BYGPT5

UNIVERSITÄT BIELEFELD

End-to-End Style-conditioned Poetry Generation with Token-free Language Models

Jonas Belouadi and **Steffen Eger**
Natural Language Learning Group (NLLG)

61 ACL 2023 NLLG

Introduction

What is ByGPT5?

ByGPT5 is a language model capable of generating formal verse poetry in the form of quatrains. Users can condition ByGPT5 on aesthetic styles.

The sweet wild strain, the sudden start, A
Which shakes the perfumed altar's flame, B
To make its shrine a sacred name, B
And sing its praise in every heart. A

— ByGPT5

Conditioning on Styles

Definition

Aesthetic styles are strict patterns and rules of language (e.g., rhyme, meter, and alliteration), to which formal verse poetry must conform.

Conditioning on Styles

Definition

Aesthetic styles are strict patterns and rules of language (e.g., rhyme, meter, and alliteration), to which formal verse poetry must conform.

- conditioning a language model on such styles for generating formal verse poetry automatically is a challenging task

Conditioning on Styles

Definition

Aesthetic styles are strict patterns and rules of language (e.g., rhyme, meter, and alliteration), to which formal verse poetry must conform.

- conditioning a language model on such styles for generating formal verse poetry automatically is a challenging task
  - both semantic and aesthetic constraints

Conditioning on Styles

Definition

Aesthetic styles are strict patterns and rules of language (e.g., rhyme, meter, and alliteration), to which formal verse poetry must conform.

- conditioning a language model on such styles for generating formal verse poetry automatically is a challenging task
  - both semantic and aesthetic constraints
  - requires a deep understanding of the meaning of language

Prevalence of Human Guidance

Previous state-of-the-art poetry generation systems rely on human guidance. They either:

Source: de.vecteezy.com

Prevalence of Human Guidance

Previous state-of-the-art poetry generation systems rely on human guidance. They either:

- inject prior knowledge as hard-coded constraints to filter model outputs or modify probability distributions directly

Source: de.vecteezy.com

Prevalence of Human Guidance

Previous state-of-the-art poetry generation systems rely on human guidance. They either:

- inject prior knowledge as hard-coded constraints to filter model outputs or modify probability distributions directly
- break the whole process down into sophisticated task-specific model pipelines

Source: de.vecteezy.com

Prevalence of Human Guidance

Previous state-of-the-art poetry generation systems rely on human guidance. They either:

- inject prior knowledge as hard-coded constraints to filter model outputs or modify probability distributions directly
- break the whole process down into sophisticated task-specific model pipelines

Problems

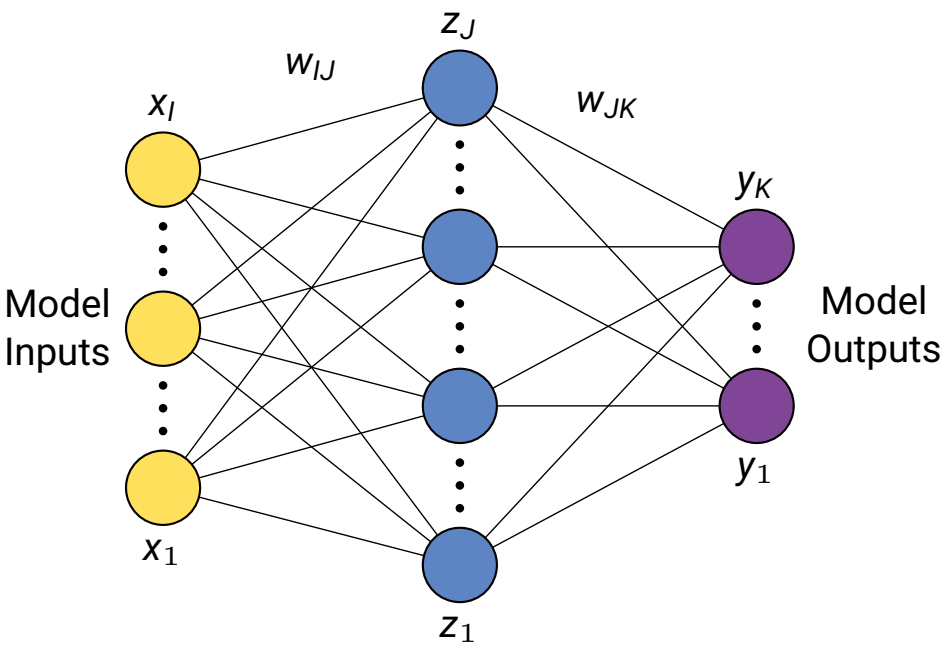
Pipelines are susceptible to cascading of errors, incorporating prior knowledge is hard, autoregressive text-generation architectures do not lend themselves well for manual constraints, …

Source: de.vecteezy.com

In Contrast: End-to-end Learning

Definition

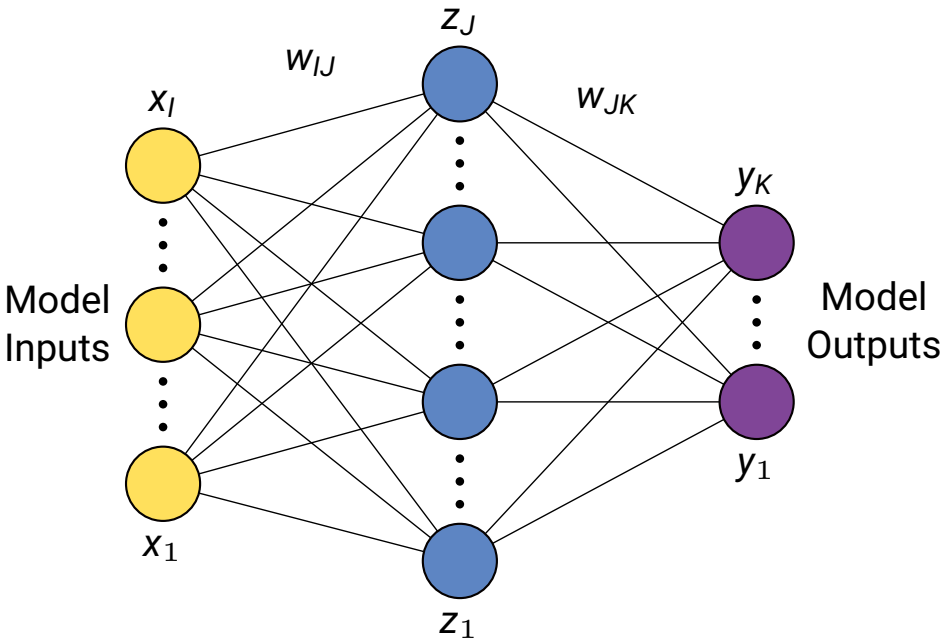
End-to-end models complete a task by directly learning all steps, without intermediary algorithms such as hand-crafted rules or post-processing.



