt other learners to take notice. In addition, the WHIMC server captures additional data, including player coordinate and directional facing data to better understand what students were observing at the time.

MineObserver

Previously, MineObserver introduced a AI-method to assess Minecraft imagery from students in a Minecraft world. This method used Image Captioning and cosine similarity to check if a student was correctly observing the intended structure, building, etc. This initial effort was not implemented in-game and instead acted like an outside grader. Moreover, the feedback system lacked depth and did not adequately focus on improving learner observation skills. Our current work improves on this method by directly interacting with the student while improving all aspects of the framework (captioning, feedback, etc.) to give the students a better learning experience and more directly support their emerging skills in making scientific observations.

MineObserver 2.0 AI Framework

Our AI framework can be broken down into three major parts: the Photographer, the AI Architecture, and the Visualizer. The sequence is as follows. The student makes a visual observation which consists of the student's in-game coordinates and a caption. Next, the Photographer teleports to the student's coordinates and takes a screenshot of the student's POV. The Photographer then sends the student's POV and caption to our AI Architecture. The AI Architecture uses the student's POV and caption to predict how accurate the observation is and gives feedback to the student. The AI Architecture also sends the feedback and a score to the Visualizer. At the end, the student receives feedback via the Photographer and the Visualizer displays the observation made by the student with the AI's feedback. The feedback aims to help an individual student's observation skills while the Visualizer acts an intervention tool for the teacher to further guide a class of students to highlight, focus, and understand which observations are strong, weak, average, etc. Our entire framework can visually depicted in Figure 2

Photographer

To enable real-time captioning and feedback in game, we introduce the Photographer. The Photographer is a Minecraft client (a separate account with an independent login) that teleports to a learner's coordinates to capture their point of view via screenshot. This client is tied via unique ID to a spigot-plugin that sends data to our AI Architecture and displays the AI's result to the student in real time, right after

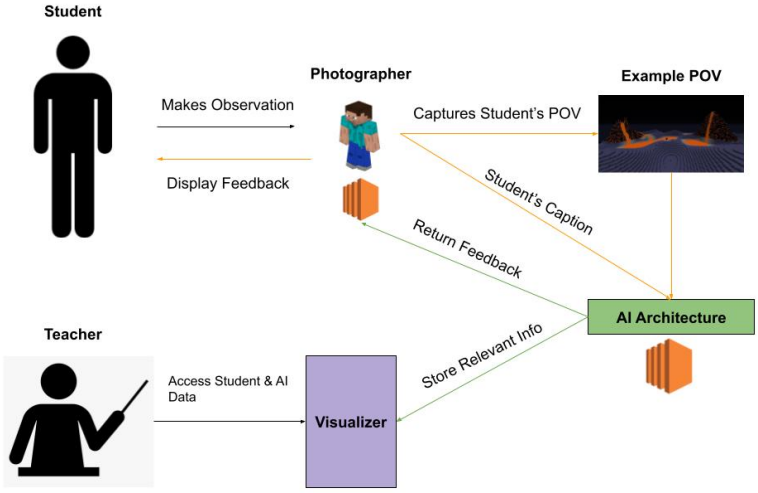

Figure 2: MineObserver 2.0's AI Framework. Note that the Photographer and AI Architecture are hosted on AWS EC2 instances for ease of access.

they make the observation. To make the Photographer readily available, we host our entire client on an AWS EC2 instance. The student will never physically see the Photographer client in action as it is always in invisible "spectator" mode. This way we avoid interrupting the student's gameplay when making observations.

AI Architecture

The AI Architecture consists of three parts: a Image Caption model, a RoBERTa (Liu et al. 2019) model and the feedback system. From the Photographer, the Image Caption takes the student's POV, we pass the image to a Image Caption model to generate a caption. In this case, the generated caption acts like an expert observer which can be used to compare against the student's observation. Using the student's and generated captions, we pass them through the RoBERTa model. At the end, we use the data from the RoBERTa model to generate feedback and pass the results to the teacher and student. To make the AI Architecture available, we create a REST API and host it on an AWS EC2 instance. We summarize our AI Architecture in Figure 3.

