Digitizing images of electrical-circuit schematics

Cite as: APL Mach. Learn. 2, 016109 (2024); doi: 10.1063/5.0177755
Submitted: 24 September 2023 • Accepted: 1 January 2024 •
Published Online: 26 January 2024

Charles R. Kelly[1] and Jacqueline M. Cole[1,2,a)]

AFFILIATIONS

[1] Department of Physics, Cavendish Laboratory, University of Cambridge, J. J. Thomson Avenue, Cambridge CB3 0HE, United Kingdom
[2] ISIS Neutron and Muon Source, Rutherford Appleton Laboratory, Harwell Science and Innovation Campus, Didcot OX11 0QX, United Kingdom

[a)] **Author to whom correspondence should be addressed:** jmc61@cam.ac.uk

ABSTRACT

Electrical-circuit schematics are a foundational tool in electrical engineering. A method that can automatically digitalize them is desirable since a knowledge base of such schematics could preserve their functional information as well as provide a database that one can mine to predict more operationally efficient electrical circuits using data analytics and machine learning. We present a workflow that contains a novel pattern-recognition methodology and a custom-trained Optical Character Recognition (OCR) model that can digitalize images of electrical-circuit schematics with minimal configuration. The pattern-recognition and OCR stages of the workflow yield 86.4% and 99.6% success rates, respectively. We also present an extendable option toward predictive circuit-design efficiencies, subject to a large database of images being available. Thereby, data gathered from our pattern-recognition workflow are used to draw network graphs, which are in turn employed to form matrix equations that contain the voltages and currents for all nodes in the circuit in terms of component values. These equations could be applied to a database of electrical-circuit schematics to predict new circuit designs or circuit modifications that offer greater operational efficiency. Alternatively, these network graphs could be converted into simulation programs with integrated circuit emphasis netlists to afford more accurate and computationally automated simulations.

I. INTRODUCTION

Schematic diagrams are an important tool for communication in all engineering disciplines[1] and, therefore, they occur frequently in published literature. It is, therefore, important that we can digitally interpret these schematics so that they can be quickly understood, analyzed, and compared. Approaches to digital interpretation of images that are embedded within documents vary, but they typically involve image pre-processing techniques, pattern recognition, and rule-based algorithms to segment various types of input such as circuit schematics,[1–3] microscopy images,[4–6] and chemical schematics.[7,8]

Various solutions have been proposed for digitally interpreting schematics of electrical circuits. For example, a method by Mishra and Vinayak requires a redrawing of the circuit with boxes instead of its components, and then, using optical-character recognition (OCR), it reads the contents of these boxes.[9] This method uses a program to read a circuit; however, the circuit must first be encoded into a format that the program can read.[10] Another paper by Fukada[11] shows a method that can only operate with the prerequisite that the component schematics of an electrical circuit must fall between a size of 3 mm^2 and 2 cm^2, and connecting lines between components must be about 0.55 mm in width.

The problem with these rule-based approaches is that they require some user input to be employed. We present a new methodology that provides a pipeline for the fully automated segmentation and classification of components in electrical-circuit schematics. Thereby, our methodology incorporates tried-and-tested, highly optimized image-processing algorithms that take advantage of common attributes that are contained within electrical-circuit schematics. This enables the segmentation of these schematic images by locating the connecting wires and then segmenting the components that lie between the wires. The ability to segment electrical-circuit schematics into wires and components is the first step in automating their digital interpretation. Once the electrical-circuit components have been segmented from their schematics, they are passed on to a custom-trained OCR model that can be used for component identification and labeling; this affords in-depth digital interpretation

of the schematic image. In addition, this paper offers a proof-of-concept for an extendable option for this pipeline. This option lays the foundations for a quantitative and dynamic interpreter of electrical-circuit schematics such that their efficiencies could be predicted using a data-driven approach. A case study illustrates how this extendable option would function: using our custom-trained OCR model to identify electrical-circuit components and employing this information to plot network graphs of the circuit. These network graphs can then be used to extract matrices to find fundamental equations about voltages and currents at the junctions in the circuit, as well as extract netlists that can be directly imported into SPICE (Simulated Programming with Integrated Circuit Emphasis)[12] simulators such as LTspice[13] or advanced simulations for designing electrical-circuit operations.

II. IMAGE SEGMENTATION AND CLASSIFICATION OF COMPONENTS IN ELECTRICAL-CIRCUIT SCHEMATICS

A. Image segmentation via pattern recognition

Electrical circuits can vary greatly from one another, but one thing that they share is that they consist of various components connected via horizontal and vertical wires. The first step employed when segmenting an image of an electrical-circuit schematic involves using a thresholding method created by Otsu[14] to binarize the entire image, which separates the circuit from the background. The image is then skeletonized using the algorithm outlined by Zhang and Suen[15] to make all contours of the circuit one pixel wide. Wires are then detected using the progressive probabilistic Hough's transform (PPHT),[16–18] which identifies straight lines as per the nature of wires in a schematic diagram. Each wire then goes through various automated checks in an attempt to remove any false positives. These checks include minimum and maximum line-length detection and an inspection of the immediate area located on either side of the wire in order to determine how much information is present. If there is more information than a set threshold, then the detected wire is disregarded, as this information likely corresponds to an electrical component. The minimum and maximum line-length parameters, as well as the size of the immediate area that is inspected, can be configured to tune the algorithm for a particular diagram. The application of these steps is now illustrated via a series of images that feature the progressive processing of an electrical circuit, which we employ as the worked example throughout this paper. This worked example is shown in Fig. 1.