In Contrast: End-to-end Learning

Definition

End-to-end models complete a task by directly learning all steps, without intermediary algorithms such as hand-crafted rules or post-processing.



End-to-end models do not face the ramifications of human guidance!

Research Question

Can we induce an end-to-end poetry generation system which learns to adhere to specified styles by itself without any hard-coding / micro-managing?

Source: Mentalitanissarda, CC BY-SA 3.0, via Wikimedia Commons

Challenges

Past end-to-end attempts were unsuccessful. We identify unfavorable tokenization algorithms and lack of training data as possible root cause.

Source: de.vecteezy.com

Challenges

Past end-to-end attempts were unsuccessful. We identify unfavorable tokenization algorithms and lack of training data as possible root cause.

- aesthetic styles often expressed at the character-level; most pre-trained models operate at the subword-level
  - character-level models might have an advantage on poetry

Source: de.vecteezy.com

Challenges

Past end-to-end attempts were unsuccessful. We identify unfavorable tokenization algorithms and lack of training data as possible root cause.

- aesthetic styles often expressed at the character-level; most pre-trained models operate at the subword-level
  - character-level models might have an advantage on poetry
- existing poetry corpora labeled with styles are few and far between
  - not enough to train a generalized model

Source: de.vecteezy.com

Introducing ByGPT5

	Token-level	Character-level
Encoder-Decoder	mT5	ByT5
Decoder-only	GPT-2	our own

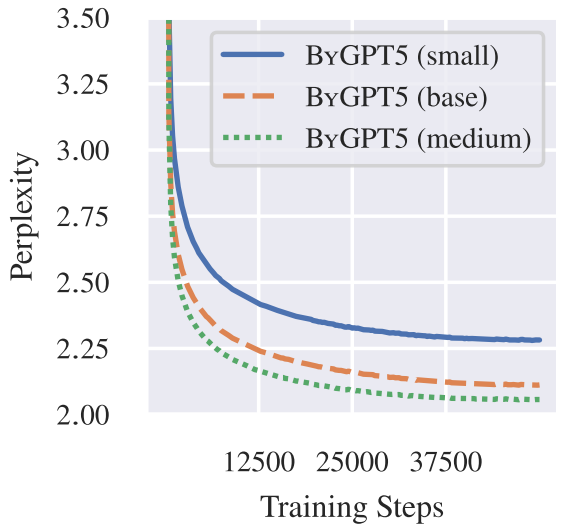
Since we are not aware of existing character-level, generative (decoder-only) models, we train our own.

Introducing BYGPT5

	Token-level	Character-level
Encoder-Decoder	MT5	BYT5
Decoder-only	GPT-2	our own

Since we are not aware of existing character-level, generative (decoder-only) models, we train our own.

Since our model shares some similarities with the GPT family and we initialize the weights with the decoder of BYT5, we refer to it as BYGPT5.



Datasets

We collect small labeled and big unlabeled English & German poetry datasets:

Dataset	Language	Verses	R	M	A
EPG64	English	1k	✓	✓	✗
Prosodic	English	2k	✗	✓	✗
FORB	English	1k	✓	✓	✗
Chicago	English	95k	✓	✗	✗
Anti-K	German	4k	✓	✓	✗
GRC	German	41k	✓	✗	✗
EPG	English	2.8m	✗	✗	✗
DLK	German	2.8m	✗	✗	✗

Datasets

We collect small labeled and big unlabeled English & German poetry datasets:

- we train classifiers on labeled data for labeling bigger, unlabeled datasets automatically*

Dataset	Language	Verses	R	M	A
EPG64	English	1k	✓	✓	✗
Prosodic	English	2k	✗	✓	✗
FORB	English	1k	✓	✓	✗
Chicago	English	95k	✓	✗	✗
Anti-K	German	4k	✓	✓	✗
GRC	German	41k	✓	✗	✗
EPG	English	2.8m	✗	✗	✗
DLK	German	2.8m	✗	✗	✗
QuaTrain	English	2.7m	✓*	✓*	✓*
	German	5.9m	✓*	✓*	✓*

Datasets

We collect small labeled and big unlabeled English & German poetry datasets:

- we train classifiers on labeled data for labeling bigger, unlabeled datasets automatically*
- our new corpus, QuaTrain, contains quatrains labeled with rhyme, meter, and alliteration

Dataset	Language	Verses	R	M	A
EPG64	English	1k	✓	✓	✗
Prosodic	English	2k	✗	✓	✗
FORB	English	1k	✓	✓	✗
Chicago	English	95k	✓	✗	✗
Anti-K	German	4k	✓	✓	✗
GRC	German	41k	✓	✗	✗
EPG	English	2.8m	✗	✗	✗
DLK	German	2.8m	✗	✗	✗
QuaTrain	English	2.7m	✓*	✓*	✓*
	German	5.9m	✓*	✓*	✓*

Datasets

We collect small labeled and big unlabeled English & German poetry datasets:

- we train classifiers on labeled data for labeling bigger, unlabeled datasets automatically*
- our new corpus, QuaTrain, contains quatrains labeled with rhyme, meter, and alliteration

