https://doi.org/10.2352/CIC.2025.33.1.36


# LLMs speak LAB

*Ján Morovič, Peter Morovič, HP Inc.*

## Abstract

*Large Language Models (LLMs) are advanced neural networks designed to interpret and generate human-like text thanks to their structure and to having been trained on vast amounts of data. They can perform a wide range of natural language processing tasks, including text generation, translation, summarization, and question-answering, and are the engines of conversational AI platforms like ChatGPT, Gemini or Claude. A key feature of such LLMs is their inference of a subsequent piece of text from preceding pieces of text. As such, their computational structure lends itself to making other, similar sequential inferences. While the acquisition of color measurements may at first seem far removed from the domain of LLMs, it too can be thought of as a sequential process, consisting of the measurement of a sequence of stimuli, and therefore open to sequential inference. The present paper introduces an adaptation of LLMs to color data, and more broadly to sensor data, and their application to generating measurements from a preceding sequence, based on pre–training transformers with sensor data sequences. Promising first results are shared that point to low color differences when models are prompted with similar magnitude data to those being constructed using generative AI (GenAI).*

## Introduction

AI has long been used to perform a variety of tasks in the context of color and imaging. Among these, the modelling of color imaging systems and of color constancy are among the most extensively explored, with successful applications of both multi-layer perceptron (MLP) [1], [2] and Kolmogorov Arnold neural networks (KANs) [3], [4]. What these applications have in common though is that an AI is built to learn the relationship between inputs to an imaging system and its outputs (e.g., RGB or CMYK to LAB), where that relationship is static. In essence the precondition is for the modelled system to be in a stable state and for a model to then learn its behavior. Given a successfully trained model, outputs can be inferred from inputs and vice versa, which allows both for making predictions about outputs and controlling inputs.

A different, hitherto unaddressed, challenge is to model the behavior of a variable system. While previous approaches either assumed a fixed system in a stable state or relied on for that system being calibrated and therefore stabilized, an unresolved challenge is how to take advantage of AI in assisting with color calibration. Instead of there being a single relationship that is to be learned and modelled, the challenge is to make predictions about the variable state of an imaging system.

This challenge can be re-framed as a sequential inference challenge, where the question is whether AI can be used to take some measurements from a system in an unknown, varying state, and infer from them further measurements that would be obtained on that same system in its unknown state. In other words, the question is, whether relationships among measurements under varying, unknown states of an imaging system can be successfully learned. If the answer is yes, then this would open the way to being able to color calibrate or profile them based on fewer measurements than are needed today.

Since it turns out that this question is fundamentally about sensor data, of which color sensor data is just an example, the following section will provide background on the use of LLMs with such data and an introduction to the generative pretrained transformer (GPT) model that they use and that will be adapted for color here. Next, the use of LLMs with color measurements will be attempted by treating them like any other text, followed by an introduction to modifying a key LLM component, the tokenizer, to make it tuned to color measurement data. Results of inference color accuracy will be shared along the way and the paper will finish with conclusions and a sketch of next steps.

## Background

As a preamble to exploring the use of LLMs for making inferences in color measurement sequences, background will be provided here on two areas: the structure of GPT models and the use of LLMs (that build on GPT) with sensor data.

### *Generative Pretrained Transformers*

Unlike MLP neural networks and KANs, where an input is transformed via a series of multiple layers of neurons to yield an output, the key to GPTs [5] is that they operate on a sequence of inputs, which allows them to vary their inferences based on input (prompt) context.

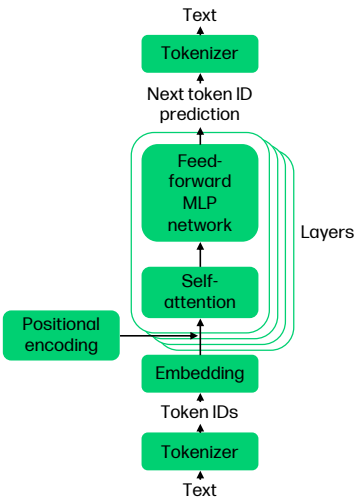


***Figure 1.*** *Key elements of a generative pretrained transformer model.*

The way in which a GPT takes a text prompt and infers subsequent text is the following (Fig. 1). First, text is tokenized, i.e., broken down into smaller units, tokens, which can be words, sub–words, or even characters. Each token within a token vocabulary is put in one-to-one correspondence with a unique numerical identifier.

