University of Bamberg Press

Joachim, Silvia; Hennecke, Martin

MAL-E : Understanding Text-to-Image Generation

In:

Schmid, Ute; Leidner, Jochen L.; Kohlhase, Michael; Wolter, Diedrich (Eds.), Proceedings of the Second Work shop on Artificial Intelligence for Artificial Intelligence Education (AI4AI Learning 2024), Bamberg: University of Bamberg Press, p. 23-32. 2025. DOI: 10.20378/irb-107661

Bookpart - Published Version

DOI of the Article: 10.20378/irb-108885

Date of Publication: 07.07.2025

Legal Notice:
This work is protected by copyright and/or the indication of a licence. You are free to use this work in any way permitted by the copyright and/or the licence that applies to your usage. For other uses, you must obtain permission from the rights-holder(s).

This document is made available under the **Creative Commons Lizense CC BY.**

This licencse information is available online:
https://creativecommons.org/licenses/by/4.0/

MAL-E: Understanding Text-to-Image Generation

Silvia Joachim and Martin Hennecke

University of Würzburg, Computer Science Education

{silvia.joachim, martin.hennecke}@uni-wuerzburg.de

Abstract

Generative AI, particularly in image generation, has attracted a lot of attention in recent years. These technologies are here to stay. Understanding how they work demystifies them, showing that they are driven by algorithms, not magic. We present a learning and experimentation module for Unplugged Text-To-Image Generation, which we have called MAL-E. It explains several key steps in the process, starting with Tokenization, Embedding, and Positional Encoding. Students learn Masked Self-Attention, an important step of the Decoder-only Transformer. They also learn about Image Components Selection and Image Assembly to produce the final image. MAL-E provides an inclusive hands-on way to understand how pre-trained neural networks and transformers create images from text prompts.

Keywords: Artificial Intelligence · CS Unplugged · Inclusive Material · Generative AI · AI Education · Neural Networks · K-12 students.

Introduction

The ability to quickly create individual and professional images plays a crucial role in various aspects of life. With the right software, generating images from text is straightforward. For example, a prompt like “Draw a mushroom with stripes” directly produces an image of a mushroom with stripes. However, if you draw a mushroom yourself and then provide the software with a photo of that mushroom alongside the prompt “I want this exact mushroom, but with a striped pattern”, the resulting image will indeed show a striped mushroom. In many tools, the resulting image may

differ from what one might have originally envisioned. To understand this, you need to understand at least a few basic ideas of the text-to-image generation. Incidentally, mushrooms are a recurring example in our materials for teaching various AI algorithms [3, 8, 9, 10], as they work very intuitively for almost all age groups.

In this paper, we introduce an unplugged approach designed to explain the concept of text-to-image generation using neural networks to an audience without a deeper background in AI, which includes high school students, university students without AI focus, and computer science teachers. The idea of teaching CS content without computers goes back to [4, 5] and has proved to be very successful in many areas. In the AI competency framework for students [14], unplugged learning materials are mentioned as one of the ways to support understanding AI. The training phase of the neural network is beyond the scope of this learning module, as our primary objective is to explain how the pre-trained model transforms text into images. Some AI-systems use only the decoder part of the transformer architecture. We do this also and simplify the steps involved in text-to-image generation to effectively teach the fundamental concepts. To ensure inclusivity for visually impaired students, we use materials with tactile features.

Unplugged Text-to-Image Generation

Understanding generative AI currently plays no role in the curricula of German schools (even if its application does). Initial work in CS education has focused on continuing character strings, sound sequences or hand drawings with Markov chains and making this functionality teachable [6, 15]. There are also examples of unplugged material here [11, 12]. [1] presents a training unit for teachers that also includes a technical explanation of text-to-image generation. Based on infographics, the unit primarily explains the diffusion model, while our learning module focuses on the Decoder-only Transformer architecture and emphasizes unplugged activities. It is challenging to count this as related work because the underlying technology and the generation processes differ significantly.

Of course, there are numerous academic articles on text-to-image generation (e.g. [17, 16]), well-known books [18] that at least outline this aspect, lecture manuscripts, etc. What these sources have in common is that they try to represent the generative process as well as possible and avoid simplification where possible. Since the aim of these sources is to present the state of the authors' work to other scientists or to explain to students with a particular interest in AI how they can develop such systems themselves, this approach is the right one. For our audience, such sources are usually too complex. Here a simplification is needed that preserves the essential functionalities of the real process but is understandable at the level of the audience.