Image Captioning Model

Since Minecraft primarily focuses on building and crafting structures, we decided to use an Image Captioning model that can focus on parts of the images. This led us to use an attention based model similar to (Xu et al. 2015). In our case, we used a Convolutional Neural Network (CNN) and a Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) with visual attention. While the approach mentioned above uses a pre-trained CNN on ImageNet (Deng et al. 2009), we found that this resulted in poor performance thus we train both the CNN and LSTM.

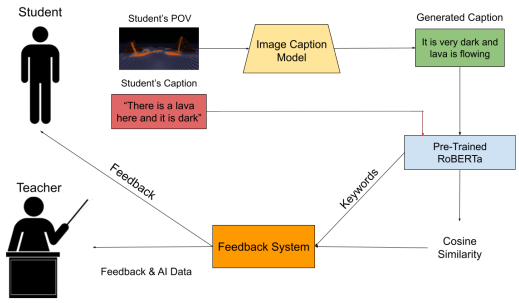

Figure 3: MineObserver 2.0's AI Architecture.

During testing time, instead of using a greedy approach to decode our image, we use beam search with $k = 3$ possible candidates. This allows us to consider the entire sequence rather than individual words when generating our captions.

RoBERTa Model

As stated previously, students engaging with the WHIMC platform make observations while exploring different Minecraft maps. These observations are important: they simultaneously reflect how deeply engaged the learner is in the experience and reveal (to an extent) the level of understanding they have for the science concepts. Our agent must assess the content of these observations to guide their pedagogical actions.

To accomplish this, We utilize the RoBERTa model fine-tuned on the Semantic Textual Similarity benchmark (STSb) and Natural Language Inference (NLI) dataset to encode the image caption generated by the CNN and the student's observation. Facebook's RoBERTa model has been shown to

outperform BERT on the General Language Understanding Evaluation (GLUE) benchmark, Stanford Question Answer Dataset (SQuAD), and ReAding Comprehension from Examinations (RACE) dataset (Liu et al. 2019). In addition, fine-tuning the model with STSb and NLI has been shown to improve sentence encodings for common text similarity tasks (Reimers and Gurevych 2019).

For our purposes, we use RoBERTa to perform cosine similarity between the student's caption and the generated caption which produces a score that allows us to understand how closely related the two captions are. We also use this model to perform keyword detection which will be later used in our feedback system.

Feedback System

Using the RoBERTa model, we request λ keywords from the generated caption and collect the cosine similarity score. We set a threshold for the cosine similarity γ to decide how similar the captions are. If the score is at or above the threshold, then we return the following phase: "Excellent work, you noticed the {keywords} here!". And similarly if the score is less than the threshold, then we return the following phase: "Try again! Did you notice the {keywords}?". In our case, the keywords acts as primary features in the image to focus on. That way, when the student does not meet the threshold score, our system can guide the student to focus on those parts of the observation and try again. At the end, we send the results back to the student (via Photographer) and the teacher (via Visualizer).

Visualizer

The Visualizer is a tool for the instructors to guide students when they make observations. Every time an observation is made, the Visualizer instantly picks it up and displays the student's username, observation, AI-generated observation, and feedback. At the end of the session, the instructor can export all of the data to a folder that contains the images and a CSV of all text data. The goal of the visualizer is to support instructors in initiating discussions with students on observations made and evaluate students' performance and engagement with relevant scientific content. Figure 4 shows how our Visualizer looks.

Experiments

Dataset

Since there was no existing dataset of Minecraft images paired with their observations, we had to create our own. We collected hundreds of images of size 1920 x 1080 across the WHIMC server and the internet, captioned them using Amazon Mechanical Turk or manually, and labeled the type of observation (e.g Descriptive, Comparative, or Inferential). To target our intended users of this system, we had children from ages 8-14 also caption a set of images and filter out any captions that were either off-topic or extraneous. An important aspect of the WHIMC project is to broaden participation. As such, a substantial amount of our observation training data came from summer camps with participants that are from historically underrepresented groups in STEM. Eventually, our dataset consisted of over two thousand unique datapoints which was sufficient to capture main points of interest in the WHIMC server.

Pre-processing & Data Augmentation

Before we load images into our model, we center-cropped them to 1024 x 1024 and resized them to be 256 x 256 since most of the intended locations of interest lied in the center of the image. When training the model, we augmented our data by adding random horizontal flips and randomly rotated our image by -5 to 5 degrees.