Second, each token is converted into a dense vector representation in a multi-dimensional space through a process known as embedding. The embedding layer transforms the

discrete tokens into continuous vectors that capture semantic information. Positional encoding is integrated into this embedding stage, where it provides information about the position of each token in a sequence.

Third, at the core of the GPT model are multiple transformer layers, each containing two main components: attention mechanisms and feed-forward neural networks. These layers are stacked to form a deep network. The attention mechanism allows the model to weigh the importance of different tokens by computing attention scores for each token based on its relevance to other tokens in a sequence. This helps the model focus on the most relevant parts of the input when generating the output. The self-attention mechanism, in particular, enables the model to capture long-range dependencies and contextual relationships between tokens.

Fourth, the output of the attention mechanism is passed through a feed-forward MLP neural network. It consists of fully connected layers that apply non-linear transformations to the input, helping the model learn complex patterns and representations from the data.

Fifth, to stabilize training and improve performance, each transformer layer includes layer normalization and residual connections. Layer normalization normalizes the input to each layer, ensuring that the model's activations remain within a stable range. Residual connections add the input of a layer to its output, allowing the model to learn incremental improvements and preventing the vanishing gradient problem.

Sixth, after processing the input through the deep stack of transformer layers, the model generates the final output. In the case of text generation, the model produces a probability distribution over the token vocabulary, and the token with the highest probability is selected as the output. This process is repeated iteratively, by adding the inferred token to the previous input sequence and taking it as the input to the next iteration to generate coherent and contextually relevant text.

In summary, a GPT model's core processing stages involve tokenizing the input text, embedding the tokens into a multi-dimensional space, and passing them through a deep stack of transformer layers that consist of attention mechanisms and feed-forward neural networks. These stages enable the model to "understand" and generate human-like text based on the context provided.

Notice two key points about the above architecture, which are the basis of using GPTs for other than the textual data they were originally designed for. First, that any data, once tokenized, i.e., mapped to unique identifiers, can be processed with such models. This is also the basis of text to audio and text to image GenAI solutions [6]. Second, that where the above account talks about semantics and coherent, contextually relevant text, those features of what the model is designed to do can simply be expressed as being about relationships among tokens in the context of their sequences, regardless of what those tokens represent. An example from a color context here are also studies of spectral [7] and cone response [8] correlations.

### *LLMs for sensor data*

LLMs are increasingly being used with sensor data in a variety of ways around connecting text with them, around analyzing their sequential relationships and around extrapolating within their sequences.

First, they can enable interaction with and querying of sensor data by asking questions about it using natural language [9]. They can also be used to enable human interaction with the physical world on the basis of sensor data and Internet of Things (IoT) technologies [10]. LLMs can further be used associate sensor data with linguistic descriptions, which then allows for the retrieval of semantically relevant data from large sensor data sets, e.g., to complement a user's partial data set [11].

Second, a next level of LLM use is to analyze sensor data sequences, e.g., by using self-supervised learning for human activity recognition on the basis of accelerometer data from wearable sensors. LLMs are trained here to learn the sequences of temporal dependencies in human motion [12].

Third, LLMs have also been used to make forecasts in time series [13], which is the closest application to that explored in this paper, where a sequence of color sensor measurements can also be thought of as a time series.

## Treating color measurements as text

Training a GPT–based LLM on a large body of text allows for it to take an input like "There is a cat on a" and generate the word "mat" or "chair", etc. What would happen if a GPT was trained on color sensor data instead of text? Could it take a small sequence of, e.g., CIELAB values and generate subsequent ones?