Figure 2 then portrays the image of this electrical-circuit schematic once it has been binarized using the method by Otsu.[14]

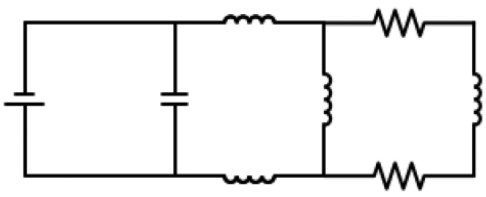

FIG. 1. Electrical-circuit schematic of the worked example.

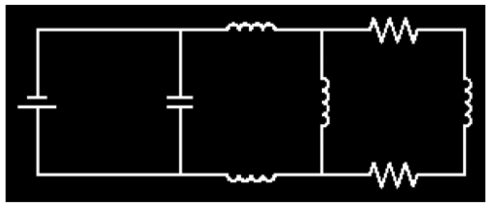

FIG. 2. A binarized image of an electrical-circuit schematic using the method by Otsu.[14]

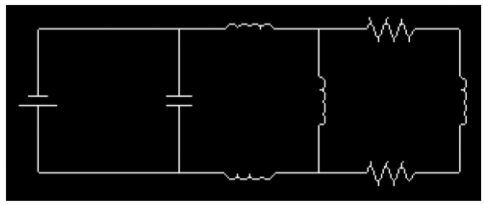

FIG. 3. A binarized skeletonized image of an electrical-circuit schematic.[15]

Figure 3 displays the binarized image shown in Fig. 2 once it has been skeletonized via the method described by Zhang and Suen.[15]

Figure 4 reveals the effect of PPHT as applied to this electrical-circuit schematic before any false positive checks have been carried out, whereby detected straight lines are differentiated by color.

The wire information that has been extracted from such an image is then stored in horizontal and vertical lists. Since wires on either side of an electrical component typically lie on the same plane, wires on the same plane can have gaps between them; these gaps are inspected to see if a component is present. Before performing each gap comparison for two wires, a check is carried out to see how far away the start of the second wire is from the end of the first wire. This check is performed to ensure that wires in different regions of the electrical-circuit schematic are not being compared, and only wires that are separated by a component are being compared. The maximum allowed gap is measured in pixels and is configurable. Figure 5 shows wires of the worked example detected by the PPHT with its detected false positives having been disregarded and with each wire list location displayed; horizontal wires are shown with "H#" and vertical wires are shown with "V#," where # is their numeric location in the list.

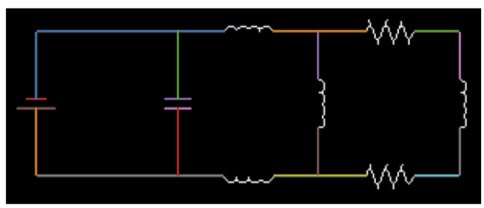

FIG. 4. A binarized skeletonized image of the worked example of an electrical-circuit schematic with the PPHT algorithm applied. Detected lines are differentiated by color.

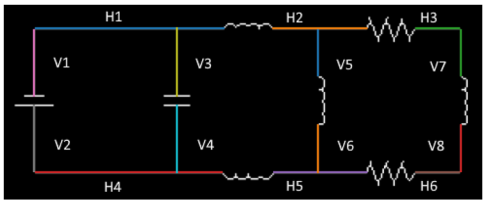

FIG. 5. A binarized skeletonized image of the worked example of an electrical-circuit schematic with the PPHT algorithm applied. Lines not classified as wires have been removed by the workflow. List locations for each detected wire are shown adjacent to them.

Figure 6 shows the electrical components that were detected by comparing the wires shown in Fig. 5. Components are marked with a red dot at their centroid and a red bounding box.

The final part of digitally segmenting the circuit diagram using pattern-recognition techniques involves finding the nodes, or wire junctions, of the circuit. All start and end points of horizontal and vertical wires are checked against each other, and where they are found to be intersecting, a node is said to be found.

B. Optical character recognition

1. Training OCR model

Once the locations of the components on the electrical-circuit schematic have been identified, a method is needed to identify the operational nature of each electrical component. Thereby, a custom neural-network-based OCR model was trained with Tesseract OCR[19] to distinguish the symbolic identification of one electrical component from another. The training dataset contained images of symbols for 30 different electrical components. Since the Tesseract OCR was designed to read text, each symbol was treated as a letter of an alphabet of electrical components, the imagery of which would vary slightly; this, in turn, mimics the various fonts that exist for a textual character. Various fonts of each electrical-component symbol were computationally generated and presented in the dataset as lines of "text" to resemble many series of disconnected electrical components.

45 000 lines of text were computationally generated in the form of images, whose content featured a randomly sequenced set of 5–30 components, each of which represents any character in the alphabet of the electrical-component symbols. The randomness was achieved by mixing up the filenames of each component image. Additionally,

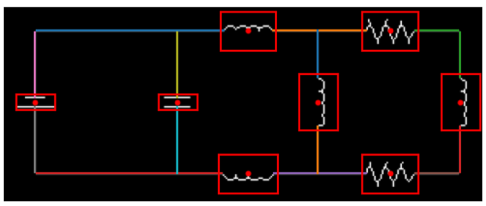

FIG. 6. A binarized skeletonized image of the worked example of an electrical-circuit schematic with PPHT applied and components detected. Detected wires are differentiated by color, detected components are marked with a red dot at the centroid and a red bounding box, and lines that are not detected remain white.