Teaching Concept of MAL-E

MAL-E offers two levels of difficulty and includes individual and team activities. The name is derived from the German word for "paint", with its spelling paying homage to DALL-E [2], a practical tool in the field of text-to-image generation. One of our key challenges was designing a module that would exclude as few learners as possible. To achieve this, we opted for special unplugged materials. The game pieces feature tactile dice numbers, and the game board provides a stable base, allowing for the addition of Braille numbers to accommodate visually impaired students. We deliberately chose to work with only small numbers, and we kept everything within a two-dimensional framework. The transparent game boards and game pieces are available through the AI Experiment Kit [7]. If you want to do MAL-E without the desirable tactile properties, the material can also be realized in the form of printed worksheets. As the most important learning prerequisite, learners should already know how feedforward networks work, because they use simple pre-trained networks for MAL-E.

Activities in MAL-E

First, students divide into groups of four. Each group selects a slip of paper with a prompt. Different prompts, such as "Draw mushroom with stripes" or "Draw mushroom with flowers" can be chosen, or the same prompt can be selected multiple times. This prompt serves as the basis for the activities tokenization, embedding, positional encoding, masked self-attention, image components selection and image assembly.

Activity 1: Tokenization We consider the prompt "Draw mushroom with stripes". There are multiple ways to approach tokenization. We simply use the words of the prompt as tokens. In this activity there are game pieces with different words, from which the team selects the required tokens.

Activity 2: Embedding Neural networks require vectors as input. The goal of the Embedding activity is to assign a two-dimensional vector to each token. The students use a prepared coordinate system (figure 1 left) where various tokens are already placed. The closer the tokens are in meaning, the nearer they are positioned to each other. For example, the tokens "mushrooms" and "dots" are closer together than the tokens "mushrooms" and "stripes". For visually impaired students, only the tokens required for the specific prompt can be used, and instead of words, game pieces with the corresponding number of dots representing the position are utilized. Instead of coordinate axes, a rubber band can be stretched.

Activity 3: Positional Encoding The meaning of a prompt can change when the order of words is rearranged. For example, the prompts "Draw mushroom with stripes" and "Draw stripes with mushroom" have different meanings, even though they contain the same words, and would result in completely different images. Transformer models do not have a built-in sequence order for input, so positional encoding is added to represent the position of each token in the sequence. We simply add the vector (1,1) to the first token, (2,2) to the second token, and so on. This simplification ensures that we work only with small natural numbers while remaining in two dimensions, which allows us to represent the process enactively using game pieces for hands-on learning. The students read the coordinates for each of the used tokens from the left coordinate system in figure 1. The right coordinate system in figure 1 is initially empty. After performing the positional encoding, the students place their tokens at the corresponding positions in the right coordinate system.

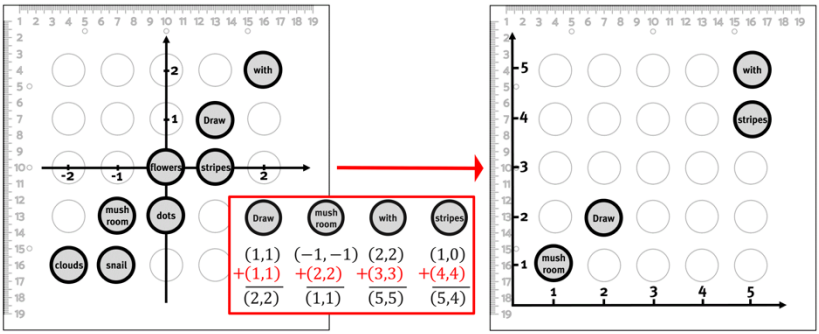

Fig. 1. Embedding and result after Positional Encoding for the selected prompt

Activity 4: Masked Self-Attention The unplugged materials are structured to visually demonstrate the computation of the Masked Self-Attention mechanism step-by-step. This is the central step in the text-to-image generation process, where the Decoder-only Transformer calculates the relationships between tokens in the sequence, ensuring that each token only considers the tokens that came before it. Masked Self-Attention means each token can only "see" the previous tokens, e.g. in the prompt "Draw mushroom with stripes" the token "mushroom" can see itself and "Draw", but not "with" and "stripes". Each student selects a token and takes one of four wooden tablets. Each tablet is equipped with the same game board and some game pieces with dice numbers. The tablets are large enough to allow the board to be shifted down during the activity.

The chosen token is placed in the top left corner of the board. Beneath the transparent board, there is a representation of three feed-forward networks with two input fields. The students places game pieces with the corresponding number of dots (representing the vector after Positional Encoding) in the fields at the top.

Query, Key, and Value are calculated using the neural networks. The activation function used in MAL-E is the identity function, which means the output of each neuron is the same as the combined input. These results are recalculated by each student for their token. The students place game pieces with the corresponding number of dots into the output fields, representing the results of the calculations. Each student performs the same

calculations on their respective tablets, but each uses a different input vector. Figure 2 shows the results for the first two tokens. While each student works on their own, they can help one another, as the process is fundamentally the same for everyone.