Training

The only part of our framework that is required to be trained is our Image Caption model. We coded our model using PyTorch (Paszke et al. 2019). For our CNN, we decided to use ResNet (He et al. 2015) as our main choice as it is performed against other CNN models such AlexNet (Krizhevsky, Sutskever, and Hinton 2012) and VGGNet (Simonyan and Zisserman 2014) while being relative lightweight compared to models such as DenseNet (Huang et al. 2018).

Implementation

We trained our entire model via backpropagation by using an Adam optimizer (Kingma and Ba 2014), with a learning rate of 3e-4, and used a cross-entropy loss function for 150 iterations. We used an Nvidia Tesla T4 GPU to train our model quickly.

For our purposes, we decided to detect $\lambda = 2$ keywords from the generated caption and let the cosine similarity threshold to $\gamma = 0.5$.

Results

Summer Camps

Our tool was used as an integral component of our summer camps, which took place from June until August. A total of 73 participants ages 8 to 14 where ages 8 to 10 made up roughly 40% and ages 11 to 14 made up roughly 60% of the student age group. Each camp consisted of roughly 15 hours of instructional time focused on teaching science and astronomy concepts, as well as building in response to habitation requirements. They visited a variety of "what-if" simulations of earth, from "what if earth had two moons" to exoplanet gas giants like Gliese-436b. Along the way students learn about science variables like gravity or atmospheric composition and engage in making scientific observations to later develop and investigate a hypothesis. Many students struggle to know what or how to observe and sometimes get distracted or off-task, our system helps to remedy this. Additional information on our curriculum and approach to teaching and learning with AI and technology can be found at https://whimcproject.web.illinois.edu/.

For each camp, we conducted a double blind experiment. Each of the camps received either our work or the previous work's framework. Neither did the students nor the instructors knew which work was running at a camp. To ensure a

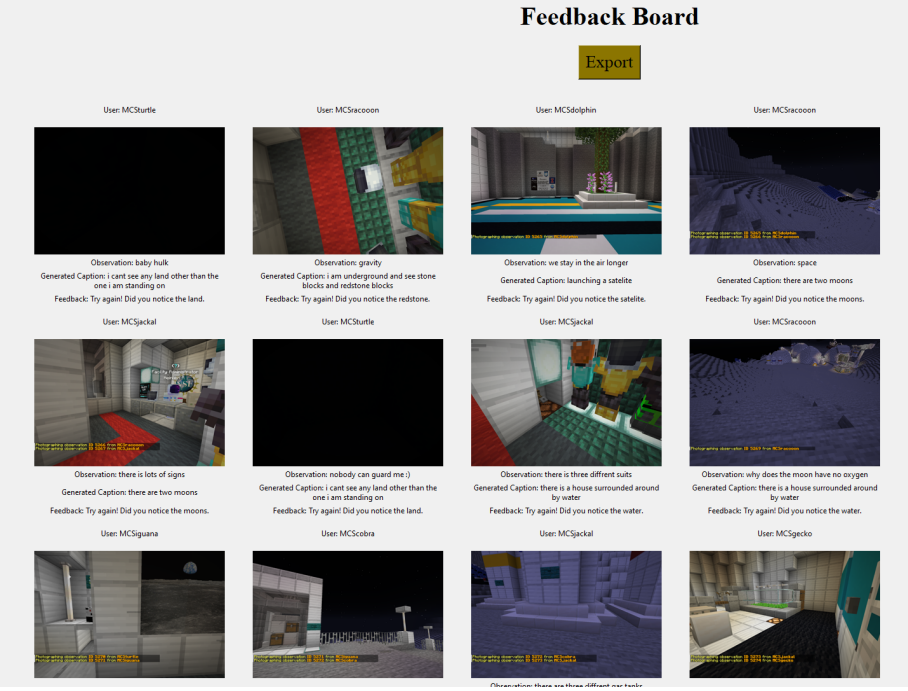

Figure 4: Main UI of the Visualizer. Each of students' observations are listed with their POV, the observation made, the AI's generated caption, and the feedback from the AI's system. At the end, an instructors can export the results to get a folder of images and a CSV of all the Visualizer data

