DeTikZify

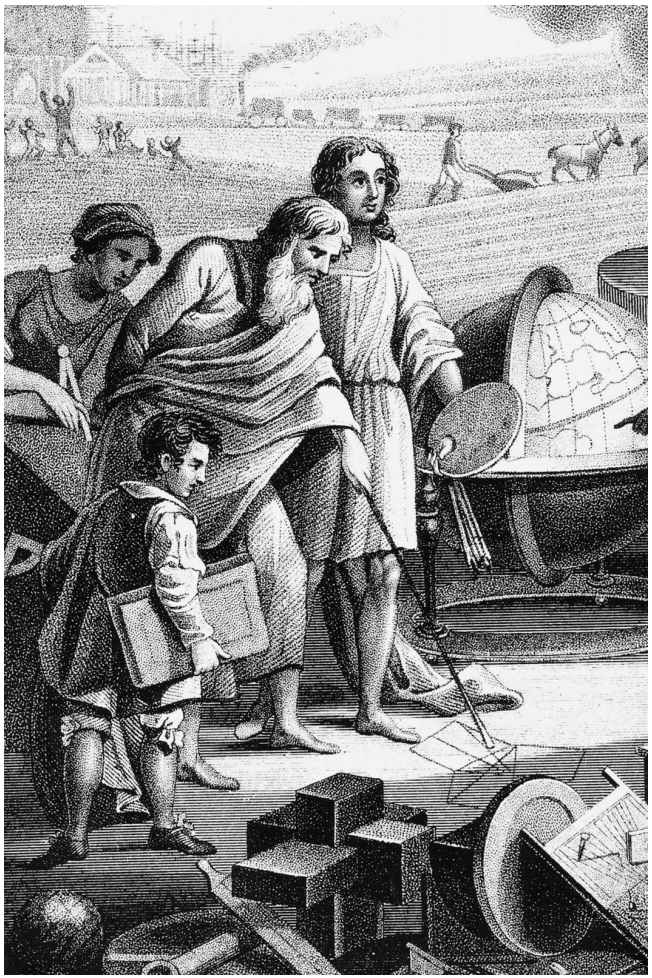
UNIVERSITY OF MANNHEIM

DeTikZify: Synthesizing Graphics Programs for Scientific Figures and Sketches with TikZ

Jonas Belouadi and **Simone Ponzetto** and **Steffen Eger**
Natural Language Learning Group (NLLG) & Data and Web Science Group (DWS)

NLLG

On Scientific Figures

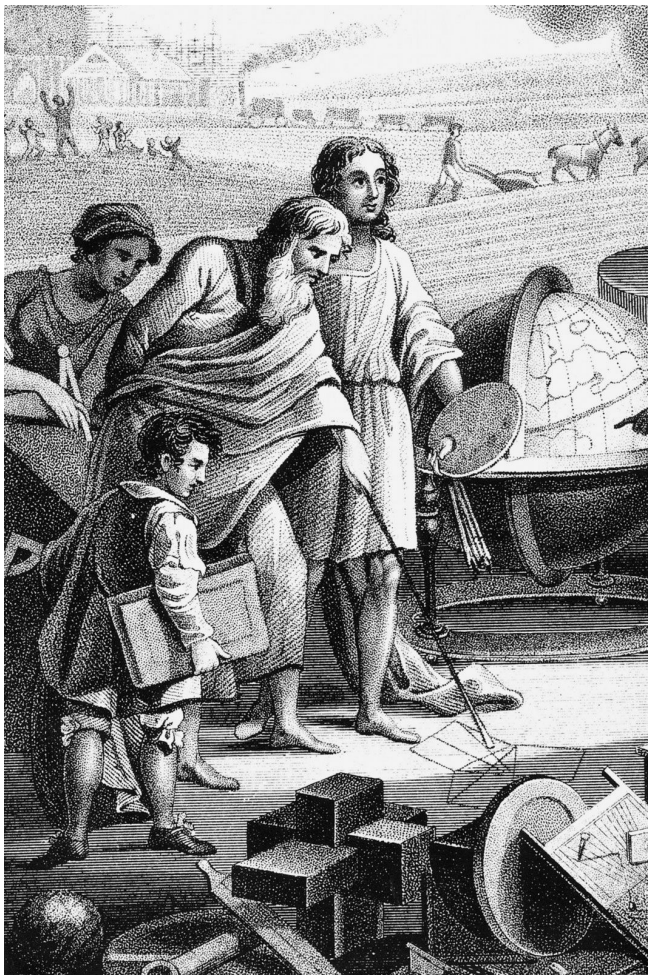


Observation 1

Even though sketching ideas on paper is easy, creating high-quality scientific figures can be time-consuming and challenging.