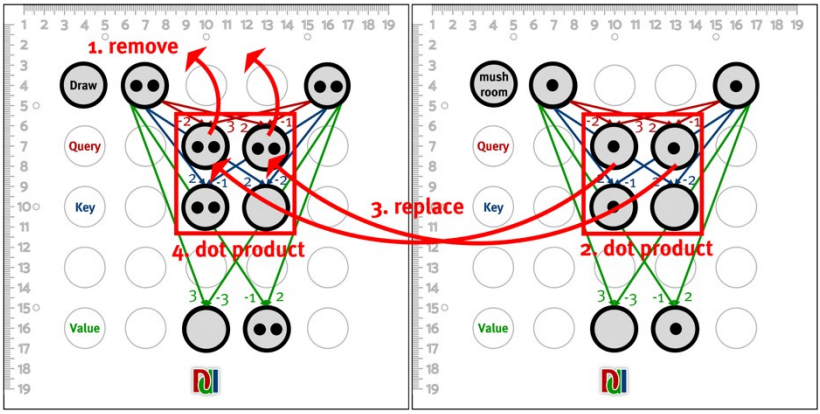

Fig. 2. Part of the process of Masked Self-Attention

The Query vector of a token represents the kind of information this token is looking for. The Key vector represents the kind of information that token holds. The alignment between the Query and Key vector can be mathematically calculated using the dot product. After each student has completed their game board, they place their tablets side by side in the order of the tokens in the prompt, forming a large square. The game boards are arranged diagonally, starting from the top left. The students now work together as a team. First, the first token “Draw” is considered. “Draw” can only see itself, so only the dot product between the Query and Key of token 1 is calculated. Next, the second token is considered. To do this, the game board of token 1 is placed next to the game board of token 2, as shown in figure 2. Token 2 considers both itself and Token 1. The game pieces representing the Query vector of Token 1 are no longer needed and are removed from the board. We now calculate how Token 2 perceives itself by computing the dot product between its own Query and Key vectors. Next, the game pieces from Token 2’s Query vector are placed in the fields for the Query vector of Token 1. This allows us to compute the dot product for Token 1’s board, giving us a result that represents how Token 2 perceives Token 1. To consider the third token, the two game boards are

shifted down next to the board of token three. Now, only the Query game pieces from token three are needed, and the three dot products are calculated accordingly. The boards are then shifted down to consider token four, using only the Query game pieces from token four. The dot product can also be thought of as the angle between the original vectors pointing to the coordinates (x, y) of Query and (x, y) of Key. The students can use an Excel spreadsheet to apply the Softmax function to normalize the attention scores (dot products). Finally, the normalized attention scores are multiplied by the Value vectors to obtain the final vectors. For younger students and students with visual impairments, the Max function can be used instead of the Softmax function for certain prompts, allowing the final vectors to be calculated mentally.

Activity 5: Image Components Selection The previously explained steps assign a final vector to each token. For example, we obtain the vector (3,3) for the token "stripes". The students therefore choose a box labeled (3,3). In this box they find different parts of the picture, which are transparent with engraved lines. Each of the picture parts has one of the markings 1, 2, 3 or 4. The students may now select a picture part from all the picture parts with the marking 1. They do the same with the picture parts marked 2 to 4. Once the students have chosen a picture part for all the tokens, they move on to the next activity.

Activity 6: Image Assembly The decoder generates the final image. The students overlay and arrange the transparent sheets side by side. They place all sheets labeled with the number 1 in the top right corner, number 2 in the bottom right, number 3 in the bottom left, and number 4 in the top left. Figure 3 shows how these transparent sheets can be combined to achieve the final composition of a new mushroom with stripes, giving the impression of a creative process.

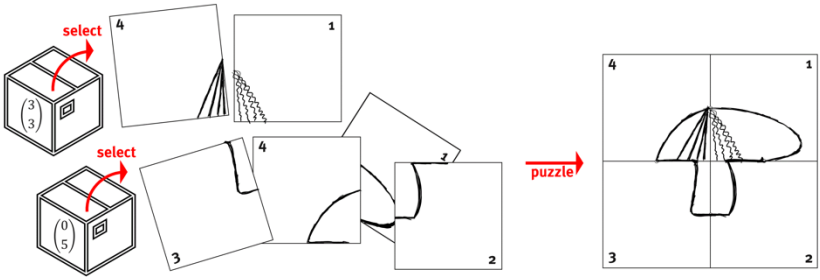

Fig. 3. Process of image component selection and assembly

Conclusion

The unplugged material presented here to explain text-to-image generation is part of a larger teaching sequence on AI. We found it necessary to create an enactive approach for this topic as well. Teachers should be aware of possible algorithmic biases [13]. By MAL-E, the picture parts in the boxes represent predefined components that the system selects based on its prior training data. If the boxes only contain certain parts or styles, the final output will be biased towards those, limiting the variety of images. Enactive materials have proven to be particularly effective in introducing new topics into the classroom.