Significance Test	degrees of freedom	p-value	Significant at $\alpha = 0.05$
Mean student's rating of AI's generated caption	56.806	**0.03196**	✓
Mean student's rating of AI's feedback	49.521	**0.001719**	✓

Table 1: Summary of 2-Welch t-test for each experiment. In each case, we compare our model against MineObserver. $n = 61$

fair comparison, we adjusted the prior work to work at runtime rather after the camp. We conducted this for each of the 6 camps where we explain to the students what AI is via a video and talk and point out that the tool is there to help them. Students first used a simple web interface to each contribute 5 practice observations of still images from our simulation worlds to our future training dataset. They then made a variable amount of observations individually on their own in-game. We summarize the results in table 1 and in table 2.

Accuracy of Generated Captions of Minecraft Images

We first asked students about the AI's accuracy when generating a caption (observation) to determine if the attention-based image captioning was a useful improvement:

How accurate was the AI's generated observations?

The students rated the AI's overall accuracy of the generated observations from 1 (not accurate) to 5 (very accurate).

In this case, a 2-sample mean Welch t-test (Welch 1947) would be appropriate to compare the students who had our version against the students who had the previous work (MineObserver). Thus, our null and alternative hypothesis can be framed as:

H_o : *The mean students' rating of the AI's generated obervations does not differ between our work and the prior work.*

H_a : *The mean students' rating of the AI's generated obervations for our work is greater than the previous work.*

If we let μ_R be the mean students' rating of the AI's generated observation, then mathematically we can expresses this as:

$$H_o : \mu_{Ra} = \mu_{Rb}$$

$$H_a : \mu_{Ra} > \mu_{Rb}$$

Where a is our work and b is the previous work.

With a degrees of freedom of 56.806, we receive a t-value of 1.8896 which corresponds to a p-value of 0.03196. At $\alpha = 0.05$, this is statistically significant thus we reject the null hypothesis and suggest that the student's mean rating of the AI's generated observations for our work is better than

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
MineObserver	0.197	0.0294	**0.0117**	0	0.12165
Ours	**0.2**	**0.033**	0.0095	**0.0055**	**0.14647**

Table 2: Standard Image Captioning Metrics

the previous work. This can be seen in table 1 in our second row.

To further prove that our image captioning model was more accurate, we also computed several pre-existing image captioning metrics. We showcase these in table 2. In almost of all of the metrics, our work beats the previous work.

Feedback Accuracy

Similarly, we asked the students to rate the feedback given by the AI's feedback system. This would ultimately test if students prefer the keyword feedback rather than the generic feedback from the previous work. We posed the following statement to the students:

The feedback from the AI was useful.

The students rated this statement from 1 to 5 where a rating of 1 would correspond to not useful feedback and a rating of 5 would correspond to very useful feedback.

In this case, a 2-sample mean Welch t-test would be appropriate to compare the students who had our version against the students who had previous work. Thus, our null and alternative hypothesis can be framed as:

H_o : *The mean students' rating of the AI's feedback does not differ between our work and the prior work.*

H_a : *The mean students' rating of the AI's feedback for our work is greater than the previous work.*

If we let μ_F be the mean students' rating of the AI's feedback, then mathematically we can expresses this as:

$$H_o : \mu_{Fa} = \mu_{Fb}$$

$$H_a : \mu_{Fa} > \mu_{Fb}$$

Where a is our work and b is the previous work.

With a degrees of freedom of 49.521, we receive a t-value of 3.0733 which corresponds to a p-value of 0.001719. At $\alpha = 0.05$, this is statistically significant thus we reject the null hypothesis and suggest that the student's mean rating of the AI's generated observations for our work is better than the previous work. This can be seen table 1 in our third row.

Other Related Results

In-Game Performance

The experience in game is seamless. A player will make an observation (depicted in figure 5) and will get feedback about their caption about 20 seconds later. Once the observation is created the Photographer is put to work. From the student's point of view, they will not see anything; the Photographer is invisible.

Figure 5: An observation to be graded by MineObserver 2.0's AI

Once the feedback has been generated and returned by the Photographer, the player receives a message in chat with feedback about the player's caption as well as a caption generated by the AI. An example of the response can be seen in figure 6.

