Dobson, James E. "Beyond Computational Formalism or, Architecture Matters." *Journal of Cultural Analytics*, vol. 10, no. 3, Sept. 2025, https://doi.org/10.22148/001c.133923.

ARTICLE

Beyond Computational Formalism or, Architecture Matters

James E. Dobson[1]

[1] Dartmouth College

Keywords: neural networks, formalism, machine learning, artificial intelligence, computational literary studies

https://doi.org/10.22148/001c.133923

Journal of Cultural Analytics
Vol. 10, Issue 3, 2025

Despite frequently avowed commitments to formalist methodologies, computational literary studies (CLS) has insufficiently accounted for the importance of the formal architectures of the computational models it employs—particularly deep learning neural networks. Arguing against the tendency to treat neural networks with an abstract gloss of their operation or to focus attention on the outputs, this article posits that architecture is not merely a technical detail but a crucial site where meaning is made and historicity registered. By examining the genealogy of neural network architectures—from Frank Rosenblatt's Perceptron to contemporary transformer-based models—this article demonstrates how these architectures materially shape the capacities, outputs, and interpretive possibilities of machine learning models.

The present tendency to reduce the complex assemblage of technologies, social and cultural environments, applications and platforms, corporate imperatives, and ideology to simply "AI" has not gone unnoticed, and several critics and theorists have sought to restore some grounding to this lofty signifier (Lindgren; Pasquinelli; Crawford). Even in these attempts to add specificity and clarify concepts and methods, however, major differences in the specific forms of computation used are usually elided. The rapid pace of development of new applications and models—and the subsequent obsolescence of once cutting-edge technologies—intentionally abstract and deliberately obscured platforms, and a lack of interest in or knowledge of technical and operational details have combined to frustrate the task of understanding the significance of deep learning, especially in terms appropriate to the humanities. Deep learning, as a class of particularly dense neural networks, should be understood primarily as an architectural feature; the deep in deep learning registers in its name the importance of network architecture and is the distinguishing feature of a hierarchical arrangement of many layers of network components, modules, and blocks. The blocks used in common contemporary transformer-based Large Language Models (LLMs) are themselves small but complex mini-architectures. These range from relatively simple two-layer fully connected networks, known as Multi-Layer Perceptrons (MLPs), to self-attention blocks implementing multi-head attention, those key features that have contributed significantly to the impressive capabilities of these models. It is because of the formal innovations of recent deep learning networks instantiated in specific architectures and paired with training data, specific tasks, and fine-tuning data that this most recent moment in the long history of AI has been possible.

In the case of neural networks, as with so much more, form is function. Different architectures establish different possible universes of meaning. In taking up questions and problems that emerge at the intersection of formalist practices in literary, cultural, and media studies and those found in the use of the term in the computational sciences, this essay argues that, despite its long-term investment in formalism, computational literary studies has not been formal enough. It does so through a critical reading of two essays by literary studies scholars that stage formalist arguments about the ahistoricity of neural networks without taking into account the formal features of these networks. My argument, in short, is that because of the process by which outputs are derived from computational transformations, any formal analysis of these outputs must take into account a formal analysis of the transform that created them.

Form and formalism are invoked in multiple ways in this essay. Form is used at times in the familiar way in which literary scholars use the term, to name aesthetic or cultural features of an object. To the study of such forms, of course, we give the name formalism. An approach termed computational formalism might be concerned with recognition, cataloging, and categorization of these forms (say for example, genres or plots). To encode, directly or indirectly, knowledge about these forms with computation means formalizing this knowledge (in terms of samples, explicit features, etc.) in a model. All models, as Annabel Jane Wharton argues, differ from their referents, but computational models involve formalizations (code, algorithms, pipelines, data) that register that difference in the form of assumptions about the referent. Nonetheless, Wharton's succinct observation that "all models are entangled in discourse; they have histories, and they act politically" (Wharton 17) applies equally to computational models. In what follows, I am arguing that the form—as instantiated in computational transformations, including neural network models—of the formalizations used in computational literary studies have similar discursive entanglements, histories, and politics. This is to say that the study of forms, in the literary sense, with computation cannot be separated from the forms of computation used in that analysis and the historicity of those computational transformations.