### *Data set*

To explore the above question, a data set will be used to train and test a GPT model that consists of 450 CMYK ECI2002 color charts [14] printed using a variety of substrates and inks and measured with a spectrophotometer. Each color chart here consisted of 1504 measurements, resulting in a total of 676,800 CMYK-LAB pairs in 450 sequences. A small data set by LLM standards, but a relatively large color data set. Since virtually no hyperparameter optimization was performed here, the data set was split 80:20 into training and validation sets (360 charts and 90 charts respectively). Note that the measurements were converted and aggregated from the format used by the spectrophotometer to a simple text format where "START" and "END" indicated the limits of a chart and where each chart's data consisted of a sequence of alternating rows starting with "CMYK:" and "LAB:" each followed by 4 and 3 real values respectively.

### *GPT model*

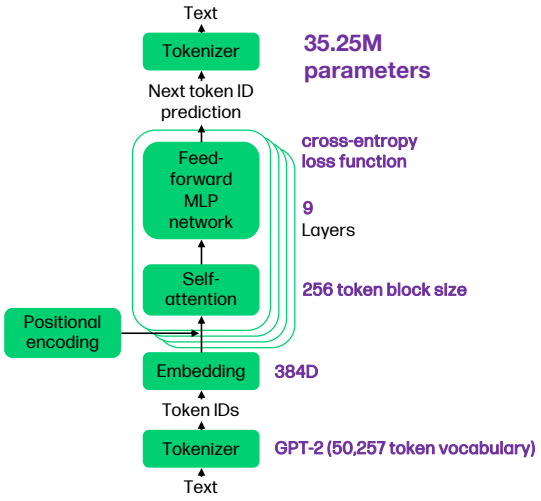


***Figure 2.*** *nanogpt parameters of model trained on color sensor measurements.*

Andrej Karpathy's *nanogpt* [15] was configured as shown in Fig. 2, resulting in a tiny 35M parameters compared, e.g., with the 1.75B parameters of GPT-3. Worth noting here is the use of the GPT-2 tokenizer [16], which means that the measurement text file was tokenized as if it was any other text, resulting in 20,337,644 tokens. The dimensionality of the embedding space, the token block size as well as the number or layers and other hyperparameters were chosen to allow for training on a single workstation. Finally, it is important to note that the loss function used here was cross–entropy [17] rather than a ΔE equation, since the model's outputs are not color values, as in the case of previous NN models applications to color, but token indices. In other words, the GPT here, as in the original text case, is used for classification and the appropriate loss function therefore is one that measures classification goodness.

### *First results*

To explore the potential for inferring measurements from a printing system in an unknown state, the first 10 CMYK–LAB pairs of the first validation chart were used as the input to the model, trained for 10K iterations on the 360-chart training set (i.e., a chart containing measurements from a system on which the model was not trained):

```
START
CMYK: 0.0 100.0 20.0 0.0
LAB: 45.38 71.93 5.74
CMYK: 0.0 85.0 20.0 0.0
LAB: 49.9 67.22 4.21
CMYK: 0.0 70.0 20.0 0.0
LAB: 56.48 58.19 3.08
CMYK: 0.0 55.0 20.0 0.0
LAB: 63.54 46.84 4.53
CMYK: 0.0 40.0 20.0 0.0
LAB: 71.65 32.74 7.43
CMYK: 0.0 30.0 20.0 0.0
LAB: 77.37 22.44 9.54
CMYK: 0.0 20.0 20.0 0.0
LAB: 82.25 14.13 12.2
CMYK: 0.0 10.0 20.0 0.0
LAB: 86.87 5.79 14.91
CMYK: 0.0 0.0 20.0 0.0
LAB: 93.03 -4.77 18.19
CMYK: 0.0 100.0 10.0 0.0
LAB: 45.71 73.18 -1.08
```

Tab. 1 shows the following 3 CMYK–LAB pairs, generated by the model. As can be seen, the "CMYK:" and "LAB:" prefixes are generated correctly, the CMYK values are also error free and the LABs inferred by the model are around the 4 ΔE00 mark. In other words, while such inferred values would not be usable for color calibration or profiling, the output is at least in the ballpark of the previously unseen ground truth. The cross–entropy loss for training was 0.6731, and for validation 0.6932, showing room for further optimizing the model to the training data without overfitting.