Figure 6: An example of feedback from the AI's feedback system

To make sure that the gameplay felt smooth, we asked the students to rate the speed of the AI system from 1 to 5 where a rating of 1 would suggest that the AI was too slow to reply whereas a rating of 5 would suggest that the AI was very quick. Bar chart (figure 7) shows the overall result of this question.

The student gave a median score of 3. This suggest that the in-game speed was adequate but can be improved for a better experience. One suggestion to have more than one Photographer client active to handle the students' activity.

Visualizer Results

Qualitative data was collected via short post-session reflection interviews with five different instructors to assess the effectiveness of using the visualizer dashboard. The instructors relayed that the visualizer was helpful in two major ways. First, it was able to help the them more easily identify which students needed interventions to help stay focused

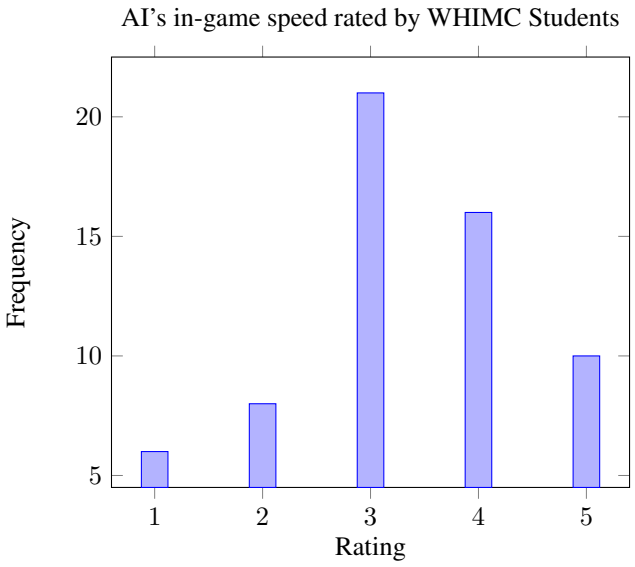

Figure 7: Bar Graph of AI's in-game speed rated by WHIMC Summer Camp Students.

and/or make better observations. This allowed them to focus more on teaching the content of the camps rather than try to visually spot or analyze logs to determine which students were struggling in the middle of session. Second, the instructors used the visualizer to conduct a post-assessment of each day. If a camp had too many students to allow an instructor to pay attention to the dashboard in the moment they could instead leave it on and examine the aggregate visualizer results afterwards. This enabled them to focus on how the overall class performance and reflect on how to address gaps in knowledge or skill during the following camp day. Overall, the instructors unanimously agreed that the visualizer accomplished its primary goal of identifying which students needed intervention in the context of making sufficient observations. That said the interface is still fairly rudimentary and has presents many opportunities for improvement, including (1) cross-platform compatibility via being able to be run as a web app, (2) ability to sort or summarize results for a given time period, (3) the addition of learner profiles to track progress and engagement over time, which could in turn inform the AI, and (4) identify and highlight links that may be common trends or collaboration between participants.

Future Work & Conclusion

There are several ideas we wish to add or improve to MineObserver 2.0. This includes styling our image caption model, continuous learning, and stronger feedback.

Styling Our Image Captioning Model

Many generative models, such as StyleGAN (Karras, Laine, and Aila 2018) allow for generative styles. In our case, we might seek a way to style our image caption model to style based on the type of observation. This would enhance our framework by allowing the student or teacher to choose how they should be graded based on the type of observation.

Continuous Learning

After an observation is made by a student, our framework stores any relevant info to the Visualizer where a teacher can take a look and track student progress. However, since WHIMC is continuously in development, we can use the data from the Visualizer to re-train our image caption model if generated captions are not strong enough. This would allow our model to continually learn as the server grows over time.

Stronger Feedback

Currently, feedback to the learner is limited, typically only directing attention to pertinent visual features nearby and general questions. Future iterations may be able to challenge them with a question to prompt new or additional observations based on their past interests or behaviors, as well as additional engagement with the supporting AI agent or teacher.

