Power Electronics Turing Test: A Path Toward Strong AI in Power Electronics

Minjie Chen ¹ and Dak Cheung Cheng¹

¹Princeton University

June 26, 2024
Abstract—This paper presents a hypothetical Turing test in power electronics, leveraging structured computer vision as a step towards domain-specific artificial general intelligence (AGI). To illustrate the key principles of such a power electronics Turing test, we developed PowerVision, a computer vision framework designed to teach machines to understand schematic drawings. PowerVision comprises four key components: 1) ComponentNet: an image database for component recognition; 2) CircuitNet: an image database for schematic recognition; 3) NetlistMaker: a schematic recognition tool that converts human-readable schematics into netlists for SPICE simulations; and 4) NetlistClassifier: a circuit classification tool that can categorize different power electronics circuits based on machine-generated netlists. The PowerVision platform can facilitate the learning of power electronics fundamental principles by large-scale AGI models through human-accessible information including texts, schematics, computer simulations, and experimental results, ultimately enabling machines to comprehend power electronics.

Index Terms—artificial general intelligence, Turing test, machine learning, computer vision, netlists, SPICE simulation

I. INTRODUCTION

Artificial general intelligence (AGI), a type of AI that can perform as well as or better than human and interact with human across a wide range of cognitive tasks, holds great potential in engineering design [1]–[4]. A critical aspect of human learning in power electronics is the coherent integration of vast amounts of information in various formats, such as text descriptions, schematics, computer simulations, and images in textbooks, scientific publications, and engineering handbooks. For domain-specific AGI models to assist human designers or to perform automated design tasks, they must be capable of learning from human-accessible information.

Training AGI models (e.g., ChatGPT) to understand domain knowledge is challenging because of the lack of good training data and domain-specific knowledge abstraction. There is no large-scale high-quality database (such as those in [5], [6]) for AGI models to learn about power electronics. Modern large language models (e.g., ChatGPT-3.5, Google Gemini, Github Copilot) understand texts and netlists, but there is no database for them to understand the connections among texts, netlists, and drawings. This paper tries to address the above challenges by developing PowerVision, an end-to-end framework for exploring AGI in power electronics, comprising:

1) **ComponentNet**: a hand-collected image database with 2,561 power electronics component drawings;

2) **CircuitNet**, a hand-collected image database with 200 hand-labeled power electronics circuit schematics;

3) **NetlistMaker**, an automated schematic recognition tool which converts human-readable schematics;

4) **NetlistClassifier**, an automated circuit classification tool which can classify different power electronics circuits based on the machine-readable netlists;

5) **NetlistSimulator**, an automated circuit simulator which allows the AGI model to run SPICE simulations based on the netlist created.

All data and tools are open-sourced in GitHub.

II. POWER ELECTRONICS TURING TEST

The Turing test, initially proposed as the imitation game by Alan Turing in 1950 [7], assesses a machine’s capability to demonstrate intelligent behavior comparable to, or
Fig. 2. Power electronics topologies and their operational principles: https://www.ti.com/lit/ml/sluw001g/sluw001g.pdf. The majority of practical power electronics converters are derived from these topologies. Each of these topologies can serve as a basis for conducting a domain-specific Turing test in power electronics. Comprehending the operational principles of these topologies constitutes the initial step towards achieving AGI in power electronics.

indistinguishable from, that of a human. A more specific interpretation of the Turing test focuses on a computer’s ability to successfully mimic human behavior within a designated domain of human intelligence, such as power electronics. Illustrated in Fig. 1, the hypothetical power electronics Turing test entails a human evaluator posing questions to both a human expert and a computer model, seeking explanations of schematic drawings. The evaluator remains unaware of which conversational partner is the machine, and all participants are isolated from each other. The conversation begins with a schematic drawing as input and culminates in a human-language description of the schematic or a response to a specific inquiry. If the evaluator cannot reliably distinguish between the machine and the human, the machine is deemed to have successfully passed the test.

Figure 2 showcases twelve main-stream dc-dc power electronics topologies commonly employed in practical applications, comprising: Buck, Boost, Buck-Boost (Inverting), SEPIC, Flyback, Forward, 2 Switch Forward, Active Clamp Forward, Half Bridge Push Pull, and Full Bridge Phase Shift ZVT. A significant portion of power electronic systems are designed based on these switching cells. An average power electronics engineer possesses the ability to swiftly classify these topologies from schematic drawings and articulate their operational principles using concise natural language sentences. In a power electronics Turing test, the evaluator might present a drawing of a power converter to both the human expert and the computer model, asking, “Is this converter a pulse-width-modulated (PWM) converter? How does it work? Can you simulate it, and describe how it works?” If the evaluator cannot discern between the responses from the machine and the human expert, the machine would be considered to have successfully passed the specific test.