MAL-E was tested with students and received very positive feedback. We conducted interviews based on self-assessment of their knowledge. All participants confirmed that MAL-E can effectively helps to understand how AI systems generate an image from a text prompt. MAL-E is an engaging and motivating learning module that introduces students to the workings of generative AI in image creation. It encourages students to understand how things work. Along the way, they learn about algorithmic thinking and the basics of neural networks. Further tests are planned as part of teacher training programs.

References

1. Ali, S., Ravi, P., Moore, K., Abelson, H., Breazeal, C.: A picture is worth a thousand words: Co-designing text-to-image generation learning materials for K-12 with educators. Proceedings of the AAAI Conference on Artificial Intelligence 38, 23260–23267 (03 2024), https://doi.org/10.1609/aaai.v38i21.30373

2. Amatriain, X.: Transformer models: an introduction and catalog (02 2023), https://doi.org/10.48550/arXiv.2302.07730

3. Andres, D., Joachim, S., Hennecke, M.: Den k-Means-Algorithmus verstehen: Mit Stift & Papier und BlueJ. Informatische Bildung in Schulen 2(1) (2024), https://doi.org/10.18420/ibis-02-01-06

4. Bell, T., Rosamond, F., Casey, N.: Computer Science Unplugged and Related Projects in Math and Computer Science Popularization, pp. 398–456. Springer Berlin Heidelberg, Berlin, Heidelberg (2012), https://doi.org/10.1007/978-3-642-30891-8_18

5. Bell, T.C.: Teaching teachers to teach computer science - unplugged or plugged-in? Proceedings of the 2020 ACM Conference on International Computing Education Research (2020), https://api.semanticscholar.org/CorpusID:221035199

6. Hielscher, M.: SoekiaGPT - ein didaktisches Sprachmodell. Informatische Bildung in Schulen 1(1) (2023), https://doi.org/10.18420/ibis-01-01-04

7. Joachim, S.: Experimentiersatz Künstliche Intelligenz. MEKRUPHY GMBH (2023), https://mekruphy.com/de/ki

8. Joachim, S.: Experimentiersatz KÜNSTLICHE INTELLIGENZ - Experimentierheft. MEKRUPHY GMBH (2023).

9. Joachim, S.: Experimentiersatz KÜNSTLICHE INTELLIGENZ - Handreichung für die Lehrkraft. MEKRUPHY GMBH (2023).

10. Joachim, S., Hennecke, M.: Enaktive Bestimmung der Hyperparameter beim Entscheidungsbaum- und k-nächste-Nachbarn-Algorithmus. In: INFORMATIK 2023 - Designing Futures: Zukünfte gestalten, pp. 415–418. Gesellschaft für Informatik e.V., Bonn (2023), https://doi.org/10.18420/inf2023_47

11. Joachim, S., Hennecke, M.: Reinforcement-Learning enaktiv und inklusiv vermitteln. INFOS 2023 - Informatikunterricht zwischen Aktualität und Zeitlosigkeit (2023), https://doi.org/10.18420/infos2023-049

12. Lindner, A., Seegerer, S., Romeike, R.: Unplugged activities in the context of AI. In: Informatics in Schools. New Ideas in School Informatics: 12th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP 2019, Larnaca, Cyprus, November 18–20, 2019, Proceedings. p. 123–135. Springer-Verlag, Berlin, Heidelberg (2019), https://doi.org/10.1007/978-3-030-33759-9_10

13. Miao, F., Cukurova, M.: AI competency framework for teachers. UNESCO (2024), https://doi.org/https://doi.org/10.54675/ZJTE2084

14. Miao, F., Shiohira, K.: AI competency framework for students. UNESCO (2024), https://doi.org/https://doi.org/10.54675/JKJB9835

15. Mönig, J.: Snap!GPT – Bausteine für generative künstliche Intelligenz. Informatische Bildung in Schulen 2(1) (2024), https://doi.org/10.18420/ibis-02-01-04

16. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with CLIP latents. ArXiv abs/2204.06125 (2022), https://api.semanticscholar.org/CorpusID:248097655

17. Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., Sutskever, I.: Zero-shot text-to-image generation. ArXiv abs/2102.12092 (2021), https://api.semanticscholar.org/CorpusID:232035663

18. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach (4th Edition). Pearson (2020), http://aima.cs.berkeley.edu/