Additional Applications

Our tool was developed for the specific context of written observations attached to in-game visuals but may be able to be adapted for alternative use scenarios, including other Minecraft learning simulations or game play. Users could intentionally employ the tool as a kind of note-taking and learner support system while building or exploring. More broadly the ability to take screenshots in a semi-automated fashion based on a Minecraft command interrupt could form the basis of far more complicated pedagogical agents. The plugin and AI essentially mimics and automates the functionality of the camera tool and notebook in Minecraft Education Edition, and could potentially be adapted to create solo-learner self-guided learning experiences. Seeing the great potential we've released our code as open source on Github.

Acknowledgements

The authors would like to thank the participants and instructors who provided support for the data collection and fellow researchers on the project for help with the coding and statistics. This material is based upon work supported by the National Science Foundation under Grants 1713609 and 1906873.

References

Arias, A. M.; and Davis, E. A. 2016. Making and recording observations. *Science and Children*, 53(8): 54–60.

Bos, B.; Wilder, L.; Cook, M.; and O'Donnell, R. 2014. Learning Mathematics through Minecraft. *Teaching Children Mathematics*, 21(1): 56 – 59.

Comins, N. F. 1993. What if the moon didn't exist: voyages to earths that might have been. *New York*.

de Andrade, B.; Poplin, A.; and Sousa de Sena, 2020. Minecraft as a Tool for Engaging Children in Urban Planning: A Case Study in Tirol Town, Brazil. *ISPRS International Journal of Geo-Information*, 9(3): 170.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255.

He, K.; Zhang, X.; Ren, S.; and Sunr, J. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.0338.

Hochreiter, S.; and Schmidhuber, J. 1997. Long Short-Term Memory. Neural Computation:1735–1780.

Huang, G.; Liu, Z.; van der Maaten, L.; and Weinberger, K. Q. 2018. Densely Connected Convolutional Networks. arXiv:1608.06993.

Karras, T.; Laine, S.; and Aila, T. 2018. A Style-Based Generator Architecture for Generative Adversarial Networks.

Kingma, D. P.; and Ba, J. 2014. Adam: A Method for Stochastic Optimization. arXiv:1412.6980.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. NeurIPS:1512.0338.

Lane, H. C.; Gadbury, M.; Ginger, J.; Yi, S.; Comins, N.; Henhapl, J.; and Rivera-Rogers, A. 2022. Triggering STEM Interest With Minecraft in a Hybrid Summer Camp. *Technology, Mind, and Behavior*, 3(4: Winter). Https://tmb.apaopen.org/pub/5pkfd4sc.

Lane, H. C.; and Yi, S. 2017. Chapter 7 - Playing With Virtual Blocks: Minecraft as a Learning Environment for Practice and Research. In Blumberg, F. C.; and Brooks, P. J., eds., *Cognitive Development in Digital Contexts*, 145–166. San Diego: Academic Press. ISBN 978-0-12-809481-5.

Lim, K. 2020. Redstone Jammin': Conversational Analysis from a Collaborative Music-Making Activity in Minecraft. *Journal of Virtual Worlds Research*, 13.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692.

Mahajan, J.; Hum, S.; Ginger, J.; and Lane, H. C. 2022. MineObserver: A Deep Learning Framework for Assessing Natural Language Descriptions of Minecraft Imagery. In *The International FLAIRS Conference Proceedings, 35*. FloridaOJ.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Köpf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library.

Reimers, N.; and Gurevych, I. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv:1908.10084.

Renninger, K. A.; and Hidi, S. E. 2015. *The power of interest for motivation and engagement*. Routledge.

Simonyan, K.; and Zisserman, A. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition.

Welch, B. L. 1947. The Generalization of 'Student's' Problem When Several Different Population Varlances are Involved. *Biometrika*, 34(1-2): 28–35.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhutdinov, R.; Zemel, R.; and Bengio, Y. 2015.

Yi, S.; Gadbury, M.; and Lane, H. C. 2020. Coding and analyzing scientific observations from middle school students in Minecraft.

Yi, S.; Gadbury, M.; and Lane, H. C. 2021. Identifying and Coding STEM Interest Triggers in a Summer Camp. In *Proceedings of the 15th International Conference of the Learning Sciences-ICLS 2021.* International Society of the Learning Sciences.