FIG. 7. Three separate component lines showing the (top) "normal," (middle) "skeletonized," and (bottom) "binary filled" hyperparameters.

each line of electrical-component symbols had been computationally generated with randomized characteristics such as stroke width, component width, and component height. To increase the variance in the training dataset even further, three different formats of electrical-component symbols were used: "normal," "skeletonized," and "binary filled," as exemplified in Fig. 7 top, middle, and bottom, respectfully. Out of the 45 000 images generated, 15 000 were normal, 15 000 were skeletonized, and 15 000 were binary filled. The selection of one of these three types of components when building a text line is determined by a hyperparameter that is to be refined in the training process. Examples of these text lines of electrical-component symbols, which are portrayed in each of the three types of formats, are shown in Fig. 7. Since OCR is a method for identifying text, each component had to be assigned a ground-truth typable character within a model that is constructed for OCR training. Tesseract OCR has support for letters (capital and lowercase), numbers, special characters, and other typescripts such as Greek and Arabic letters. Once each type of electrical component had been paired with such a text character, a set of training data was generated for training the OCR model.

The OCR model took 4 h and 23 min to train on consumer-grade hardware. The model was trained inside a virtual machine running Ubuntu 16.04.7 LTS Xenial with four dedicated CPU cores and 8 GB of RAM. The host machine was configured with an Intel i7 9700K 8-core CPU at 4.5 GHz, 8 GB of RAM at 3000 MHz, and Windows 10 as an operating system.

Once the OCR model had been trained, it was verified using a testing dataset before its implementation. Testing data were generated via the same method as that employed to curate the training dataset, except that they were generated once the OCR model had been trained to ensure they had not been seen by the model during the training process.

500 text lines in each of the three types of format shown in Fig. 7 (i.e., 1500 text lines in total) were generated and tested with the model. 1494 of these text lines returned a correct OCR result whereby all components in the text line were identified correctly, affording a 99.6% success rate on average (99.6%, 99.6%, and 99.6% for normal, filled, and skeletonized formats, respectively). The results for each individual hyperparameter are shown in Fig. 8.

These data show that this OCR method is valid for identifying the symbolic forms of electrical-circuit components when they are placed in a line, pending that there is a sufficient amount of available training data. An illustrated example of the detection process is shown in Fig. 8: the red bounding boxes represent a detected component in the line of electrical-component symbols; its symbolic meaning is recognized by OCR; and its identity is then assigned. For example, the voltmeter symbol shown in Fig. 8 was recognized as such by OCR and then labeled with the ground-truth character

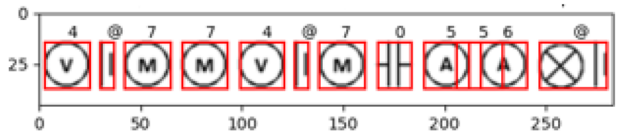

FIG. 8. A component-line image of testing data after it has been passed to the OCR model. A red bounding box represents an identified component, and the character above represents the assignment of the symbol that the OCR model has returned.

FIG. 9. Cropped components from the working example of Fig. 1, once pasted into a "text line" of components.

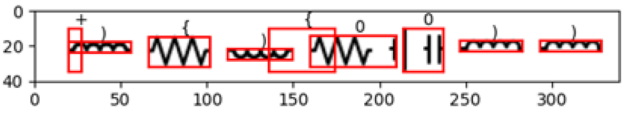

FIG. 10. The "text line" of components from the worked example of Fig. 1 has been evaluated by the OCR model.

for a voltmeter, which is "4." Similarly, the cell was labeled as "@," the motor as "7," the capacitor as "0," the ammeter as "5," and the lightbulb as "6."

Figure 8 also shows that while the bounding boxes are not always completely accurate, the sequence of components appearing in the image is identified correctly, which is the key result.

2. Application of the OCR model

Having demonstrated that the OCR model exhibits an appropriate level of efficacy, the components from the unseen images of electrical-circuit schematics were then cropped and pasted onto a text line, in a similar fashion to the process described for training the dataset. Figure 9 shows the components from the worked example of the electrical-circuit schematic from Fig. 1, before they were binarized and skeletonized.

They were then evaluated using the custom OCR model to identify the nature of each component, as shown in Fig. 10.

C. Evaluation of pattern-recognition and OCR-model efficacy on real data

The full pipeline of electrical-circuit-image segmentation and identification of each electrical-component symbol herein described was tested on images of 15 electrical-circuit schematics. 3 of these images were drawn during the testing process as part of this study; the remaining 12 images were electrical-circuit schematics that were taken from seven different published papers. The goal was to obtain schematics that contained the components that had been trained into the custom OCR model, whose circuitry configurations were sufficiently disparate from each other to ensure a diverse image set, and which also displayed various levels of complexity to test the limits of the pipeline. The schematic images from these papers were significantly more complex than the electrical-circuit schematics that had been used in testing. Examples are shown below in Fig. 11[20] and Fig. 12.[21] Diagrams of all 15 schematics are given in the supplementary material.

Any component not in contact or bounded by a red bounding box was deemed to have had its location go undetected by our workflow. In Fig. 11, this includes the MOSFET labeled "M3" and the inductor labeled "L4." In Fig. 12, this includes the switches between nodes "f" and "e," "\" and "[," and "_" and "b," as well as the switch to the left of node "o" and above node "l."