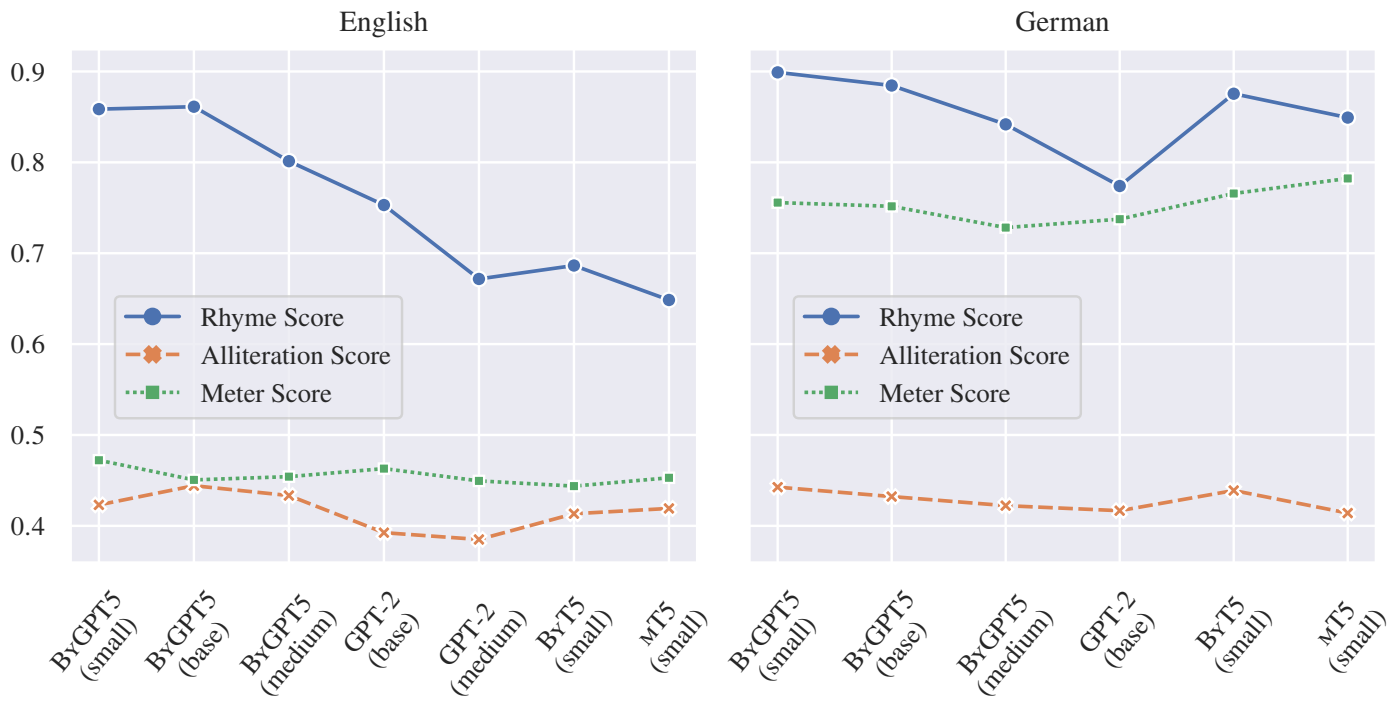
Dataset	Language	Verses	R	M	A
EPG64	English	1k	✓	✓	✗
Prosodic	English	2k	✗	✓	✗
FORB	English	1k	✓	✓	✗
Chicago	English	95k	✓	✗	✗
Anti-K	German	4k	✓	✓	✗
GRC	German	41k	✓	✗	✗
EPG	English	2.8m	✗	✗	✗
DLK	German	2.8m	✗	✗	✗
QuaTrain	English	2.7m	✓*	✓*	✓*
	German	5.9m	✓*	✓*	✓*

Idea

After fine-tuning ByGPT5 on QuaTrain we can reuse the style classifiers for automatic evaluation.

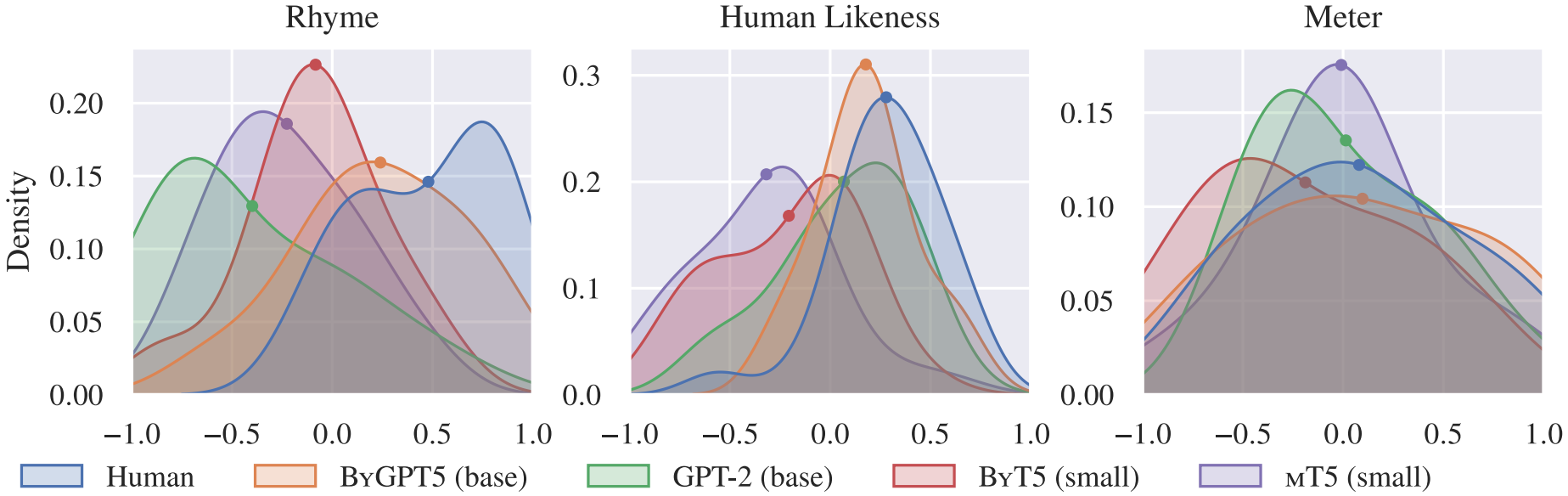
Automatic Evaluation

We find that BYGPT5 learns character-level styles better than subword-based systems, such as GPT-2 and mT5, and even other token-free models like ByT5, while being much more parameter efficient (75% less parameters).