Figure 3. State-of-the-art AGI (e.g., ChatGPT-3.5 [8]) can generate commonly used power electronics netlists based on user prompts. However, it lacks the ability to interpret power electronics schematics or offer design guidance. A framework for machines to comprehend human-accessible information is pivotal for developing a domain-specific AGI capable of assisting power electronics design.

II. POWERVISION FRAMEWORK

As depicted in Fig. 3, contemporary AGI models like ChatGPT possess the ability to read and generate text as well as simple drawings [8]. However, they currently struggle with comprehending power electronics components or schematic drawings, primarily due to the absence of high-quality training data. Early attempts of power electronics topology synthesis lacked the support of modern artificial intelligence [9]. The PowerVision framework aims to bridge this gap by educating computers on understanding power electronics through the process of mapping circuit schematic drawings to netlists.

Figure 4 outlines the fundamental information flow of PowerVision. To effectively train large language models for
human interaction, it’s imperative to convert human knowledge into machine-readable data. Human-accessible information encompasses schematic drawings and textual descriptions, where the operational principles are articulated in natural language. Although qualitative and concise, human-accessible information can be prone to inaccuracies. Machine-accessible information in power electronics comprises netlists, component bill-of-materials, printed circuit boards (PCBs), and 3D assembly drawings. Additionally, computers can engage circuit simulation tools to interpret netlists or utilize finite-element modeling (FEM) software like ANSYS to analyze physical components, PCBs, interconnects, and electromagnetic radiation. Machine-readable information is characterized by its quantitative, structured, complex, and precise nature.

At the core of PowerVision lies the circuit netlist, which empowers human and AGI models to utilize simulation tools for cultivating new understandings. The PowerVision platform encompasses the following five pivotal function blocks:

- **ComponentNet**: a large scale image database comprising thousands of images of different electrical components. Currently, the database contains 2,544 hand-collected and labeled component images in 11 categories, including “ac-src”, “battery”, “cap”, “curr-src”, “diode”, “inductor”, “resistor”, “swi-ideal”, “swi-real”, “volt-src”, and “xformer”. Each image is a $64 \times 64$ .PNG file with about 6 KB of size.

- **CircuitNet**: a large scale image database comprising hundreds of power electronics schematics for component detection and netlist generation. Currently the database contains 200 hand-collected and labeled power electronics schematics for different purposes.

- **NetlistMaker**: a computer program which reads the schematic drawing and generates a netlist and an incident matrix based on the detected components and the wires that connect the components together.

- **NetlistClassifier**: a computer program which classifies different power electronics schematic drawings and create labels such as “PWM”, “isolated”, etc.

- **NetlistSimulator**: The created netlist is SPICE compatible. The NetlistSimulator can simulate the netlist in SPICE (e.g., PySPICE in Python) to autonomously create new knowledge and label new data.

### A. ComponentNet and CircuitNet

Figure 7. Key principles of the NetlistMaker tool. Initially, the 200 schematics stored in the schematic database undergo segmentation for component detection, utilizing OpenCV. Following segmentation, a CNN is trained using the component database to classify the detected components within the segmented schematics. Once classified, the components, along with network information, are transformed into an incidence matrix and represented as a netlist.

B. NetlistMaker

Figure 7 illustrates the underlying principles of NetlistMaker, a fully automated image recognition tool designed to convert schematic drawings into netlists. Apart from employing a suite of standard image processing techniques like sharpening, color inverting, object detection, scaling, and normalization, NetlistMaker’s core comprises a convolutional neural network (CNN) tailored for component classification. This CNN architecture, akin to those developed in [13]–[15], is specifically crafted and trained for power electronics. It features: 1) an image pre-processing layer for data augmentation, which randomly rotates, flips, and rescales component images; 2) three sets of 2D convolution and pooling layers; 3) a flatten layer followed by three dense layers for layer aggregation and classification. The training images are formatted as binary (64×64) pixels, and the CNN kernel size is (3×3). The model encompasses 276,491 parameters and occupies 1.05 MB of space. Figure 8 presents the confusion matrix of the classification results across 11 categories, with an overall accuracy of approximately 90%. Notably, the performance varies across different component models. Enhancements in component detection accuracy can be achieved through improved data, refined models, and optimized training processes.