Across the 15 evaluation images of electrical circuit schematics, our pattern-recognition algorithm correctly identified the locations of 166 of the total number of 194 electrical components that were present; thus, the algorithm afforded an 85.6% success rate. Of the 15 images tested, the workflow took, on average, just 3.59 s to open

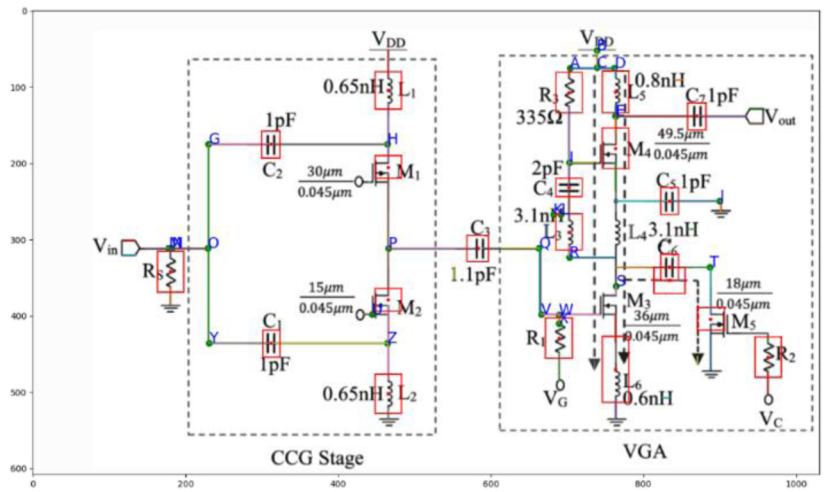

FIG. 11. A circuit diagram taken from Ref. 20, where 20 out of 22 component locations have been successfully identified by our pattern-recognition algorithm.

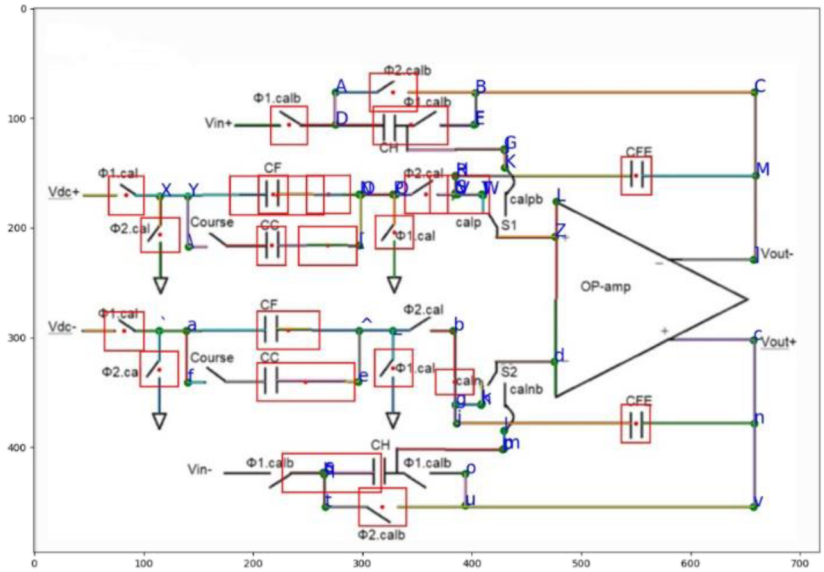

FIG. 12. A circuit diagram taken from Ref. 21, where 21 out of 26 component locations have been successfully identified by our pattern-recognition algorithm.

an image file, fully segment it, and output the OCR result, with the maximum and minimum inference times being 4.57 and 3.12 s, respectively. A breakdown of the component-detection level for each image is shown in Fig. 13. It should be noted that there is a limitation in the design of the pattern-recognition algorithm in that it only allows the detection of electrical component symbols that have two terminals. Therefore, any electrical component that contains three or more terminal points and is present in any of the 15 images that were tested was not considered when calculating the electrical component identification success rate.

Overall, these evaluation results show that the pattern-recognition algorithm performs well. Specifically, they demonstrate that the algorithm can process images of electrical-circuit-schematics successfully while maintaining a high component-detection rate. This is even the case when such schematics are significantly more complex than the electrical-circuit schematics that we had employed in testing.

The OCR model was then applied to the generated "text lines" of electrical components that had been located by the pattern-recognition algorithm, such that their symbolic form could

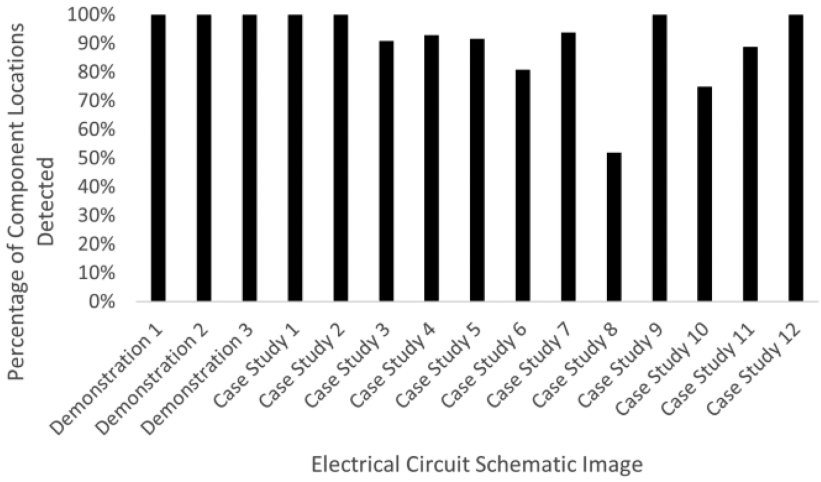

FIG. 13. Success rate (%) of the component-location detection for each schematic image. Schematics labeled "Case Study" refer to images that were taken from the literature; those labeled "demonstration" refer to schematics that were drawn in-house for the purpose of this study.