**Table 1: 3 CMYK–LAB pairs generated by 9–layer model trained for 10K iterations.**

| Ground truth | Generated | Difference / ΔE00 |
|---|---|---|
| CMYK: 0.0 85.0 10.0 0.0 | CMYK: 0.0 85.0 10.0 0.0 | [0,0,0,0] |
| LAB: 51.08 67.73 -4.26 | LAB: 55.76 62.75 -5.96 | 4.78 |
| CMYK: 0.0 70.0 10.0 0.0 | CMYK: 0.0 70.0 10.0 0.0 | [0,0,0,0] |
| LAB: 57.75 58.29 -5.6 | LAB: 61.45 52.08 -5.34 | 3.73 |
| CMYK: 0.0 55.0 10.0 0.0 | CMYK: 0.0 55.0 10.0 0.0 | [0,0,0,0] |
| LAB: 64.96 47.15 -5.22 | LAB: 68.73 40.21 -4.18 | 3.84 |

To explore the potential for improving model performance, the number of transformer layers was increased from 9 to 18, resulting in 51.18M parameters and generating the output shown in Tab. 2. While the training and validation loss were 0.6820 and 0.6935 respectively, the generated LAB data was improved.

**Table 2: 3 CMYK–LAB pairs generated by 18–layer model trained for 10K iterations.**

| Ground truth | Generated | Difference / ΔE00 |
|---|---|---|
| CMYK: 0.0 85.0 10.0 0.0 | CMYK: 0.0 85.0 10.0 0.0 | [0,0,0,0] |
| LAB: 51.08 67.73 -4.26 | LAB: 51.47 59.4 -2.27 | 2.32 |
| CMYK: 0.0 70.0 10.0 0.0 | CMYK: 0.0 70.0 10.0 0.0 | [0,0,0,0] |
| LAB: 57.75 58.29 -5.6 | LAB: 58.6 48.1 -2.43 | 3.35 |
| CMYK: 0.0 55.0 10.0 0.0 | CMYK: 0.0 55.0 10.0 0.0 | [0,0,0,0] |
| LAB: 64.96 47.15 -5.22 | LAB: 65.87 35.1 -2.96 | 4.38 |

Worth noting is that different approaches to structuring the data resulted in a complete breakdown in the model. E.g., using only the LAB sequences (i.e., without pairing them with CMYKs) resulted in wildly different predictions, as did a grouping of the first 10 pairs with each of the remaining pairs. While this, second, approach of data augmentation increased the training set to 279M tokens (from the original 20M) it too resulted in inferences wholly unlike the test data.

Reflecting on this initial exploration suggested a focus on the tokenizer component of the model, since it is the element that bridges that specific modality of the data and the token index abstraction that only has a nominal relationship to the data. To see how the GPT-2 tokenizer used here represents color sensor data, tokenizing breaks it down into "["Lab", ":", "Ġ", "83", ".", "4675", "Ġ", "-", "37", ".", "3842", "Ġ", "7", ".", "911"]," where Ġ represents any whitespace (space, tab, ...), and then mapping these to the following token index sequence: [17822, 25, 9698, 13, 3510, 2425, 532, 2718, 13, 2548, 3682, 767, 13, 35549]. A key challenge here is that the LAB values whose sequence an LLM needs to learn (83.4675 -37.3842 7.911) are broken down in ways that compromise the integrity of the numerical data - e.g., of the three floating point values in this example.

