Workshop No. 1
Drawing with LLMs

Daniel Alonso Chavarro Chpatecua - 20241020066
Carlos Andrés Brito Guerrero - 20241020147
Jose Fernando Ramirez Ortiz - 20241020080

April 5, 2025

1 Competition Overview

This competition mainly focuses on using LLMs to generate images in SVG (Scalable Vector Graphics) format as code to render them as images. The idea is to use freely available LLMs such as Gemma or Deepseek, specializing them in the area of image generation to surpass current generative models like Midjourney or ChatGPT. The competition challenges participants to build practical and reusable solutions that follow robust design models and patterns.

Dataset Description

The competition test data contains approximately 500 descriptions of objects and everyday scenes from a variety of domains. These descriptions have the following characteristics:

- The descriptions cover common and generic topics. No brand names, trademarks, or personal names are mentioned. No individuals are referenced, even in a generic manner.
- The described topics span a dozen categories. Three of these categories (landscapes, abstract, and fashion) are present in each of the training, public test, and private test sets. The remaining categories, both in the public and private test sets, are unique to each set. More than half of the descriptions come from the three shared categories.
- No description exceeds 200 characters. The average length is about 50 characters.

Files

- `train.csv` - A set of representative descriptions within the categories of landscapes, abstract, and fashion.
- `kaggle_evaluation/` - A Python package with competition helpers, such as the `test(Model)` function that simulates the evaluation system using a simulated `test.csv`.

- `kaggle_evaluation/test.csv` - An example set of test data to help you author your submissions.

The dataset includes a Python package in `kaggle_evaluation/` with competition helpers, such as a `test(Model)` function, which tests the `Model` class with the `predict()` method. This method takes the description and returns an SVG code of the image. The solution submission must be in Notebooks.

Constraints

- The Notebook does not have internet access.
- External publicly and freely accessible data, including pre-trained models, are allowed.
- The SVG cannot exceed 10,000 bytes in length.
- Each SVG may only include elements and attributes from an allowed list. CSS styling elements are not permitted.
- No SVG may include rasterized image data or external font data.

2 System Analysis

Inputs

- Just a single generic description
- A .csv file containing the generic descriptions of diverse topics.

Output

A single SVG code that represents the image.

System Elements

Description Preprocessing

The description doesn't have any instructions, for example: "a purple forest at dusk", so this element standardizes the prompt to something general that can be more understandable for the LLM model of what to do, changing the aforementioned description to "Generate an image in SVG code format where you cannot make <*Constraints*> and that complies with the description <*description*>". Giving specific instructions in the prompt to have a more accurate result.

LLM Model

The LLM (Large Language Model) is the system component responsible for interpreting the preprocessed prompt and generating unstyled SVG code. Using structured natural language instructions, the model transforms the request into a graphical representation, producing SVG code that may contain syntactical or semantic errors, which are corrected in the post-processing stage. Its performance depends on the quality of the prompt and its training data, and may introduce ambiguities in interpretation if the instructions are not sufficiently precise. Reason that why there is an input processor.

Sensitivity: High, as the main element of the competence, changing its composition would lead to an entire different output, which could help to improve the quality or make it worse.

Output processor

The output processor acts as an intermediary for the SVG generator and the evaluation system, assuring that the SVG code is valid, clean and functional. Since raw SVG output from an LLM may contain syntax errors, missing attributes, or redundant code, this component is designed to automatically detect and correct those issues as well as to optimize the code, deleting unnecessary lines, grouping styles and improving the efficiency in the final product. Another key function belonging to it, is to verify visual compatibility, which means that all colors, shapes and proportions are loyal to the original intention and that it also looks like the prompt stated, in a way that could not leave room to interpretation. In more complex cases, a preview can be offered with alerts about some problematic elements, allowing manual settings if it is seen as necessary. Finally, this processor assures that generated SVGs are not only technically valid, but also visually consistent, setting the ground for a more precise and constructive evaluation.

Evaluation System

The evaluation system is the component that is in charge of validating the quality and utility of the SVG images generated. Its analysis contains both technical and aesthetically aspects. Firstly, it verifies that the SVG is free from syntactic errors and if whether or not checks with the format standards. Then, the visual coherence is evaluated, in which elements like symmetry or harmony come to play. This analysis can be done with traditional algorithms or through trained neural networks, with the main goal of recognizing great quality pattern designs. Besides, the system also validates if the image is scalable and adaptable to different devices, guaranteeing its applicability in various diverse contexts. The evaluation can also incorporate human feedback, allowing designers or even users to rate the given results and provide specific comments to make the system better. When failings or areas needing for improvement are identified, the system will generate an structured report that is sent to the LLM model.

Sensibility: High, Changing the parameters of evaluation of the model would lead to an entire redefinition of the model.

Relation

- The Description preprocessor creates a prompt to be used to the LLM, so is an unidirectional relation
- The LLM outputs a SVG code that could be with syntax error, so the LLM is related with the output processor

Complexity

As you can see the complexity of the system is almost minimal, due to the simple inputs, outputs, well characterized competition and not an entire data structure representing the inputs and outputs, excluding the SVG image.

Chaos and Randomness

At the beginning of the system, there is a possibility that the randomness of the LLM when generating SVG images is high due to its lack of training (it may be less random if the LLM was already pre-trained for general purposes) giving results to meaningless and unconnected figures, however, thanks to the evaluation system the system can provide feedback by changing its SVG generation parameters to obtain a higher score, identifying patterns and tending towards a balance (homeostasis process).

Feedback loops are essential for the constant learning that the model must have, since once the evaluation system has gone through all the required tests and has detected some possible errors or deficiencies, that information is going to be then translated into settings that will be used as a basis for improvement to the LLM model. This can be done through multiple ways, fine-tuning new corrected examples, or reinforcement learning based on quality ratings, even refining the prompt that was used in the initial image generation. The presence of human rating and evaluation is important since it gives space to the model to learn styling or contextual preferences, until it reaches a "soft cap" where the score is maximum and the system cannot improve further, it is at that point where the system is already in balance.

System Boundaries

Regarding system boundaries, we must consider that although we have an evaluation system as an element for the feedback process, the evaluation system for testing and grading the model is not available, leaving it outside the proposed system. On the other hand, other data that may have been used outside of the competition data are subject to modification depending on the dataset update time.

3 Conclusion

This system for generating images in SVG format from text descriptions using large language models (LLMs) presents several strengths and challenges. In the greatest strengths of the system we have its continuous interaction between the LLM and the evaluation system allows for adaptive learning. Feedback can be used to adjust instructions, update models or improve generation strategies over time. There are also versatile evaluation metrics with human comments where the evaluation system takes into account both technical (syntax, scalability) and aesthetic (symmetry, harmony) aspects of SVGs, allowing for thorough quality control and human comments are incorporated to improve results based on subjective preferences, allowing for style and context adaptation.

However, the system also has some weaknesses. Initially, the generation of SVGs can be chaotic and inaccurate due to the inherent randomness of LLMs and the lack of specific training. Also, the reliance on the evaluator to improve image quality introduces latency in model optimization. Another critical aspect is the maximum size restriction of 10,000 bytes for SVG files, which limits the complexity and level of detail of the generated images.

4 References

Drawing with LLMs. (s. f.-b). Kaggle. https://www.kaggle.com/competitions/drawing-with-llms