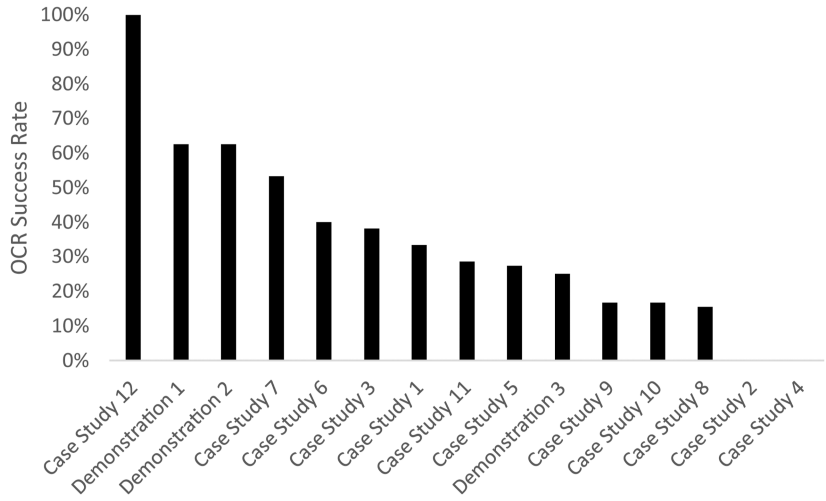

FIG. 14. OCR success rate for each electrical-circuit schematic image.

potentially be characterized. Accuracy varied significantly across the images of electrical-circuit schematics, whereby all detected electrical-component symbols were identified correctly in one image (case study 12), while none were correctly identified in others (case studies 2 and 4). Thereby, a correct result for detecting an individual component is defined as a red bounding box with the correct symbolic form, intersecting the component on the text line. False positive results occurred where bounding boxes were placed on the OCR line for a component that was not present; these were ignored in the analysis. A graph distribution showing the accuracy of the OCR model on the 15 images of electrical-circuit schematics is shown in Fig. 14.

Case Study 12 performed the best, but it is also one of the images with the least number and variety of components. Case Study 12 contains four capacitors and two inductors, with the combination of these two factors likely being the reason for the high performance in comparison to the other case studies that were tested. Analysis of case studies two and four shows that the generated "text lines" have a low resolution owing to the low-resolution of the diagram sourced. Both of their "text lines" also contain components that were not trained into the custom OCR model (i.e., wave sources), which inherently restricts the ability of the OCR model to recognize the other components. These are likely to be the main factors that led to their low detection score. The second-best-performing schematic images were two of the demonstration schematics. In common with Case Study 12, these demonstration images only contain components that have been trained into the custom OCR model; however, they each contain eight components, whereas Case Study 12 only contained 6. A differentiating factor of the two demonstration circuits is that they do not contain any annotations that may relate to features such as component labels, diagram labels, or externally referenced text of any sort. If such information is captured when the components are segmented, it will lead to a lower performance of the custom OCR model.

Therefore, these results suggest that circuits with fewer components, that contain only components trained into the custom OCR model, and whose schematics display no annotation, will perform best when using a custom OCR model to identify the components within images of electrical-circuit schematics by forming a "text line" of these components.

III. EXTENDABLE OPTION: TOWARDS PREDICTIVE CIRCUIT-DESIGN EFFICIENCIES SUBJECT TO A LARGE DATABASE OF IMAGES

There are huge advantages to using OCR to identify the "alphabet" of symbols within images that describe electrical components of circuits, especially when this capability is employed in conjunction with our demonstrated ability to identify their adjoining circuitry wiring using image segmentation. This is because the resulting decoded components of an electrical circuit and their wired connectivity data can be reframed as network graphs and matrix equations to mathematically describe how their overall electrical circuit functions. This mathematical framing of electrical-circuit schematic images opens up a way to quantitatively design and optimize the operational efficiency of electrical circuits. The following subsections demonstrate how this can be achieved via an extendable option of our methodology, using our working example for the purposes of illustration.

A. Network graph theory

The convention for drawing network graphs of electrical-circuit schematics generally follows that graph edges represent components, and graph vertices represent wire junctions. It is, therefore, the case that a network graph of an electrical circuit can be drawn directly from its electrical-circuit diagram, as we illustrate for the working example in Fig. 15.

A matrix representation of the network graph of an electrical-circuit schematic diagram can be derived from its spanning tree and co-tree.[22] These matrices use Kirchhoff's laws[23] from electronics to describe the states of the voltage and current of each vertex on the

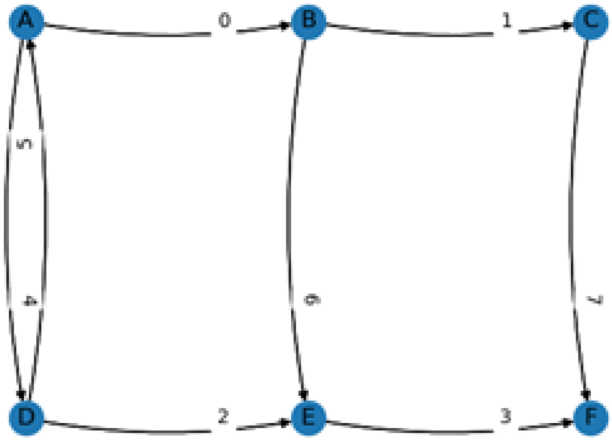

FIG. 15. A network-graph representation of the image of the electrical-circuit schematic used as our working example.

graph in terms of the other vertices. For instance, the fundamental voltage and current matrix equations for the network graph of our working example are shown in Fig. 16.