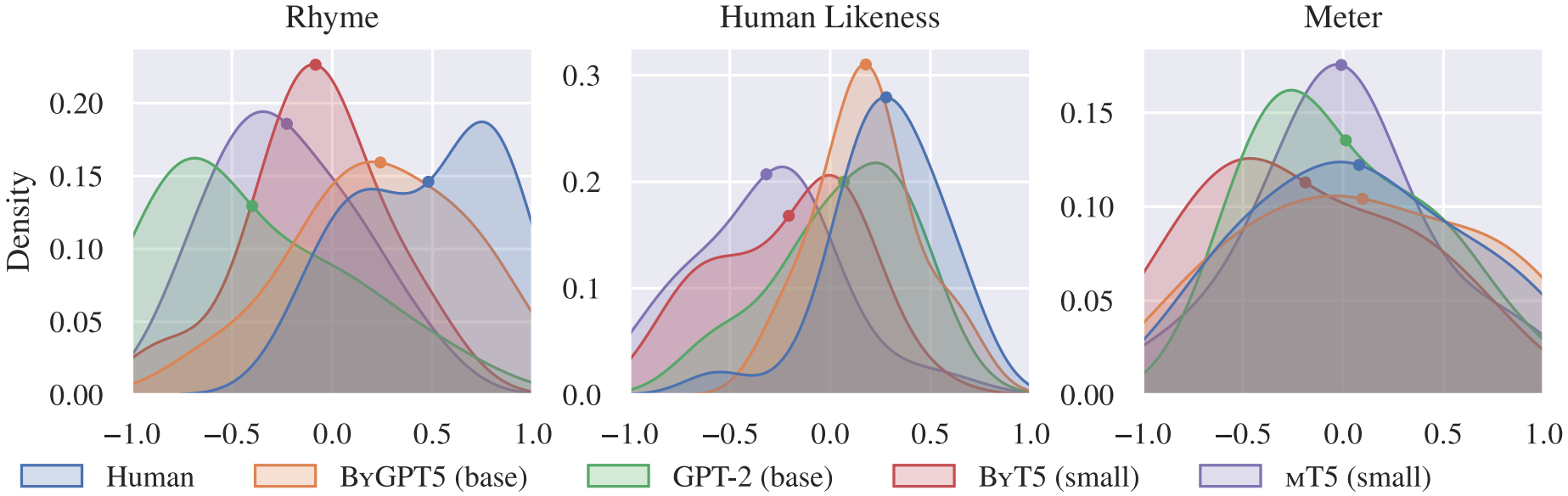
Human Evaluation

We visualize the distributions of human perception of rhyme, human likeness of quatrains, and meter. Scores range from -1 (very bad) to 1 (very good).



Human Evaluation

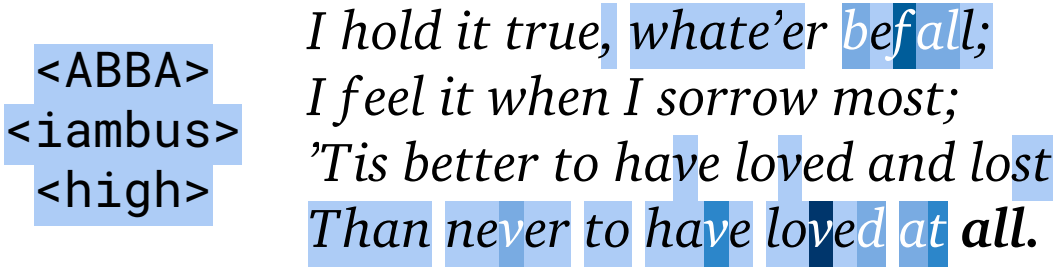
We visualize the distributions of human perception of rhyme, human likeness of quatrains, and meter. Scores range from -1 (very bad) to 1 (very good).



Overall, ByGPT5 performs best among the models. Moreover, ByGPT5 even surpasses humans in the meter property in our constrained setting.

Introspecting Decision-making Processes

To introspect what BYGPT5 has learned, we visualize the contribution of each input token on the output (when generating the last syllable of a line).



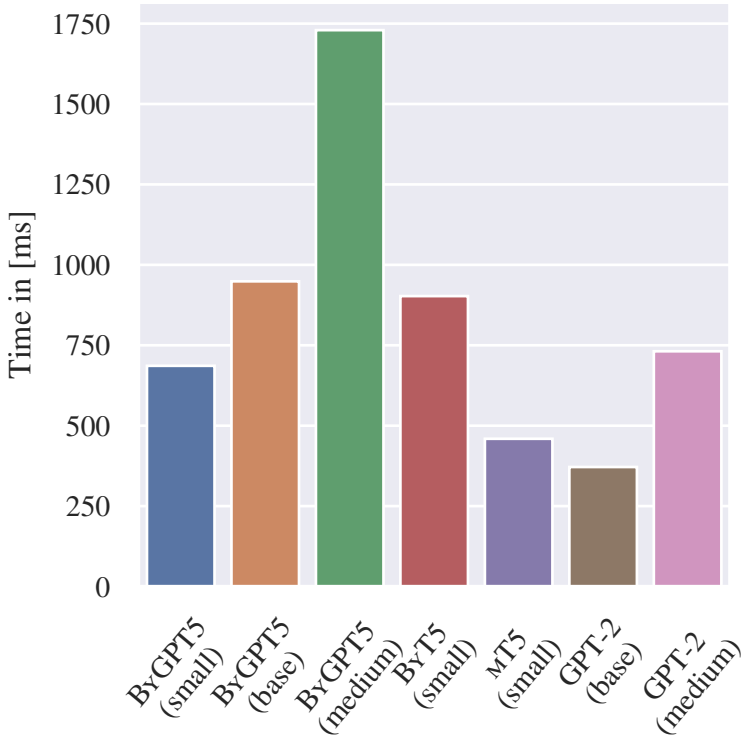
Memorization

Model	English	German
BYGPT5 (small)	0.0%	0.0%
BYGPT5 (base)	0.0%	0.04%
BYGPT5 (medium)	0.0%	0.81%
GPT-2 (base)	0.39%	**1.81%**
GPT-2 (medium)	**3.64%**	–
BYT5	0.0%	0.0%
MT5	0.0%	0.0%

We analyze how many quatrains were copied from the training data. Overall, GPT-2 suffers from memorization the most.

Limitations

Standard transformers become prohibitively slow on very long sequences. Token-free transformers run into these efficiency problems much earlier than subword-based models.



Time and memory for generating a quatrain.

Interested? There's more!

In our paper, we conduct a human evaluation and also demonstrate that all models exhibit few memorization problems and generate novel outputs. However, we also show that GPT-2 and tend to generate simpler code than the other models, leading to less complex images.