Source: https://www.britannica.com/biography/Pythagoras

On Scientific Figures



Observation 1

Even though sketching ideas on paper is easy, creating high-quality scientific figures can be time-consuming and challenging.

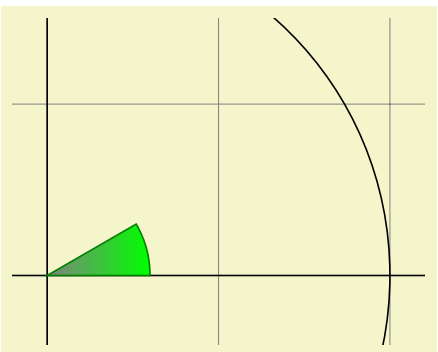
Observation 2

Recreating existing figures that are stored in low-level formats (JPG, PNG, PDF, SVG, …) which do not preserve semantic information is equally complex.

Source: https://www.britannica.com/biography/Pythagoras

Introducing DeTikZify

We introduce DeTikZify, a multimodal LLM that automatically synthesizes scientific figures based on sketches and existing figures as semantics-preserving TikZ graphics programs. We leverage TikZ because it is:



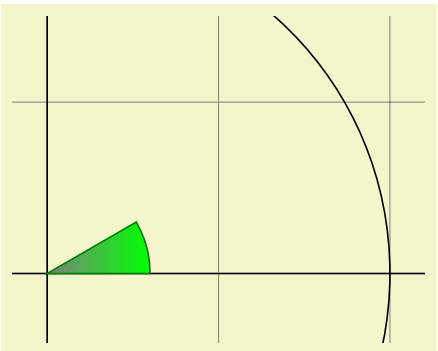
```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
\end{tikzpicture}
```

Source: https://tikz.dev

Introducing DeTikZify

We introduce DeTikZify, a multimodal LLM that automatically synthesizes scientific figures based on sketches and existing figures as semantics-preserving TikZ graphics programs. We leverage TikZ because it is:

- expressive can represent complex figures with few commands



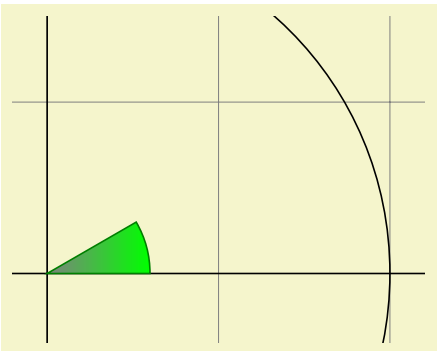
```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
\end{tikzpicture}
```

Source: https://tikz.dev

Introducing DeTikZify

We introduce DeTikZify, a multimodal LLM that automatically synthesizes scientific figures based on sketches and existing figures as semantics-preserving TikZ graphics programs. We leverage TikZ because it is:

- **expressive** can represent complex figures with few commands
- **versatile** supports a wide range of figure types (and more)



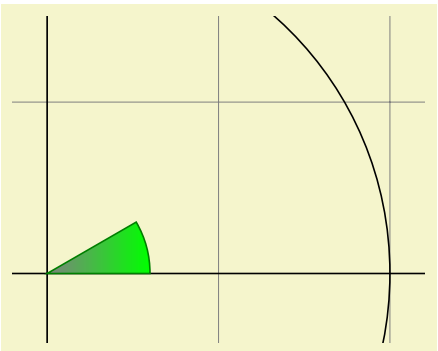
```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
\end{tikzpicture}
```