Here, the letters "i" and "v" correspond to voltage and current, and the number that follows denotes the graph edge to which the quantity corresponds on the graph shown in Fig. 15. The derivation of these matrix equations can be found in Ref. 22.

B. Combining matrix theory with our OCR workflow

In principle, the above graphs and matrices can be constructed solely from the successful image segmentation of an electrical-circuit schematic without the need for the electrical component symbols to be identified via our OCR model, although this would afford somewhat abstract graphs. The use of our OCR model to identify the nature of these electrical components adds great value to this graph theory because it allows one to map each variable of their associated matrix equations to a real electrical-component identity. The underpinning physics that interrelates the various electrical components can then be employed through these matrix equations to model the electrical behavior of the overall circuits and their sub-circuitry, where relevant, in an entirely quantitative fashion. This sub-section describes how such modeling is accomplished once the electrical component symbols have been identified via our OCR model.

The function of electrical components is governed by associated laws, e.g., a resistor always follows Ohm's law. These types of matrix equations can capture these relationships and then afford predictive

```
[      i1 + i3]
[      i0 + i2]
[-i0 - i4 + i5]  [v1 - v3 - v6 + v7]
[     -i1 + i7]  [v0 - v2 + v5 + v6]
[-i0 + i1 + i6]  [          v4 + v5]
```

FIG. 16. The fundamental current (left) and voltage (right) equations of the network graph shown in Fig. 15.

utility. The process by which a matrix equation can be built up for an electrical-circuit schematic diagram begins once we have employed the OCR model to identify the nature of each component in the electrical circuit; all components can then be labeled with a graph edge. With this pairing in hand, we can computationally solve its matrix equation via the derivation illustrated in Ref. 22. It is important to note that the ability to solve such matrix equations depends on the nature of the components that are present in the electrical-circuit diagram. The working example comprises voltage sources, resistors, capacitors, and inductors exclusively.

Such components afford matrix equations that can all be modeled as linear equations

$$V = I \times R, \tag{3.1}$$

where V is the voltage across the resistor in volts, I is the current flowing through the resistor in amps, and R is the resistance of the resistor in ohms.

The capacitor characteristic

$$I = C \times \frac{dV}{dt}, \tag{3.2}$$

where I is the current flowing through the capacitor in amps, C is the capacitance of the capacitor in farads, and $\frac{dV}{dt}$ is the change in voltage with respect to time in volts per second.

The inductor characteristic

$$V = L \times \frac{dI}{dt}, \tag{3.3}$$

where V is the voltage across the inductor in volts, L is the inductance of the inductor in Henrys, and $\frac{dI}{dt}$ is the change in current with respect to time in amps per second.

Ohm's law is a time-invariant equation; however, the characteristics of an inductor and capacitor are not. Nevertheless, one can make the approximation that inductance does not vary with time; similarly, this approach assumes that current does not vary with time in a capacitor, i.e., both properties are approximated to time invariant and thus linear laws, like Ohm's law. Realistically, this is not the case, but the variations are usually small and insignificant for inductance and capacitance,[22] so this approximation tends to hold. Using this assumption allows us to keep modeling the electrical-component functionality as linear and thus sufficiently simple, such that the associated matrix equation can be solved computationally. The matrix equation for the working example is given in Fig. 17.

Here, the letters "i," "v," "R," "L," and "C" correspond to voltage, current, resistance, inductance, and capacitance, while the number that immediately follows each letter denotes the graph edge to which the electrical component corresponds on the graph shown in Fig. 15. The letter "V" corresponds to the voltage of the source, which is located at edge 5 on the network graph.

A second assumption is also made in the formation of the matrix equation shown in Fig. 17: the components are assumed to have identical behavior regardless of the electrical circuit in which they are embedded. These assumptions are necessary, as without them, the mathematics would become intractable, such that a matrix equation could not be solved.[22]

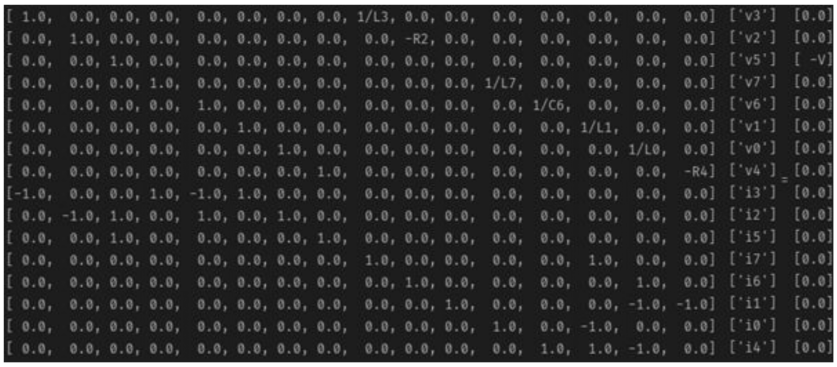

FIG. 17. The computationally solvable matrix equation for the working example of the network graph shown in Fig. 15.

C. SPICE netlist embedding

One of the great potential aspects of this workflow is the idea that it can be used to quickly and autonomously convert electrical-circuit schematic images into a format where their electrical behavior can be fully and accurately simulated once the values of the electrical components are known. This enumeration can be realized by combining our methodology with the functionality of a tool known as SPICE (Simulation Program with Integrated Circuit Emphasis).