Paper `arxiv.org/abs/2212.10474`

Code `github.com/potamides/uniformers`

Demo `colab.research.google.com/drive/1Ast-seDV6_pSzCvpGapNfTV_qDe0KRdF`

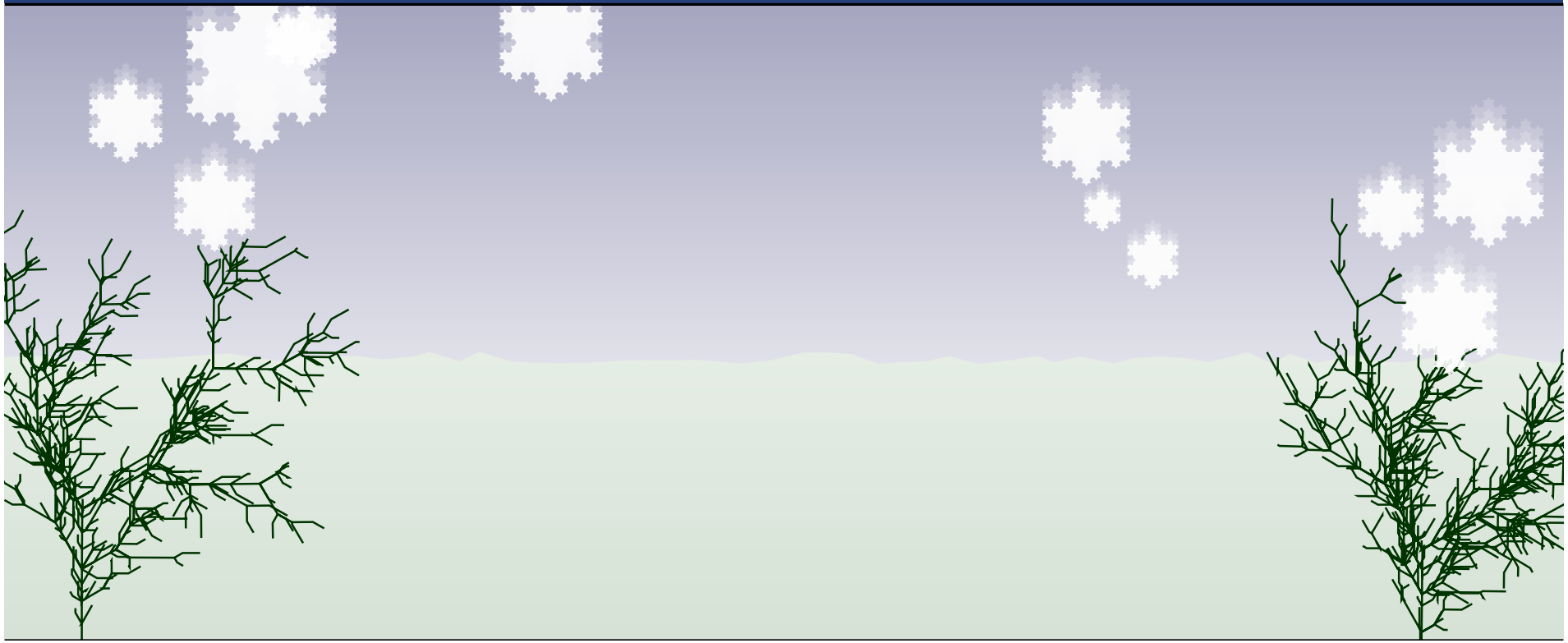
Group `nl2g.github.io`

AutomaTikZ

UNIVERSITÄT BIELEFELD

Text-Guided Synthesis of Scientific Vector Graphics with TikZ

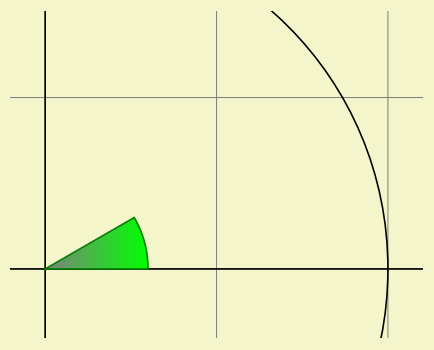
Jonas Belouadi and **Anne Lauscher** and **Steffen Eger**
Natural Language Learning Group (NLLG)



NLLG

The Problem

Generating raster graphics from text is a popular research topic, but in scientific settings vector graphics would be more desirable.

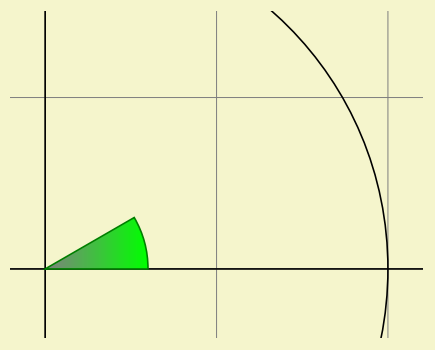


```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0)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

The Problem

Generating raster graphics from text is a popular research topic, but in scientific settings vector graphics would be more desirable.

Problem vector graphics are typically encoded using low-level (SVG) graphics primitives and generating them directly is difficult



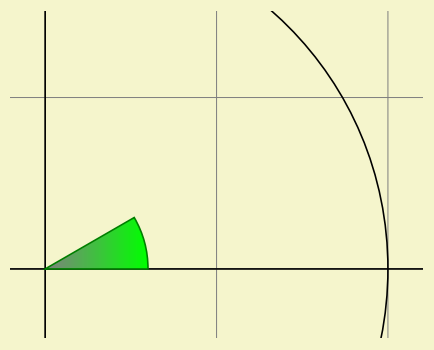
```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0)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

The Problem

Generating raster graphics from text is a popular research topic, but in scientific settings vector graphics would be more desirable.

Problem vector graphics are typically encoded using low-level (SVG) graphics primitives and generating them directly is difficult

Idea use Ti*k*Z, a higher-level programming language that can be compiled to vector graphics, as an intermediate representation



```
\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}
```

Where to find data

Ti*k*Z is well-known within the TEX community, but in general it is still relatively obscure and its resources are scattered. We therefore collected the first large-scale Ti*k*Z dataset of 120k text-Ti*k*Z pairs from various sources:

1. examples from dedicated websites

Where to find data

Ti*k*Z is well-known within the TEX community, but in general it is still relatively obscure and its resources are scattered. We therefore collected the first large-scale Ti*k*Z dataset of 120k text-Ti*k*Z pairs from various sources:

1. examples from dedicated websites
2. code snippets from GitHub projects

Where to find data

Ti*k*Z is well-known within the TEX community, but in general it is still relatively obscure and its resources are scattered. We therefore collected the first large-scale Ti*k*Z dataset of 120k text-Ti*k*Z pairs from various sources:

1. examples from dedicated websites
2. code snippets from GitHub projects
3. questions from TEX StackExchange

Where to find data

Ti*k*Z is well-known within the TEX community, but in general it is still relatively obscure and its resources are scattered. We therefore collected the first large-scale Ti*k*Z dataset of 120k text-Ti*k*Z pairs from various sources:

1. examples from dedicated websites
2. code snippets from GitHub projects
3. questions from TEX StackExchange
4. Ti*k*Z-caption pairs from arXiv papers

Where to find data

Ti*k*Z is well-known within the TEX community, but in general it is still relatively obscure and its resources are scattered. We therefore collected the first large-scale Ti*k*Z dataset of 120k text-Ti*k*Z pairs from various sources:

1. examples from dedicated websites
2. code snippets from GitHub projects
3. questions from TEX StackExchange
4. Ti*k*Z-caption pairs from arXiv papers
5. simple, tangible objects generated with GPT-4

Data Augmentation

All texts should describe their Ti*k*Z pictures (i.e., be captions), but some have the wrong format or are too vague, so we did some data augmentation:

Data Augmentation

All texts should describe their Ti*k*Z pictures (i.e., be captions), but some have the wrong format or are too vague, so we did some data augmentation:

TEX StackExchange We use a language model to rephrase each question into a description of the desired result.