Source: https://tikz.dev

Introducing DeTikZify

We introduce DeTikZify, a multimodal LLM that automatically synthesizes scientific figures based on sketches and existing figures as semantics-preserving TikZ graphics programs. We leverage TikZ because it is:

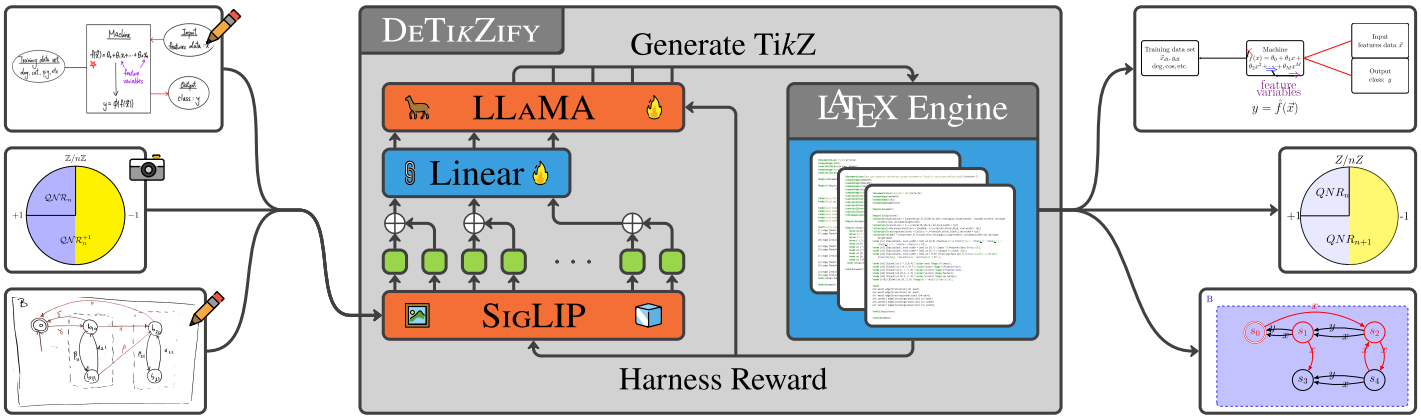
- **expressive** can represent complex figures with few commands
- **versatile** supports a wide range of figure types (and more)
- **popular** most frequently discussed topic on TeX.SE



```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
\end{tikzpicture}
```

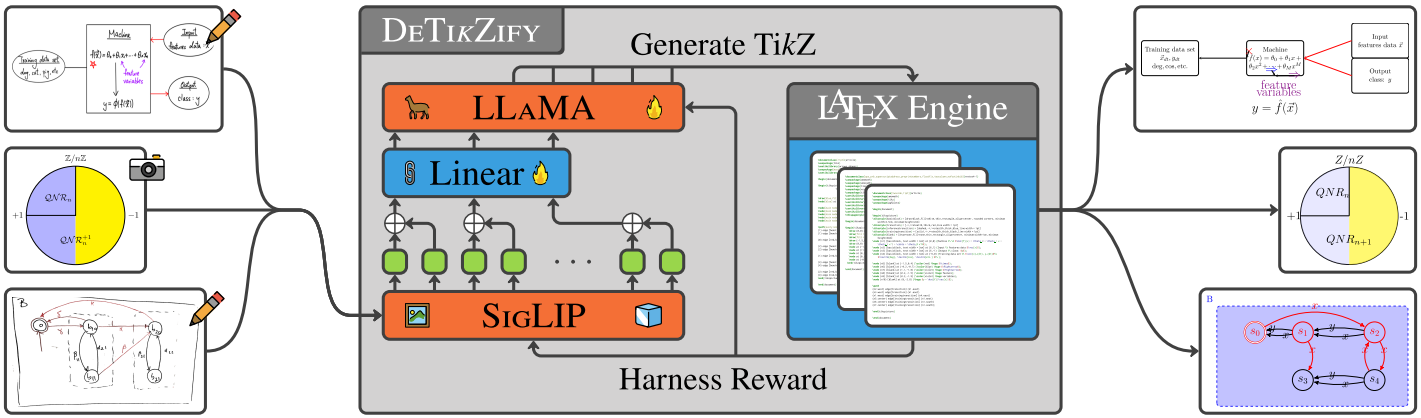
Source: https://tikz.dev

Key Contributions



datasets We collect DaTikZ$_{v2}$ with over 360k TikZ graphics; SketchFig consisting of sketches paired with scientific figures; and MetaFig, a meta-dataset of figures and associated metadata.