## A custom tokenizer

The importance of the tokenizer, when using LLMs with sensor data, has been recognized already in previous work [13], [18], with a focus on analyzing the shortcomings of existing text tokenizers or identifying more suitable ones from among them for numerical data. A fascinating example of developing custom tokenization here is also a recent paper where a sequential encoding sequence is devised for directed acyclic graphs, thereby enabling their processing with LLMs [19].

A closer look at the data used here suggested the idea of a new, sensor data specific tokenizer, to take advantage of the fact that the basic building blocks of sensor data sequences are numerical values, potentially interspersed with so–called "special" tokens. Special tokens were introduced already in the original transformer architecture paper [20], where [CLS] (classification) and [SEP] (separator) tokens were used alongside those derived from the training data.

In the print color measurement example explored here, it is sufficient to consider four special tokens, one each for "START", "END", "CMYK:" and "LAB:" The remaining inputs are all real numbers, which can be represented discretely by normalizing and quantizing them appropriately. For the CMYK and LAB values considered here, CMYKL are all on a [0,100] range and can be mapped to a token index in the same way, while AB are on [-128,127] ranges and need to be normalized accordingly.

Tab. 3 therefore shows the new tokenizer tuned to CMYK and LAB data, where analogous schemes could be devised for other sensor data.

**Table 3: A CMYK–LAB tokenizer.**

| Index | Token |
|---|---|
| 0 | START |
| 1 | END |
| 2 | CMYK: |
| 3 | LAB: |
| x | x = round(v * ($2^{\text{bit-depth}}$ - 1) + offset)<br>v=(in - min) / (max - min) |

where offset is the number of special tokens (4 here), bit-depth is the number of bits to use to quantize the floating-point values (e.g., 8 results in a token vocabulary size of 256+offset), round() rounds a real number to the nearest integer, v is a real number normalized to a [0,1] range based on the [min, max] rage of in. Having different such ranges for CMYKL and AB can be handled given the fixed sequence of the "CMYK:" token always being followed by 4 reals and then by the "LAB:" token and 3 reals. Which real number of CMYKL or AB can be inferred sequentially. In other words, this tokenization scheme is tied to the data representation of CMYK-LAB pairs used here. As can be seen, the tokenization set out above is anything but a general-purpose text tokenizer. Instead, it is tied both to the specific sensor data dealt with here and its sequential structure. Note that an alternative here would be to base the tokenizer on an encoding like IEEE 754 [21] that also represents floating point values as integers but does so for an exponent range rather than the specific LAB or CMYK ranges to which the above is adjusted.

Applying this tokenizer to the "Lab: 83.4675 -37.3842 7.911" example, results in 4 tokens with the following indices: [3, 27354, 11519, 17340].

### *Data and model configuration*

The same training and validation data as used above was tokenized using the custom sensor data tokenizer using a bit-depth of 15 bits, which resulted in a token vocabulary of 32,772 (i.e., 4 special tokens + $2^{15}$=32,768 tokens to encode real values) and a data set containing 6M tokens (instead of the 20M tokens of GPT-2). This bit depth gives a high degree of quantization resolution and stays well below the size of the GPT-2 tokenizer.

In terms of the GPT model itself no change was made whatsoever. The sequence from input token index sequence to inferred output token indices remained as before.

### *Results*

Re-training the model with the new, CMYK–LAB sensor data tokenizer for 10K iterations resulted in a training loss of 2.0114, and a validation loss of 2.1487 and in the inferred sequence shown in Tab. 4. Note that cross-entropy values between the two tokenizers used here cannot be compared directly since the data sets are of different sizes (in terms of the tokens over which modelling takes place), have different distributions and different self-attention characteristics [22]. The ΔE00s, instead can be compared since they relate to color differences between ground truth validation CIELAB and inferred CIELAB values. As can be seen when comparing Tabs. 2 and 4, the model using a GPT–2 tokenizer resulted in errors between 2.3 and 4.4 ΔE00, while generating CIELABs using the sensor data tokenizer yielded significantly lower color differences in the 0.5 to 0.9 ΔE00 range.

