INFO8010: Project proposal

Simon Gardier,[1] **Lei Yang**,[2] and **Camille Trinh**[3]

[1]*s.gardier@student.uliege.be (s192580)*
[2]*Lei.Yang@student.uliege.be (s201670)*
[3]*camille.trinh@student.uliege.be (s192024)*

I. PROPOSAL 1: OREDETECTOR

A. Introduction

If you already mined in *Minecraft*, you may have find yourself stressing out of fear of walking past valuables ores without seeing them.
With `OreDetector`, this time is over!

B. MVP and more

1. MVP

The idea is to re-implement the famous YOLO object detection model and then train it on a dataset of Minecraft screenshots, labeled with the ores present in the screenshots and their position.

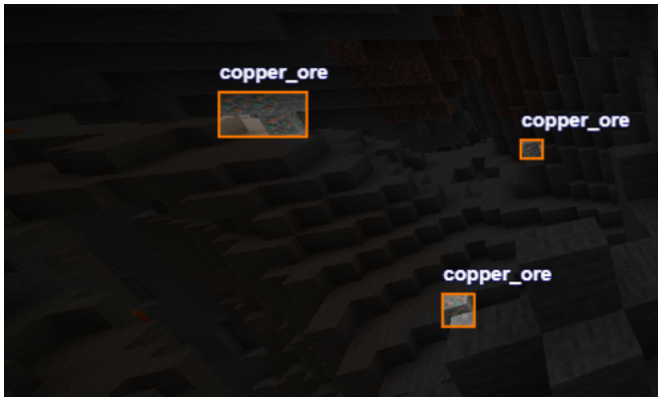

FIG. 1. Result of the MVP

The project would re-implement YOLO v2 (9000). We choose YOLO v2 among the different implementations based on this article comparing the first YOLO models. YOLO is improvable and YOLOv3 present advanced features (e.g. Multi labels prediction) we do not need for our simple use case.

The user will be able to use the tool from the terminal to input an image and get, as output, the resulting annotated image.

Technology: PyTorch

2. To go further

A nice to have would be to offer the possibility to do the ores detection on a video in real-time, not just an image.

Another one would be to integrate it as a minecraft mod to run the model "online" during gameplay. The mod would be developed as a layout that the user could turn on/off. The layout would frame the ores visible in the gameplay.

C. Dataset

For the dataset, the project needs in-game Minecraft screenshots with ores detected, defined, and labeled. We will focus on ores that are found in the Overworld dimension : coal, copper, iron, gold, lapis-lazuli, redstone, emerald, and diamond.
Many choices are already available to us online:
Link to *Minecraft ore dataset*
Link to *Minecraft ore dataset (2)*
Link to *Minecraft ore detection dataset*

D. Computing resources needed

The team possess a NVIDIA RTX3070 on which we are planning to do the training.

E. Related projects

Link to *Initial proposal of YOLO*
Link to *YOLO v2 paper*
Link to *Implementation of YOLO v2 from scratch using Pytorch*

II. PROPOSAL 2: IMAGINECRAFT

A. Introduction

The second proposal aims to create a custom Texture pack based on a user text prompt. This project necessitate to connect multiple models together. The project would be centered around the implementation of the image generation model, while pretrained models would be used for the other parts of the project.

We found this second proposal more interesting than the first proposal, thus it would be our first choice but it also seemed less clear in terms of feasibility. Indeed, no similar project could be found online, making it harder without strong experience in deep learning.

B. MVP and more

1. MVP

The idea would be to implement a *Conditional Latent Diffusion* model as the image generation model. The project could take advantage of the relative small size of the images to generate (16x16 pixels). As the implementation of the *Diffusion model* seems to be already a complete project by itself, pretrained models would be used for the rest of the project.

More specifically, CLIP would be used for the embedding of the Minecraft textures at the training step and for the embedding of the user prompts at the inference step.

The final result for the user would be the generation of a set of n images, where n is is the size of the texture pack generated. The project would generate only a subset of all the minecrfat texture as 1) there are a lot of textures (+-2000) and 2) some are not in 16x16 format (e.g. door, etc...). Only main blocks and items texture would be generated.

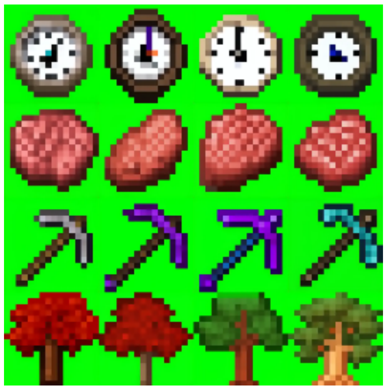
FIG. 2. Example of generated textures

Technology: PyTorch

2. To go further

A nice to have would be to create a "real" texture pack, that the user could import in Minecraft, from the set of generated images.

C. Dataset

For the training, we need as many texture packs as possible. Textures embeddings for training will be generated using CLIP. No useful dataset was found on the internet but a dataset could be easily generated by scrapping the following website. https://www.planetminecraft.com/texture-packs

With +-10000 textures packs (link) matching our search filters and most of them having relevant tags.

D. Computing resources needed

The team possess a NVIDIA RTX3070 on which we are planning to do the training.

E. Related projects

Link to *Stable Diffusion from Scratch in PyTorch — Conditional Latent Diffusion Models*

Link to *High-Resolution Image Synthesis with Latent Diffusion Models*

Texture pack generation using Stable Diffusion and LoRa

Note: While some project with similar results exist, no public implementation "from scratch" could be found for a similar project.