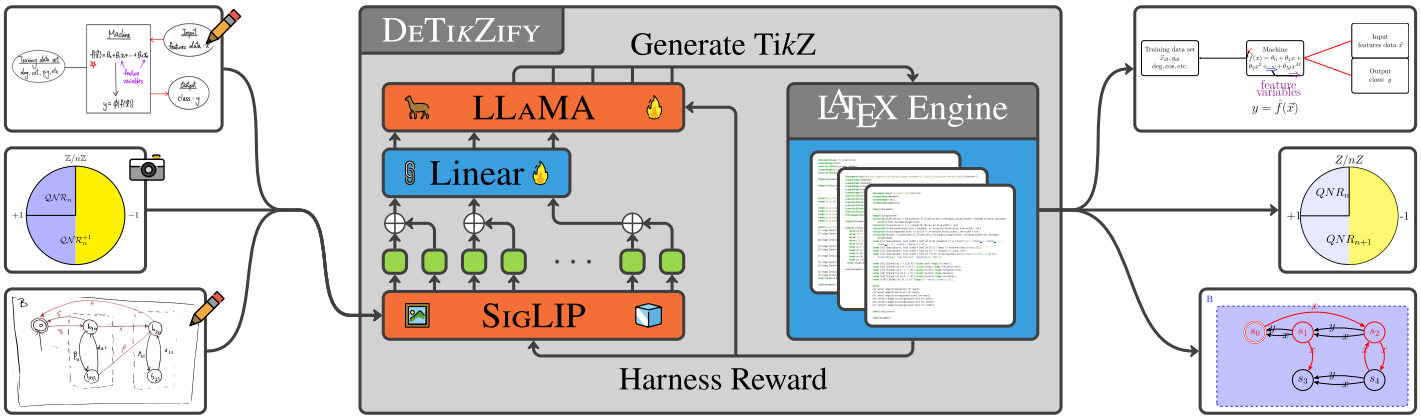
Key Contributions



datasets We collect DaTikZ$_{v2}$ with over 360k TikZ graphics; SketchFig consisting of sketches paired with scientific figures; and MetaFig, a meta-dataset of figures and associated metadata.

model We train DeTikZify on MetaFig, DaTikZ$_{v2}$, and synthetic SketchFig-like sketches, then demonstrate its performance.

Key Contributions



datasets We collect DaTikZ$_{v2}$ with over 360k TikZ graphics; SketchFig consisting of sketches paired with scientific figures; and MetaFig, a meta-dataset of figures and associated metadata.

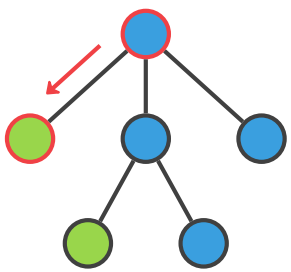
model We train DeTikZify on MetaFig, DaTikZ$_{v2}$, and synthetic SketchFig-like sketches, then demonstrate its performance.

inference We present an inference algorithm based one Monte Carlo Tree Search (MCTS) that allows DeTikZify to iteratively refine (IR) its outputs, improving performance without additional training.

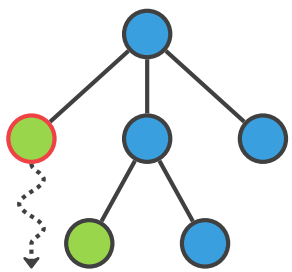
Iterative Refinement with MCTS

The algorithm iteratively builds a search tree and repeatedly runs simulations. Each node's state consists of *n* lines of TikZ code, and edges represent possible continuations. Each simulation performs four steps:

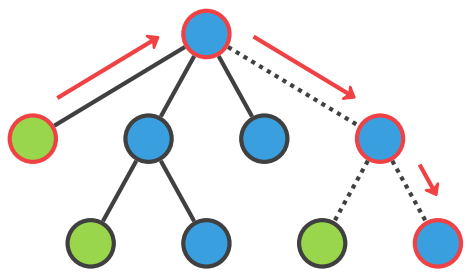
(i) Selection



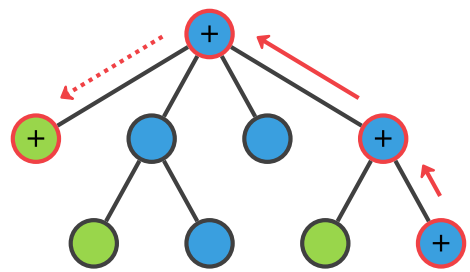
(ii) Rollout



(iii) Expansion



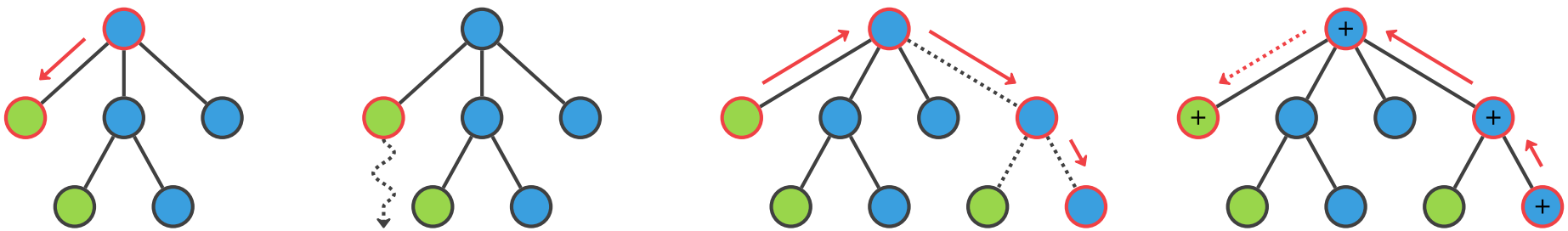
(iv) Backpropagation



Iterative Refinement with MCTS

The algorithm iteratively builds a search tree and repeatedly runs simulations. Each node's state consists of *n* lines of TikZ code, and edges represent possible continuations. Each simulation performs four steps:

(i) Selection (ii) Rollout (iii) Expansion (iv) Backpropagation

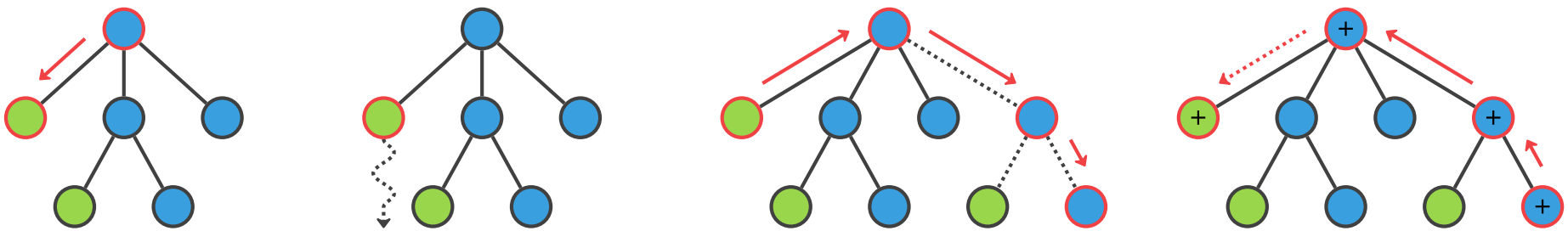


(i) Selection: Select a path to a leaf node based on the scores of each node.