**Table 4: 3 CMYK–LAB pairs generated by 18–layer model using sensor data tokenizer trained for 10K iterations.**

| Ground truth | Generated | Difference / ΔE00 |
|---|---|---|
| `CMYK: 0.0 85.0 10.0 0.0` | `CMYK: 0.0 85.0 10.0 0.0` | [0,0,0,0] |
| `LAB: 51.08 67.73 -4.26` | `LAB: 50.41 66.38 -4.96` | 0.81 |
| `CMYK: 0.0 70.0 10.0 0.0` | `CMYK: 0.0 70.0 10.0 0.0` | [0,0,0,0] |
| `LAB: 57.75 58.29 -5.6` | `LAB: 56.86 57.46 -6.25` | 0.90 |
| `CMYK: 0.0 55.0 10.0 0.0` | `CMYK: 0.0 55.0 10.0 0.0` | [0,0,0,0] |
| `LAB: 64.96 47.15 -5.22` | `LAB: 64.40 46.91 -5.57` | 0.50 |

To test the model's performance over the full set of 90 validation charts, the first 10 CMYK–LAB pairs from each were taken as a prompt and ten subsequent CMYK–LAB pairs were generated. This resulted in 90*10=900 ΔE00 color differences between measured and inferred color sensor measurements. Fig. 3 shows violin plots of the ΔEs for the ten sequential CIELAB inferences across the 90 charts, with medians between 0.8 and 1.6 ΔE00. The median overall, across the 90 sets of 10 ΔE00s was 1.17, with a 95$^{th}$ percentile of 3.59 and a maximum of 4.38. This would be the performance of the simple model tested here for inferring as many sensor readings as it is prompted with – i.e., doubling the measured data. If instead the model was used to add only 30%, the inferred CIELABs would have a median of 1.01, with a 95$^{th}$ percentile of 2.05 and a maximum of 2.98.

An interesting feature of the violin plots in Fig. 3 is also that in many cases the error distribution shows clear clustering (e.g.,

note the large separation between two clusters for inference #5). This may be related to the heterogeneity of the present data set, which involves multiple printing systems, ink sets and substrates. Having a more homogeneous training set and deploying the resulting model only to systems with the same high-level configuration would be worth exploring in the future.

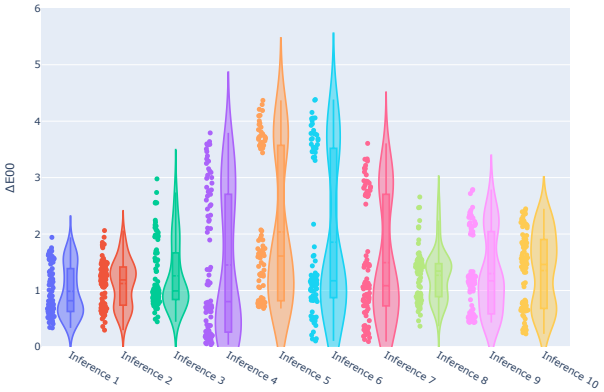


***Figure 3.*** *Violin plots of ten sequential CIELAB inference ΔE00s.*

A further choice point is about how much data is generated relative to measured, which will depend on the color reproduction use case (e.g., spot color matching, contract proofing and temporary point of sale posters all have different requirements). The level of accuracy can be balanced against savings in time and materials. Fig. 4 shows how the ΔE statistics of the first *n* inferences change as *n* increases from 1 to 10. It can be seen that the median slowly increases, while the tail of the distribution grows rapidly between the 2nd and 5th inferences and then levels off. With such dependence on the number of inferred versus measured input data characterized, a choice of appropriate balance can be made.