SPICE is commonly used to simulate electrical circuits. Therein, a circuit is represented as a "netlist," which contains all of its relevant component identity and connectivity information. The syntax for a typical netlist line is as follows:

<component label/ name> <vertex 1> <vertex 2>
<component value and units>.

This information can thus be easily compiled from the output of our network-based workflow in order to create a netlist. For instance, the vertices that lie on each side of an electrical component can be found from the network-graph theory described in Sec. III A, while the identity of the component is identified via our OCR model. Our netlist file-curation procedure also adds a title line and an .end line in order to be fully compliant with the SPICE syntax. Before the .end line, a .tran line is also added with configurable values. This tells the simulator to perform a transient analysis when the operation of an electrical circuit is simulated. In order to simulate the operation of a circuit properly, its components require values; e.g., a resistor will require a known resistance so that its behavior can be modeled correctly. Our workflow provides a Boolean option to add arbitrary component values and other options to configure those arbitrary component values. This enumeration is required, as some images of electrical-circuit schematics, such as the working example, do not display component values on the figure, and besides, this workflow does not feature a procedure to identify such values automatically. Once the netlist has been formed, it can be saved as either a text file or a .net file. This file can then easily be opened by a SPICE simulator program. Therefore, we now have a practical workflow that converts the image of an electrical-circuit schematic into a text file of an electrical circuit that is suitable as input for direct simulation by specialist electrical-engineering software such as SPICE.

D. Evaluation

Network graphs were successfully created for all three images of electrical-circuit schematics that had been drawn for the aforementioned part of this study. The ability to create network graphs from them relies on the accuracy of our pattern-recognition algorithm and OCR model that we described in Sec. II. With these graphs drawn successfully, the workflow was able to use spanning trees and co-trees from matrix theory to successfully derive the fundamental voltage and current equations for each electrical circuit.

The final matrix equation, which produces equations for all graph vertices in terms of Ohm's law, the capacitor law, and the inductor law, was able to be retrieved for the working example in this study as well as for a second trial image (see supplementary material). This matrix equation was then further expanded, using the theory outlined in Sec. III A, to obtain a computationally solvable matrix equation that takes into account the values of each electrical component. Our overarching pipeline was then able to use the information that it has acquired to derive a SPICE netlist that can be used for electrical-engineering simulations. The netlist was successfully created for the working example as well as for the other two demonstration circuits; it was then saved to a file so it could be directly opened by a SPICE simulation tool.

Full evaluation results for the images of each electrical-circuit-schematic, as well as more intricate details about the steps in the pipeline, can be accessed via the GitHub repository for this project (see "Data Availability").

IV. CONCLUSIONS

This study has demonstrated a reliable pattern-recognition algorithm for segmenting images of electrical-circuit-schematic diagrams and an OCR model that then characterizes the symbolic forms of the segmented electrical components. Thereby, the input electrical-circuit imagery can be translated successfully into textual information about its contents, which a computer can more easily interpret than an image and, therefore, use more readily in downstream data analysis.

This information can also be successfully transformed into network graphs, correctly formatted SPICE netlists that can be simulated, and matrix equations that can describe the potential state of the circuits.

This workflow appears to be the first to transform the image of an electrical-circuit diagram into text without any limitations being imposed on the size or shape of any part of the circuit (e.g., wire thickness, size of electrical component[11]); furthermore, our process is automated, in contrast to previous work.[2] Nonetheless, our workflow has some limitations. For example, our pattern-recognition algorithm only interprets electrical components with up to two-terminal points; while these represent the components that most commonly appear in images of electrical-circuit schematics, it places a limit on automatically processing more complex diagrams. A machine-learning model trained to detect more niche features of a diagram, such as ground symbols, component values, input/output terminals, or components with more than two terminals, could overcome this restriction. In addition, the accuracy of the OCR model could be improved to manage more complex electrical-circuit-schematic diagrams. Both the accuracy of the netlists and matrices are also reliant on a correct OCR result. Despite this, initial testing of the OCR model shows strong evidence that OCR can be a reliable method for identifying and characterizing symbols of electrical-circuit components, pending that a sufficiently wide variety of training data are used.

The extendable option of this workflow also has a limitation in that its matrix theory cannot consider the differential equations that show how the behavior of a given electrical component can change with time. While it would be possible to develop this capability, it would become very complicated. The alternative option of using simulation tools such as SPICE software to realize this capability presents a much better solution. Thereby, our workflow creates the netlist input that is required for SPICE so that such simulations can be performed. Finally, the current algorithms in place cannot identify which way the current is flowing, and as such, this can lead to the arrows on the network graph pointing in an incorrect direction. This leads to incorrect plus and minus signs in the fundamental equations that are extracted from the matrices. Again, simulators do a much better job of this and present a possible solution.

In summary, we have presented a novel method that can automatically segment and analyze schematic images of circuit diagrams and use the output to efficiently and quickly take advantage of electrical-engineering simulation software. In time, the accuracy of the OCR model will improve such that the workflow could be applied to the large-scale autonomous processing of many schematic images of electrical circuits. This forecasted ability to auto-extract this type of image-based data by mining so many source data stands to open up the field of electrical engineering and circuit design to the latest data-analytics approaches that source big data. For example, one could apply data analytics in conjunction with graph theory to a large set of electrical-circuit representations to look for areas of operational inefficiency in the circuits, using the extendable option to our method that we described in Sec. III. Fixing such inefficiencies could lower the power consumption of a vast range of electrical appliances. In turn, this would help to offset climate change. This research could also be used to find cheaper ways of designing circuits, perhaps by identifying a redundant component in a circuit or by simply finding a more streamlined circuit; more economical circuit designs would naturally lead to reduced costs in their industrial development and fabrication.