Iterative Refinement with MCTS

The algorithm iteratively builds a search tree and repeatedly runs simulations. Each node's state consists of *n* lines of TikZ code, and edges represent possible continuations. Each simulation performs four steps:

(i) Selection (ii) Rollout (iii) Expansion (iv) Backpropagation

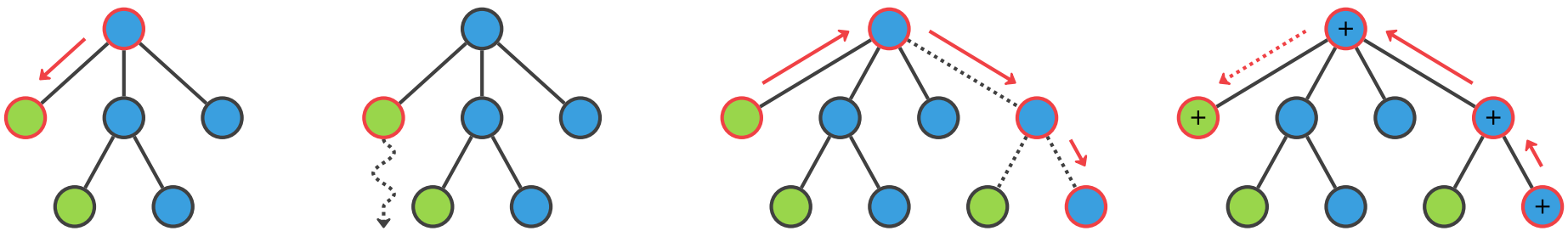


(i) Selection: Select a path to a leaf node based on the scores of each node.
(ii) Rollout: Starting from the state of the leaf node, use DETIKZIFY as a rollout policy and complete the code until an <eos> token is reached.

Iterative Refinement with MCTS

The algorithm iteratively builds a search tree and repeatedly runs simulations. Each node's state consists of *n* lines of TikZ code, and edges represent possible continuations. Each simulation performs four steps:

(i) Selection (ii) Rollout (iii) Expansion (iv) Backpropagation

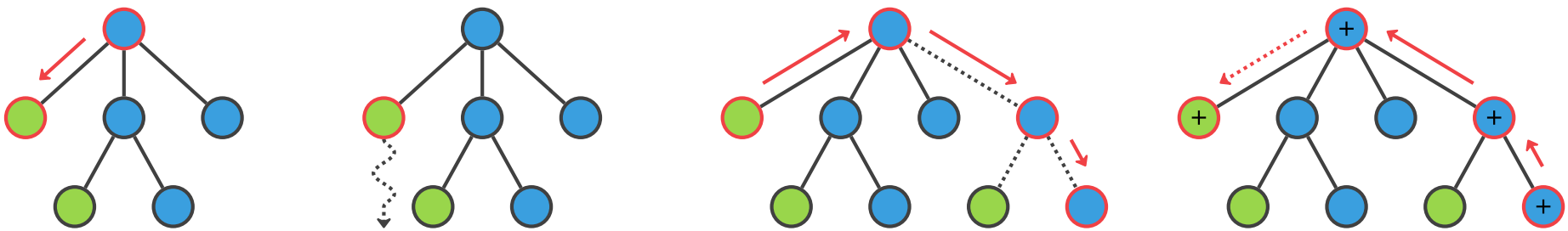


(i) Selection: Select a path to a leaf node based on the scores of each node.
(ii) Rollout: Starting from the state of the leaf node, use DETIKZIFY as a rollout policy and complete the code until an <eos> token is reached.
(iii) Expansion: Expand the search tree by appending nodes from the rollout.

Iterative Refinement with MCTS

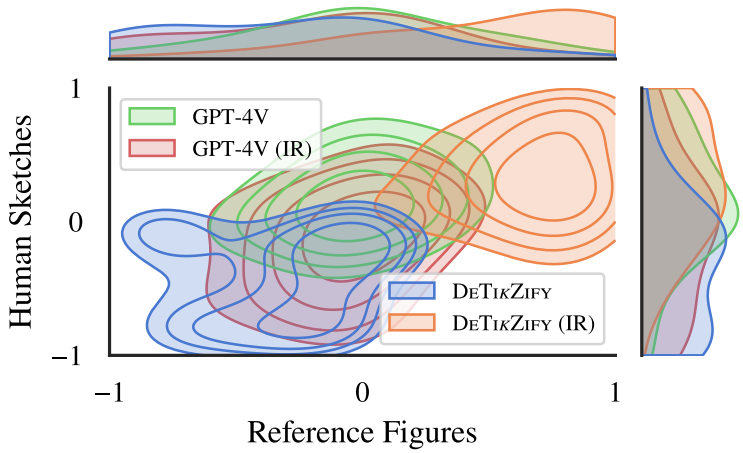
The algorithm iteratively builds a search tree and repeatedly runs simulations. Each node's state consists of *n* lines of TikZ code, and edges represent possible continuations. Each simulation performs four steps:

(i) Selection (ii) Rollout (iii) Expansion (iv) Backpropagation



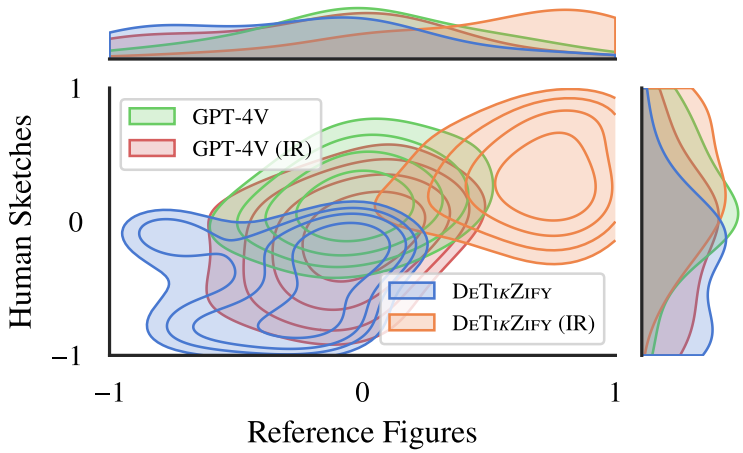
(i) Selection: Select a path to a leaf node based on the scores of each node.
(ii) Rollout: Starting from the state of the leaf node, use DETIKZIFY as a rollout policy and complete the code until an <eos> token is reached.
(iii) Expansion: Expand the search tree by appending nodes from the rollout.
(iv) Backpropagation: Use DETIKZIFY to evaluate the rollout and backpropagate the reward score to each node until the root is reached.

Evaluation



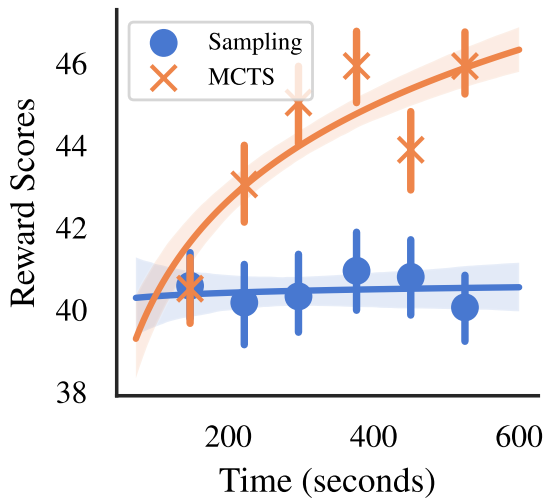
- At first, humans rate GPT-4V outputs higher than DeTikZify from -1 (bad) to 1 (good).
- However, after applying IR with MCTS, DeTikZify ranks highest. Further, IR techniques do not work well with GPT-4.