Computational literary studies (CLS), the digital humanities, computational formalism, cultural analytics—whatever we call the admixture of computational methods and humanities methods and objects, depends entirely on formalism. This is because computation always requires some degree of formalization, although as a user of computational applications or tools one might not immediately realize it. The processing of information, development of algorithms, definition of programming languages and their syntaxes, and the representation of data depend upon discrete processes and concretization, in other words, formalization. In computational research, there is a formalism embedded in the framing of questions, the articulation of a hypothesis, and the creation of a method through which this hypothesis can be evaluated. While more loosely structured and various, there is a formalism found in the presentation of inputs. To be more concrete, take the encoding of textual inputs used in common contemporary language model-based chatbots. These inputs are frequently tagged with labels or structured into a paired conversational "prompt." Beyond chatbot interfaces, all inputs to language models are rendered into formalized structures through tokenization. Such processes are one aspect of the formalization that makes computation possible. Formalization in computation exists, in part, to enable comparison and thus it asserts some degree of normalization and standardization over its objects.

The use of form and formalism in the humanities has a slightly different history and import than those found in computational science, but it has long been acknowledged that formalist approaches, and especially structuralist approaches to literary studies, are well-suited to the sort of formalization imposed by computation. Northrop Frye, for example, argued as much in his talk on "Literary and Mechanical Models" at the annual joint meeting of the Association for Literary and Linguistic Computing and the Association for Computers and the Humanities in 1989. Marveling at what was possible, even at this early stage of "humanities computing," Frye remarked that "at present, in the humanities, computers are doing an immense amount of word-crunching, and could easily do much more. Concordances have multiplied; dictionaries are no longer assembled from hand-written slips; in the study of literature the prospect opens up of having the entire verbal corpus of any given literature placed within easy reach" (Frye 8). This work, this modeling, Frye hoped, would lead to the fulfillment of his dream of a scientific criticism. "Critical schools," he writes, "like philosophical ones, are better thought of as programming models. The importance of the computer is in bringing them down to manageable scope, so that their essential assumptions can be worked through in a reasonable time before they modulate into or merge with something else" (Frye 7). This is to say that it is not just the speed or size of corpuses and models, but primarily the coding or scoping, as it were, of assumptions into variables that would make it possible to operationalize a scientific criticism.

This more basic insight, that structuralist and formalist practices are highly compatible with computation because of the strict encoding of assumptions necessary for computing—categories, genres, types, tropes, figures, etc—has informed much work in the digital humanities. That the affordances of the limited set of forms with which to encode these assumptions available within computing might shape the assumptions of humanities researchers has had far less of an influence than previously thought. Two recent monographs that make use of computational formalism in their titles are exceptions to the dominant tendency in computational work in the humanities to reduce formalist claims to only the outputs of transformations; Martin Paul Eve's *Close Reading with Computers*: *Textual Scholarship, Computational Formalism, and David Mitchell's Cloud Atlas* (2019) and Amanda Wasielewski's *Computational Formalism: Art History and Machine Learning* (2023) both make an effort to deploy versions what Wasielewski terms "methodological self-criticality" (Wasielewski 34) in their respective applications of computation to text and images. As computational methods increase in complexity and the forms with which one encodes their questions and assumptions shift from more-or-less interpretable models of well-defined units (words, sentences, paragraphs, books, etc.) to distributed sets of features across the units and layers of deep learning networks, identifying these forms and the ways in which they shape the work of humanists has simultaneously become increasingly difficult and important.