Data Augmentation

All texts should describe their Ti*k*Z pictures (i.e., be captions), but some have the wrong format or are too vague, so we did some data augmentation:

TEX StackExchange We use a language model to rephrase each question into a description of the desired result.

arXiv Some captions are too short (arXiv affected the most) to describe the image in a meaningful way. We use a vision language model to enhance them with additional visual information.

Basic Approach

Since Ti*k*Z is implemented with TEX macros, a foundational language model capable of understanding it would be desirable.

Basic Approach

Since Ti*k*Z is implemented with TEX macros, a foundational language model capable of understanding it would be desirable.

- surprisingly models that see TEX during pre-training are pretty rare

NLLG

Basic Approach

Since Ti*k*Z is implemented with TEX macros, a foundational language model capable of understanding it would be desirable.

- surprisingly models that see TEX during pre-training are pretty rare
- however, the recent LLAMA model was trained on arXiv and TEX StackExchange, so it qualifies.

Basic Approach

Since Ti*k*Z is implemented with TEX macros, a foundational language model capable of understanding it would be desirable.

- surprisingly models that see TEX during pre-training are pretty rare
- however, the recent LLAMA model was trained on arXiv and TEX StackExchange, so it qualifies.

Idea

Fine-tune LLAMA on our dataset and see how it performs.

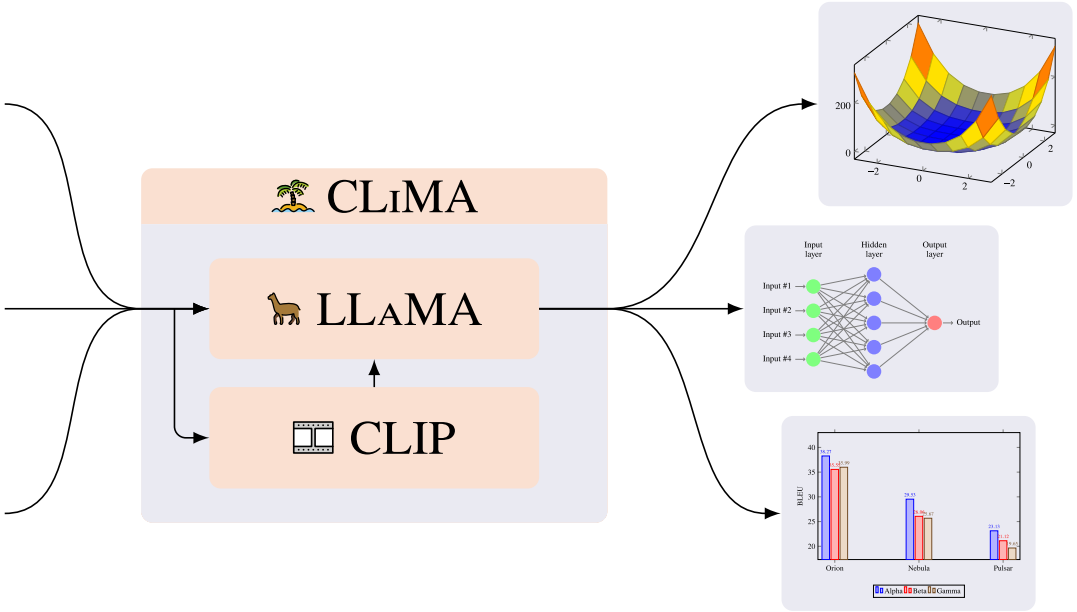
LLaMA learns to see

We hypothesize that incorporating a vision encoder like CLIP could boost our model. We thus incorporate CLIP using its multi-modal projection as a soft prompt, which should allow us to extract visual information from the input text. We call this model CLiMA (CLIP inside LLaMA).

3D contour plot of a loss function, showcasing global and local minima. The color gradient indicates function depth, providing insight into the optimization challenges in machine learning.

Visual representation of a multi-layer perceptron: an interconnected network of nodes, showcasing the structure of input, hidden, and output layers that facilitate complex pattern recognition.

Bar chart comparing BLEU scores of Alpha, Beta, and Gamma models across Orion, Nebula, and Pulsar datasets, with Alpha consistently leading.



Experiments

We measure the performance of our models on a held-out test set using a range of automatic metrics:

code-based CrystalBLEU, Extended-Edit Distance (EED)

Experiments

We measure the performance of our models on a held-out test set using a range of automatic metrics:

- code-based: CrystalBLEU, Extended-Edit Distance (EED)
- image-based: CLIPScore, $\text{CLIPScore}_{\text{IMG}}$, Kernel Inception Distance (KID)

Experiments

We measure the performance of our models on a held-out test set using a range of automatic metrics:

- code-based: CrystalBLEU, Extended-Edit Distance (EED)
- image-based: CLIPScore, $\text{CLIPScore}_{\text{IMG}}$, Kernel Inception Distance (KID)
- sampling-based: Compilation Sampling Rate (CSR)

Experiments

We measure the performance of our models on a held-out test set using a range of automatic metrics:

- code-based: CrystalBLEU, Extended-Edit Distance (EED)
- image-based: CLIPScore, $\text{CLIPScore}_{\text{IMG}}$, Kernel Inception Distance (KID)
- sampling-based: Compilation Sampling Rate (CSR)

We further evaluate the following range of models:

- LLAMA: variants with 7 billion $\text{LLAMA}_{7\text{B}}$ and 13 billion $\text{LLAMA}_{13\text{B}}$ parameters

Experiments

We measure the performance of our models on a held-out test set using a range of automatic metrics:

- code-based: CrystalBLEU, Extended-Edit Distance (EED)
- image-based: CLIPScore, $\text{CLIPScore}_{\text{IMG}}$, Kernel Inception Distance (KID)
- sampling-based: Compilation Sampling Rate (CSR)

We further evaluate the following range of models:

- LLaMA: variants with 7 billion $\text{LLaMA}_{7\text{B}}$ and 13 billion $\text{LLaMA}_{13\text{B}}$ parameters
- CLiMA: $\text{CLiMA}_{7\text{B}}$ and $\text{CLiMA}_{13\text{B}}$, as well as $\text{CLiMA}_{\text{IMG}}$ which has the compiled images as an additional input