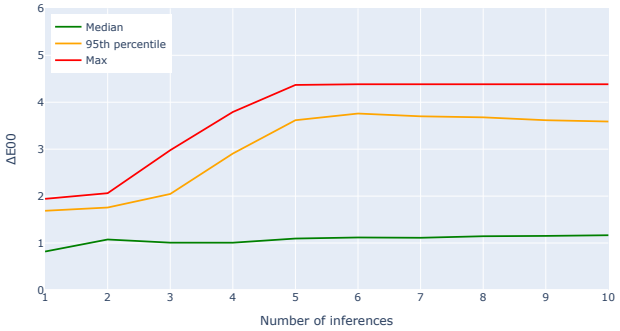


***Figure 4.*** *ΔE00s statistics for the first n inferences from 1 to 10.*

## Conclusions and next steps

In summary, this paper introduced the use of the generative pre-trained transformer architecture to the sequential inference of color sensor measurements. Coupled with the development of a custom, sensor-data-specific tokenizer, a GPT model trained on 360 color charts successfully made predictions about color measurements from systems under unknown conditions. Median ΔE00 values of around one were found for a toy example of inferring 3 measurements, given a sequence of 10, which are comparable to the print and measure repeatability of some systems and sufficient for delivering color control for some applications.

The work presented here is very much a first exploration, albeit a very promising one, and there is a wealth of aspects of this approach to explore in the future. The optimization of prompt set (instead of using the first 10 CMYK–LAB pairs of a chart), use of more focused data sets (e.g., from only a single system, ink set and substrate combination), larger data sets (e.g., collected over a population of imaging systems over extended periods of time) and larger models (since the training-validation loss difference here was still modest) is likely to lead to further improvements. Further to be explored is also the statistical nature of GPT inference, whereby the model results in token index probabilities from among which a single choice can be made in a variety of ways. Linking these statistics with the inherently statistical nature of color [23] also offers enticing prospects.

## Acknowledgements

The authors would like to thank their colleagues at HP Inc. for their support, especially Marc Isal and Alan Lobban.

## References

[1] D. Adkins, V. S. Cherkassky, and E. S. Olson, “Color Mapping Using Neural Networks,” *Color and Imaging Conference*, vol. 1, no. 1, pp. 45–48, Jan. 1993, doi: 10.2352/CIC.1993.1.1.ART00010.

[2] X. Wei, N. Zou, L. Zeng, and Z. Pei, “PolyJet 3D printing: Predicting color by multilayer perceptron neural network,” *Annals of 3D Printed Medicine*, vol. 5, p. 100049, Mar. 2022, doi: 10.1016/J.STLM.2022.100049.

[3] J. Morovič and P. Morovič, “From Gutenberg to Llamas: Print Optimization Through First Principles and AI,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* , vol. 15193 LNCS, pp. 166–180, 2024, doi: 10.1007/978-3-031-72845-7_12.

[4] L. Chen, M. R. Luo, and M. Wei, “Large Size of Color Constancy: Enhancing Pure Color Image Illuminant Estimation with Kolmogorov-Arnold Networks,” *Color and Imaging Conference*, vol. 32, no. 1, pp. 95–100, Oct. 2024, doi: 10.2352/CIC.2024.32.1.19.

[5] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving Language Understanding by Generative Pre-Training,” 2018, Accessed: May 15, 2024. [Online]. Available: https://gluebenchmark.com/leaderboard

[6] A. Ramesh *et al.*, “Zero-Shot Text-to-Image Generation,” 2021, Accessed: May 15, 2024. [Online]. Available: https://github.com/openai/DALL-E

[7] P. Morovič, J. Morovič, and J. M. García-Reyero, “Spectra from correlation,” in *Final Program and Proceedings - IS and T/SID Color Imaging Conference*, 2013.

[8] D. H. Brainard, “Color and the Cone Mosaic,” *Annu. Rev. Vis. Sci*, vol. 1, pp. 519–565, 2015, doi: 10.1146/annurev-vision-082114-035341.

[9] J. Liu *et al.*, “ChainStream: An LLM-based Framework for Unified Synthetic Sensing,” pp. 2024–2036, Dec. 2024, Accessed: Feb. 13, 2025. [Online]. Available: https://arxiv.org/abs/2412.15240v1

[10] H. Xu, L. Han, Q. Yang, M. Li, and M. Srivastava, “Penetrative AI: Making LLMs Comprehend the Physical World,” *HOTMOBILE 2024 - Proceedings of*

*the 2024 25th International Workshop on Mobile Computing Systems and Applications*, pp. 1–7, Feb. 2024, doi: 10.1145/3638550.3641130.