Evaluation

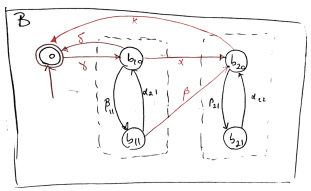
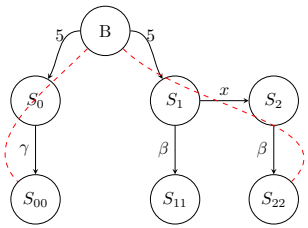
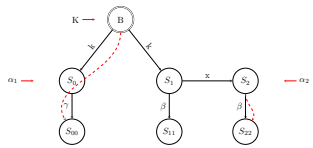
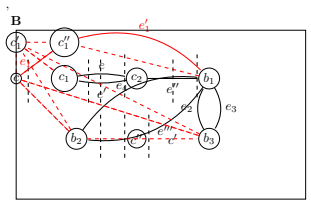
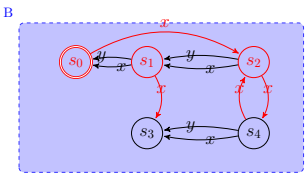
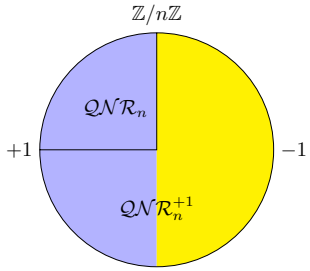
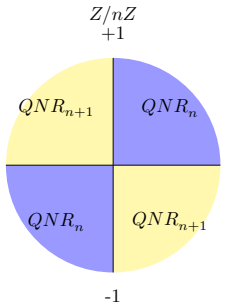
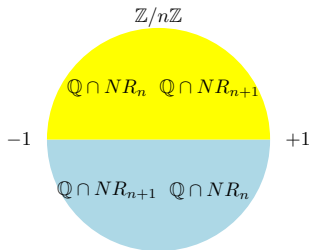
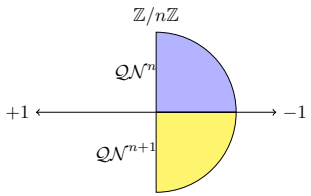
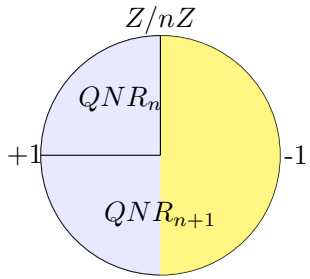
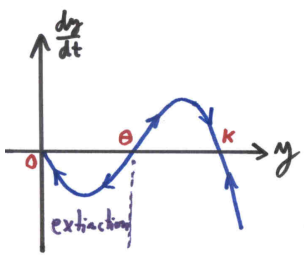
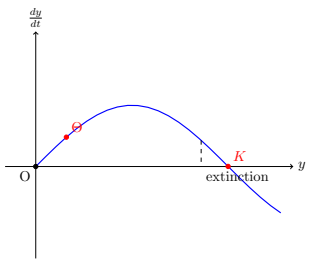
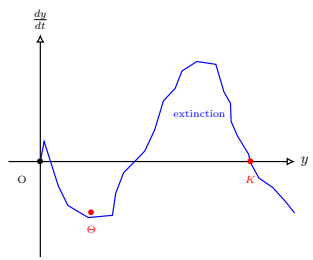
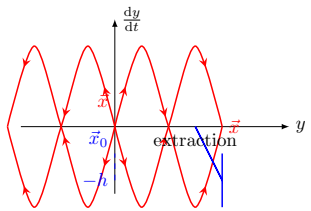
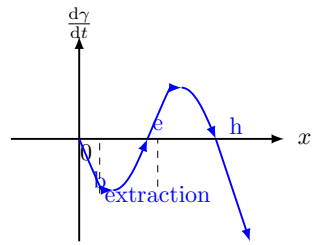


- At first, humans rate GPT-4V outputs higher than DeTikZify from -1 (bad) to 1 (good).
- However, after applying IR with MCTS, DeTikZify ranks highest. Further, IR techniques do not work well with GPT-4.

- With MCTS, DeTikZify consistently improves over time, and even after 10 minutes does not appear to converge.
- Standard sampling does not lead to better scores over time due to the absence of a feedback loop.



Examples

Input	GPT-4V	GPT-4V (IR)	DeTikZify	DeTikZify (IR)
				
				
				

Interested? There's more!

In our paper, we explore various building blocks of DeTikZify and additional baselines. We assess efficiency, code similarity, and image similarity through automatic evaluation, and correlate them with humans. We also examine the quality of synthetic sketches and show that our models are unaffected by training data memorization.

Paper	https://arxiv.org/abs/2405.15306
Code	https://github.com/potamides/DeTikZify
Artifacts	https://hf.co/collections/nllg/detikzify-664460c521aa7c2880095a8b
Demo	https://hf.co/spaces/nllg/DeTikZify
Group	https://nl2g.github.io