SUPPLEMENTARY MATERIAL

A supplementary material document is provided along with this paper, showcasing the results of three custom demonstration circuits and 12 case study circuits taken from the literature processed by the segmentation workflow outlined as a part of the pipeline in this paper, as well as the workflow configuration options used.

ACKNOWLEDGMENTS

J.M.C. is grateful for the BASF/Royal Academy of Engineering Research Chair in Data-Driven Molecular Engineering of Functional Materials, which is partly supported by the STFC via the ISIS Neutron and Muon Source. J.M.C. and C.R.K. are indebted to the EPSRC for an iCASE Ph.D. studentship (voucher 210153) that is sponsored by QinetiQ; this provided C.R.K. the time to write up this work, which is based on his MPhil dissertation.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Charles R. Kelly: Data curation (lead); Formal analysis (lead); Investigation (lead); Methodology (lead); Software (lead); Validation (lead); Visualization (lead); Writing – original draft (lead). **Jacqueline M. Cole**: Conceptualization (lead); Funding acquisition (lead); Project administration (lead); Resources (lead); Supervision (lead); Writing – review & editing (lead).

DATA AVAILABILITY

The code for this work is published under the MIT license. Details on how to access it can be found at https://github.com/C-R-Kelly/CircuitSchematicImageInterpreter.

REFERENCES

[1]H. Bley, "Segmentation and preprocessing of electrical schematics using picture graphs," Comput. Vis., Graphics, Image Process. **28**(3), 271–288 (1984).
[2]G. Sussman and R. Stallman, "Heuristic techniques in computer-aided circuit analysis," IEEE Trans. Circuits Syst. **22**(11), 857–865 (1975).
[3]H. Bunke, "Automatic interpretation of lines and text in circuit diagrams," in *Pattern Recognition Theory and Applications* (Springer, 1982), pp. 297–310.
[4]K. T. Mukaddem, E. J. Beard, B. Yildirim, and J. M. Cole, "ImageDataExtractor: A tool to extract and quantify data from microscopy images," J. Chem. Inf. Model. **60**(5), 2492–2509 (2020).

[5]B. Yildirim and J. M. Cole, "Bayesian particle instance segmentation for electron microscopy image quantification," J. Chem. Inf. Model. **61**(3), 1136–1149 (2021).
[6]B. L. DeCost, B. Lei, T. Francis, and E. A. Holm, "High throughput quantitative metallography for complex microstructures using deep learning: A case study in ultrahigh carbon steel," Microsc. Microanal. **25**(1), 21–29 (2019).
[7]E. J. Beard and J. M. Cole, "ChemSchematicResolver: A toolkit to decode 2D chemical diagrams with labels and R-groups into annotated chemical named entities," J. Chem. Inf. Model. **60**(4), 2059–2072 (2020).
[8]D. M. Wilary and J. M. Cole, "ReactionDataExtractor: A tool for automated extraction of information from chemical reaction schemes," J. Chem. Inf. Model. **61**(10), 4962–4974 (2021).
[9]D. Mishra and C. Vinayak, "*Image based circuit simulation*," in *2013 International Conference on Control, Automation, Robotics and Embedded Systems (CARE)* (IEEE, Piscataway, NJ, 2013), pp. 1–4.
[10]D. Baez-Lopez, J. L. Ballesteros, and J. Pedraza-Chavez, "*Conversion of circuit schematics from a graphic display to a netlist and its applications*," in *Proceedings of the 36th Midwest Symposium on Circuits and Systems* (IEEE, Piscataway, NJ, 1993), pp. 1159–1161.
[11]Y. Fukada, "A primary algorithm for the understanding of logic circuit diagrams," Pattern Recognit. **17**(1), 125–134 (1984).
[12]L. W. Nagel and D. O. Pederson, in *SPICE: Simulation Program with Integrated Circuit Emphasis* (Electronics Research Laboratory, College of Engineering, University of California, Berkeley, 1973).
[13]See https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html for LTspice Simulator|Analog Devices (accessed August 21, 2022).
[14]N. Otsu, "A threshold selection method from gray-level histograms," IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979).
[15]T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," Commun. ACM **27**(3), 236–239 (1984).
[16]C. Galamhos, J. Matas, and J. Kittler, "Progressive probabilistic Hough transform for line detection," in *Proceedings of the 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No. PR00149)* (IEEE, 1999), pp. 554–560.
[17]R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," Commun. ACM **15**(1), 11–15 (1972).
[18]P. Hough, "Method and means for recognizing complex patterns," US3069654A1962 (1960).
[19]See https://opensource.google/projects/tesseract for GitHub—Tesseract-ocr/tesseract: Tesseract open source OCR engine (main repository) (accessed March 22, 2021).
[20]D. Kalra, V. Goyal, and M. Srivastava, "Design and performance analysis of low power LNA with variable gain current reuse technique," Analog Integr. Circuits Signal Process. **108**(2), 351–361 (2021).
[21]C. Ramamurthy, C. D. Parikh, and S. Sen, "Deterministic digital calibration technique for 1.5 bits/stage pipelined and algorithmic ADCs with finite op-amp gain and large capacitance mismatches," Circuits Syst. Signal Process. **40**(8), 3684–3702 (2021).
[22]Open University, *Graphs, Networks and Design* (Open University, Milton Keynes, 1995).
[23]G. Kirchhoff, "Ueber die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Vertheilung galvanischer Ströme geführt wird," Ann. Phys. **148**(12), 497–508 (1847).