[11] J. Leveraging *et al.*, "Leveraging Large Language Models for Sensor Data Retrieval," *Applied Sciences 2024, Vol. 14, Page 2506*, vol. 14, no. 6, p. 2506, Mar. 2024, doi: 10.3390/APP14062506.

[12] H. Yuan *et al.*, "Self-supervised learning for human activity recognition using 700,000 person-days of wearable data," *npj Digital Medicine 2024 7:1*, vol. 7, no. 1, pp. 1–10, Apr. 2024, doi: 10.1038/s41746-024-01062-3.

[13] N. Gruver, M. Finzi, S. Qiu, and A. G. Wilson, "Large language models are zero-shot time series forecasters," in *Proceedings of the 37th International Conference on Neural Information Processing Systems*, in NIPS '23. Red Hook, NY, USA: Curran Associates Inc., 2023.

[14] "ISO 12642-2:2006 - Graphic technology — Input data for characterization of 4-colour process printing — Part 2: Expanded data set." Accessed: Feb. 05, 2025. [Online]. Available: https://www.iso.org/standard/39371.html

[15] "GitHub - karpathy/nanoGPT: The simplest, fastest repository for training/finetuning medium-sized GPTs." Accessed: Feb. 14, 2025. [Online]. Available: https://github.com/karpathy/nanoGPT

[16] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners", Accessed: Feb. 17, 2025. [Online]. Available: https://github.com/codelucas/newspaper

[17] D. E. Rumelhart and J. L. McClelland, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, 1987, pp. 318–362.

[18] D. Spathis and F. Kawsar, "The first step is the hardest: pitfalls of representing and tokenizing temporal data for large language models," *Journal of the American Medical Informatics Association*, vol. 31, no. 9, pp. 2151–2158, Sep. 2024, doi: 10.1093/JAMIA/OCAE090.

[19] M. Sun, O. Foo, G. Liu, W. Matusik, and J. Chen, "Directed Graph Grammars for Sequence-based Learning," 2025, Accessed: Jun. 03, 2025. [Online]. Available: https://arxiv.org/pdf/2505.22949

[20] A. Vaswani *et al.*, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[21] "IEEE Standard for Floating-Point Arithmetic," Art. no. 754–2008, Jun. 2019, doi: 10.1109/IEEESTD.2019.8766229.

[22] D. Brandfonbrener, N. Anand, N. Vyas, E. Malach, and S. Kakade, "Loss-to-Loss Prediction: Scaling Laws for All Datasets," Nov. 2024, Accessed: Feb. 18, 2025. [Online]. Available: https://arxiv.org/abs/2411.12925v1

[23] J. Morovic and P. Morovic, "Mean Color, Probably," *Color and Imaging Conference*, vol. 32, no. 1, pp. 114–121, Oct. 2024, doi: 10.2352/CIC.2024.32.1.22.

## Author Biographies

*Ján Morovič received his Ph.D. in color science from the University of Derby (UK) in 1998, where he then worked as a lecturer. Since 2003 he has been at Hewlett-Packard in Barcelona as a senior color scientist and later master technologist. He has also served as the director of CIE Division 8 on Image Technology and Wiley and Sons have published his 'Color Gamut Mapping' book. He is the author of over 120 papers and has filed 180+ US patents (145 granted).*

*Peter Morovič received his Ph.D. in computer science from the University of East Anglia (UK) in 2002 and holds a B.Sc. in theoretical computer science from Comenius University (Slovakia). He has been a senior color and imaging scientist at HP Inc. since 2007, has published 65+ scientific articles and has 180+ US patents filed (144 granted) to date. His interests include color science, image processing, color vision, computational photography, computational geometry.*