Experiments

We measure the performance of our models on a held-out test set using a range of automatic metrics:

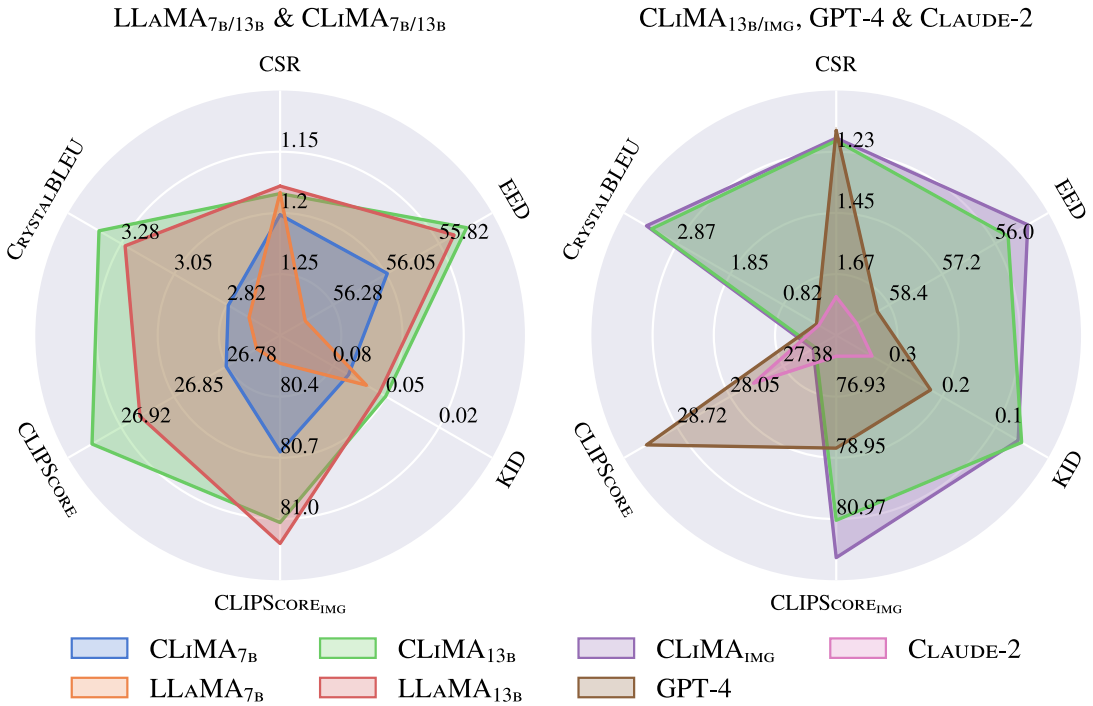
- **code-based** CrystalBLEU, Extended-Edit Distance (EED)
- **image-based** CLIPScore, $\text{CLIPScore}_{\text{IMG}}$, Kernel Inception Distance (KID)
- **sampling-based** Compilation Sampling Rate (CSR)

We further evaluate the following range of models:

- **LLaMA** variants with 7 billion $\text{LLaMA}_{7\text{B}}$ and 13 billion $\text{LLaMA}_{13\text{B}}$ parameters
- **CLiMA** $\text{CLiMA}_{7\text{B}}$ and $\text{CLiMA}_{13\text{B}}$, as well as $\text{CLiMA}_{\text{IMG}}$ which has the compiled images as an additional input
- **general-purpose models** GPT-4 and Claude 2

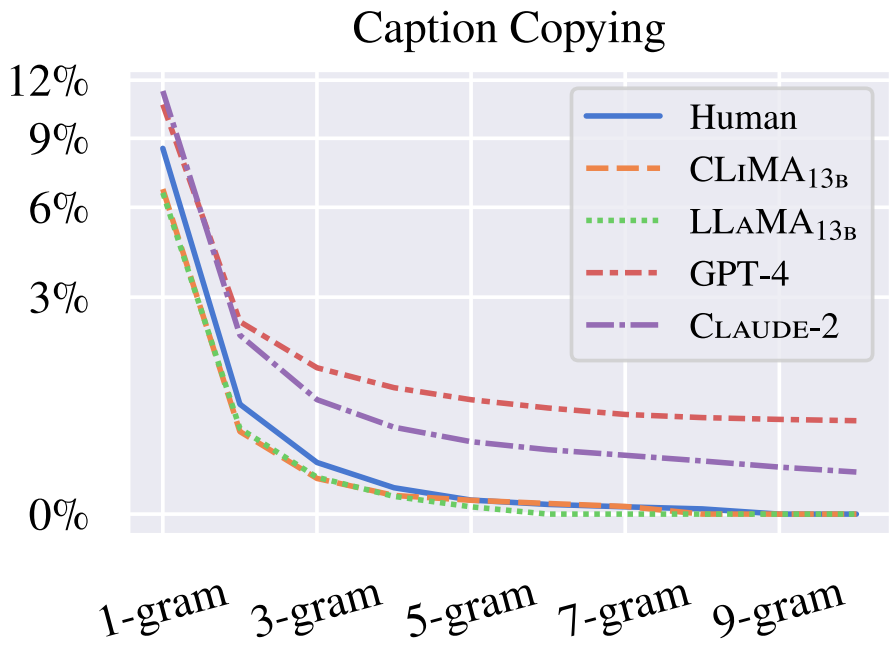
Results

Overall, CLIMA$_{7B}$ and CLIMA$_{13B}$ outperform their respective LLAMA models in five out of seven metrics each, with CLAUDE 2 and GPT-4 substantially underperforming all of them. While CLIMA$_{IMG}$ unsurprisingly improves upon CLIMA$_{13B}$, CLIMA$_{13B}$ is the best model with only textual inputs.

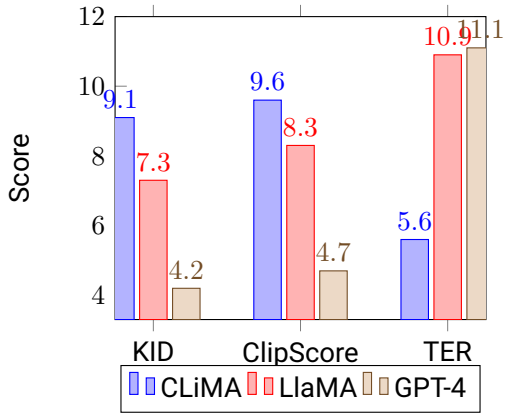


Caption Copying

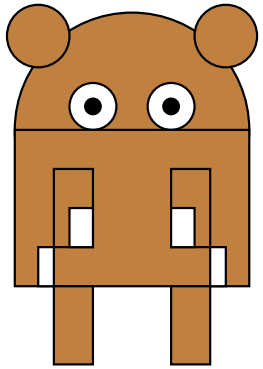
We find that GPT-4 and CLAUDE 2 tend to produce degenerate images, which visibly copy n-grams of the input caption into the output image. This can trick CLIPScore into providing scores.