After detecting and labeling each component, the NetlistMaker then identifies how these components are connected with each other and create an incidence matrix. Each row of the incidence matrix represents a component in the schematic. Each column of the incidence matrix represents a wire node in the schematic. If one component is connected to a node (overlaps detected in the schematic), the element located at the corresponding row and column is sequentially marked as integer numbers “1”, “2”, “3”, etc. Different integer numbers indicate different ports of the same component. All other

<table>
<thead>
<tr>
<th>Layer (type)</th>
<th>Output Shape</th>
<th>Param #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Flip</td>
<td>(64, 64, 1)</td>
<td>0</td>
</tr>
<tr>
<td>Random Rotation</td>
<td>(64, 64, 1)</td>
<td>0</td>
</tr>
<tr>
<td>Rescaling</td>
<td>(64, 64, 1)</td>
<td>0</td>
</tr>
<tr>
<td>Conv 2D</td>
<td>(61, 61, 64)</td>
<td>1088</td>
</tr>
<tr>
<td>MaxPooling 2D</td>
<td>(30, 30, 64)</td>
<td>0</td>
</tr>
<tr>
<td>Conv 2D</td>
<td>(26, 26, 64)</td>
<td>102464</td>
</tr>
<tr>
<td>MaxPooling 2D</td>
<td>(13, 13, 64)</td>
<td>0</td>
</tr>
<tr>
<td>Conv 2D</td>
<td>(9, 9, 64)</td>
<td>102464</td>
</tr>
<tr>
<td>MaxPooling 2D</td>
<td>(4, 4, 64)</td>
<td>0</td>
</tr>
<tr>
<td>Flatten</td>
<td>1024</td>
<td>0</td>
</tr>
<tr>
<td>Dense</td>
<td>64</td>
<td>65600</td>
</tr>
<tr>
<td>Dense</td>
<td>64</td>
<td>4160</td>
</tr>
<tr>
<td>Dense</td>
<td>11</td>
<td>715</td>
</tr>
</tbody>
</table>
Fig. 8. The confusion matrix for classifying the 11 categories of components revealed that the model achieved satisfactory performance for most categories. It exhibited poor performance specifically for the “battery” and the “swi-ideal” categories, indicating a need for further refinement and improvement.

Fig. 9. The NetlistMaker processes an example SEPIC converter schematic drawing with: (1) component detection; (2) netlist recognition; and (3) component value assignment. The component values are randomly selected. Here the six columns – \{N_0, N_1, N_2, N_3, N_4, N_5\} – represent the six nodes in the schematic, and the eight rows – \{R, C, V, L, M, D, C\} – represent the eight components in the schematic. The NetlistMaker made a perfect schematic recognition in this example with all components and wire connections correctly identified. The SPICE netlist is:

\[
\begin{array}{ccccccc}
N_0 & N_1 & N_2 & N_3 & N_4 & N_5 \\
R & 2 & 1 & 0 & 0 & 0 & 0 \\
C & 2 & 1 & 0 & 0 & 0 & 0 \\
V & 2 & 0 & 0 & 0 & 1 & 0 \\
L & 2 & 0 & 1 & 0 & 0 & 0 \\
M & 3 & 0 & 0 & 1 & 0 & 2 \\
L & 0 & 0 & 0 & 2 & 1 & 0 \\
D & 0 & 2 & 1 & 0 & 0 & 0 \\
C & 0 & 0 & 2 & 1 & 0 & 0 \\
\end{array}
\]

Figures 10 shows the component detection and recognition results of the 12 example topologies listed in Fig. 2. With a SPICE netlist, an AGI model can automatically call a SPICE simulation platform to enhance the learning process. In PowerVision, the generated SPICE netlist is then fed into PyLTSpice\(^2\) - a tool-chain of python utilities design to interact with LTSpice. It is mostly based on the SPICELib\(^3\) package, automatically selecting LTSpice to perform all simulations. Figure 11 shows an example simulation results from PyLTSpice for this SEPIC converter during a startup transient.

C. NetlistClassifier

We evaluate PowerVision’s capability in classifying different circuit topologies. Human experts classify different circuit schematics by identifying the critical components and patterns in how they are connected to each other. An incidence matrix uniquely represents a circuit netlist and can be used as a key identifier. In NetlistClassifier, a netlist incidence matrix is first converted into a 2D PNG image whose element values are coded to represent different components, e.g., a “capacitor” row of the incidence matrix will be represented by a deeper color in a gray-scale image, and an “inductor” row of the

\(^2\)PyLTSpice: https://pypi.org/project/PyLTSpice/

\(^3\)SPICELib: https://github.com/nunobrum/spicelib
Fig. 10. Component detection and recognition results of the 12 example topologies listed in Fig. 2, with key circuit patterns highlighted.

Fig. 11. PyLTspice simulation as a part of the PowerVision platform. The voltage waveforms of multiple switching nodes are shown.

incidence matrix is represented by a lighter color in a gray-scale image. In this way, the incidence matrix classification problem is converted into an image classification problem which can be solved by various computer vision tools.