Examples

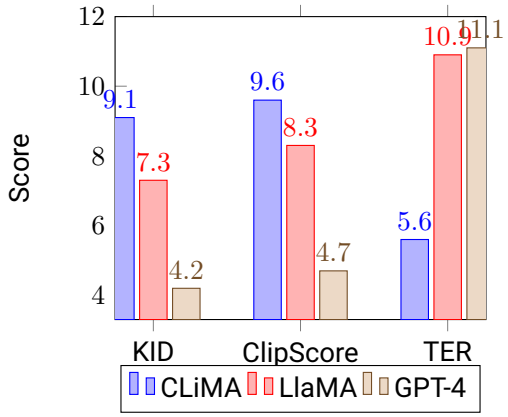


Prompt: A bar chart of the performance of CLiMA, LlaMA, and GPT-4. As can be seen CLiMA scores highest on KID, ClipScore, and TER.

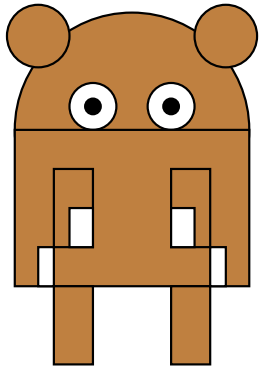


Prompt: A cute bear. The image shows a teddybear, it consists of a circle for the head and a larger elipsis for the main body.

Examples



Prompt: A bar chart of the performance of CLiMA, LlaMA, and GPT-4. As can be seen CLiMA scores highest on KID, ClipScore, and TER.

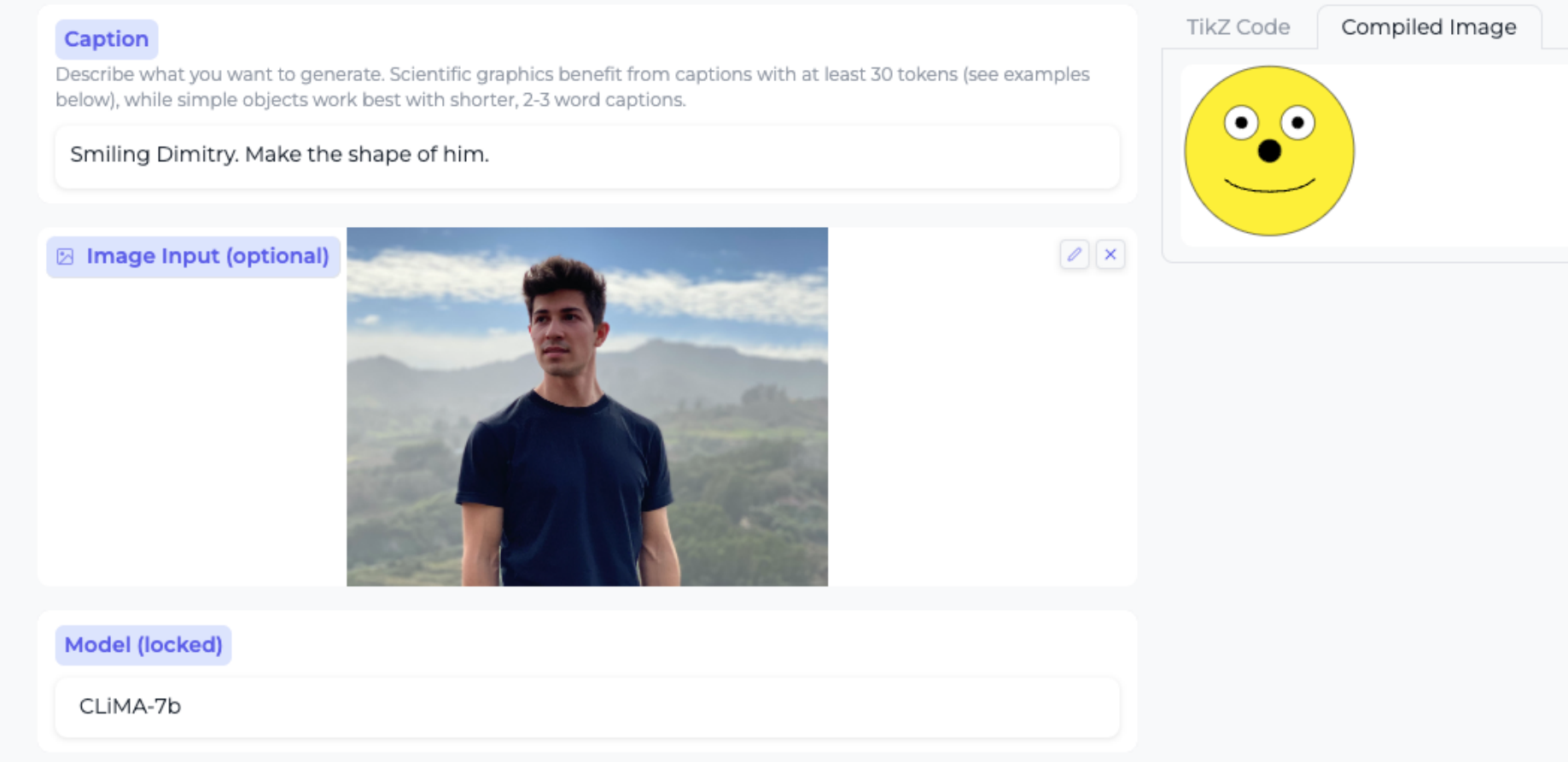


Prompt: A cute bear. The image shows a teddybear, it consists of a circle for the head and a larger elipsis for the main body.

Although it generates promising results, the model does not fully incorporate all aspects of the prompt with complete accuracy.

Examples

CLiMA's image input can also be used to vectorize images… sort of.



Interested? There's more!

In our paper, we conduct a human evaluation and also demonstrate that all models exhibit few memorization problems and generate novel outputs. However, we also show that GPT-4 and CLAUDE 2 tend to generate simpler code than the other models, that compiles to less complex images.

Paper `https://arxiv.org/abs/2310.00367`

Code `https://github.com/potamides/AutomaTikZ`

Demo `https://huggingface.co/spaces/nllg/AutomaTikZ`

Group `https://nl2g.github.io`