Figure 12 explains the correlation between the patterns in the circuit schematic and the patterns in a color-coded incidence matrix. Human eyes recognize the patterns in the circuit schematic and infer the basic circuit behavior (e.g., a half-bridge switching cell usually comprises two switches and one inductor connected to one joint node). State-of-the-art machine learning algorithms (such as a Convolutional Neural Networks [16], or Transformers [17]) have similar “attention” mechanisms to recognize unique patterns in the incidence matrix, opening the opportunity to implement AGI models which can correlate circuit schematics with circuit functions.

Figure 13 illustrates a few example incidence matrices represented as 2-D gray-scale images, which can be processed as tensors for image recognition. Each row represents a component. Each column represents a node in the schematic. The non-black pixels represent non-zero elements in the incidence matrix. The component types are coded in the pixel darkness.

Table II lists the structure of the CNN of the NetlistClassifier. The 200 schematics in the CircuitNet were all converted into 2D images, then hand-labeled as “PWM” or “non-PWM”.
“isolated” or “non-isolated” converters. In each experiment, 160 schematics were used for training the NestlistClassifier model, and 40 schematics were used for testing. The confusion matrix for classifying the “PWM” and “isolated” converters are listed in Fig. 14. Without extensive training or model optimization, the accuracy for “PWM” vs. “non-PWM” classification was 70%. The accuracy for the “isolated” vs. “non-isolated” classification was 62.5%. Clearly, the PowerVision model we developed in this paper will not pass the power electronics Turing test.

### IV. Challenges and Opportunities

PowerVision’s capabilities in component detection and topology classification are still far behind that of an average human learner. The inaccuracies in the PowerVision framework may stem from: (1) errors in component detection and netlist creation; (2) errors during the training and testing phases of classification; (3) errors in human labeling of components and schematics. These errors tend to propagate and magnify, limiting the overall end-to-end prediction accuracy.

An illustration of component detection and classification errors is provided in Fig. 15. Figure 16 shows a wide range of different symbols used for representing power transformers. PowerVision cannot differentiate these symbols, and cannot identify correct netlist if these symbols present in the schematics. Although the NetlistMaker generally performs well with over 90% accuracy, errors still occur, potentially influencing subsequent simulations or classifications. Other remaining challenges preventing PowerVision from efficiently processing and learning from human-accessible data include:

- **Understanding semiconductor devices and magnetic components.** PowerVision cannot differentiate semiconductor devices and magnetic components drawn in different ways, and do not understand the principles used for designing them (e.g., turns ratios, lumped models, etc.).

- **Distinguishing circuits, text, markers, and labels.** Humans interpret schematics by integrating graphical netlist data with text positioned near graphic components. Over time, humans naturally learn to discern various formats of information (e.g., boxes, dashed lines, arrows, color highlights). Presently, PowerVision lacks the ability to independently interpret different types of drawing data.

- **Identifying and categorizing overlapping components.** Human excels at distinguishing and isolating close or even overlapping components. However, PowerVision currently lacks this capability. Errors may occur in de-
Fig. 16. A wide range of different power transformer models (Available: https://www.electricaltechnology.org/2019/09/transformer-symbols.html).

- **Interpreting wire connections.** Humans possess the ability to recognize and understand concepts like “cross-wires” and “junctions” within schematic drawings. However, PowerVision currently lacks this capability.

- **Automated error correction.** Human readers often possess the capacity to comprehend a schematic by focusing on high-level information and overlooking minor errors. However, PowerVision currently lacks this ability.

However, specialized AGI tools like PowerVision will continue to gain from advancements in the general-purpose AGI’s capacity for image comprehension. Addressing these limitations can be achieved through: (1) expanding and enhancing the database’s size and quality; (2) forthcoming developments in computer vision and image processing technologies; (3) refining model architecture and training methodologies.

Training a comprehensive AGI model to comprehend power electronics falls beyond the scope of this paper. However, leveraging the PowerVision tool enables the swift conversion of numerous power electronics schematics into netlists, thereby establishing a sizable multi-domain database primed for training. We’ve devised a software framework to automate this process entirely. Utilizing a PDF segmentation tool (PDFSegmenter*), we automatically extract images and text from a vast array of PDF documents. Schematic images, their corresponding netlists (including incident matrices), and descriptive texts are then paired and stored within the database. Ultimately, this entire procedure can be automated and seamlessly integrated with a general-purpose AGI model.

**V. CONCLUSION**

This paper introduces PowerVision as a research framework for investigating the power electronics Turing test—a domain-specific method for assessing the proficiency of AGI models’ capability of understanding power electronics. PowerVision facilitates automatic conversion of schematic diagrams from power electronics literature into netlists, enabling basic tasks such as SPICE simulations and topology classification. These tools have the potential to significantly expedite the training of general-purpose AGIs with domain-specific knowledge, and enabling them to comprehend and apply the principles of power electronics.

**REFERENCES**