It perhaps is not surprising that the use of machine learning in humanistic research, especially with the advent of large language models, has prompted new questions about familiar concerns in literary studies with regard to formalism, reading, and interpretation (not to mention questions about intentionality and authorship). The debates about distant and close reading in the earlier digital humanities discourse contrasted the sequential and situated human reading of text with models that re-presented information about and aspects of texts. Because the common output objects from contemporary generative models are not numerical (insofar as how these outputs are typically decoded and displayed to users of web or application-based generative AI tools) but texts, many scholars have been turned back once again to key problematics in the history and practice of textual interpretation. Hermeneutics, as I have argued elsewhere, becomes a compelling framework once more, although with some work to be done on a revision of the hermeneutical concept of a text and its relation to speech (Dobson, "On Reading and Interpreting Black Box Deep Neural Networks"; Dobson, "Vector Hermeneutics"). At the same time, these familiar-looking textual outputs too frequently draw attention away from the formal procedures by which these models were created (Offert and Dhaliwal). These textual objects are informed by the partially obscured and increasingly opaque transformations that have created them. While it is well known that the nature of these transformations is shaped by the input or prompt, training data used, various types of parameterization, post-training fine-tuning procedures and the choice of model or algorithm, too little attention has been paid to the formal architecture of the models themselves. These architectures give shape to the possible meanings found within the model and their affordances determine what can be modeled. Architecture, of both the network and the pipeline in which it is embedded, structures meaning, both internal to the model and its outputs. While there is much that we do not know about how meaning is made in deep learning networks, when it comes to contemporary language models, architectural features are recognized as key to their impressive performance and the key sites for understanding how these networks model language. The obscurity and opacity surrounding the term AI can present complications, or at least undermine methodological commitments, when deep learning models are used as part of work in computational literary studies. This is especially true of work in CLS that is characterized by (or invested in) formalist concerns.

Reading the Form of Neural Networks

Claims about the ahistoricity of neural networks are central to this essay's larger argument that in its use of machine learning and artificial intelligence, computational literary studies lacks sufficient formalism. The ahistoricity depends on abstract rather than formal accounts of these networks and models. While it has been acknowledged that deep learning methods and distributed pre-trained models have histories in the material sense—which is to say that they were created in particular historical moments and are informed and shaped by the technical and cultural affordances of their creators' milieu—the historicity of the networks, weights, and their associated pipelines have been mostly ignored. Both forms of historicity give rise to situated transformations: model inputs encounter the weight of history through their processing and through the architectural constraints of models, and the outputs are marked by these encounters. This historicity, as the following critical readings and analysis will demonstrate, is registered in the formal features of the networks and models. These, in turn, inform the interpretive possibilities of model outputs.

In order to gloss the major presupposition supporting the oft-repeated claim that neural networks are ahistorical, we need to return to the foundational problem of the temporality of learning and training procedures in machine learning. While machine learning can be used in both explanatory (and related descriptive tasks) and predictive tasks (Shmueli), the primary purpose of machine learning is to learn from past patterns to predict future patterns. That foundational activity and purpose promotes the critique of the historicity of machine learning from a secondary to a primary concern. In explanatory machine learning, for example in the case of a classifier trained to separate samples into discrete categories, confidence in the decisions made by the classifier rests in its ability to generalize its learned criteria beyond the limited set of training data. Predictive machine learning has become much more important due to the advent of deep learning neural networks that alter the network's internal representation of what has been learned (essentially the decision criteria) as a result of its predictive performance, such as the now well-known masked language modeling or next token/ sentence prediction tasks used to train large language models. Contemporary generative AI applications are built on top of selection mechanisms and models from probabilities produced from predictive models. When deployed with neural networks, both categories of machine learning model learned representations from historical data, which is to say data presented to the network prior to the moment of inference of new inputs.

The frequent description of the moment of training as clear and distinct from inference, which to say deployed for evaluation or use, combined with the fact that during typical training periods of many neural networks the inputs are iterated through the network multiple times gives rise to the sense that it is only the break between training and inference that produces the temporality of the model and that everything that comes before that break exists in an atemporal continuous iteration of the entire stream of inputs. The graphic representation of model loss, which is to say the difference between the predicted and labeled (true) values, gives form to some aspects of the historicity of model training. A curve in these loss visualizations, ideally trending downward, records the history of training epochs in the form of loss values. Due to the very large number of inputs, developers of contemporary large transformer-based neural networks make use of few epochs. Batched inputs, however, are also iteratively presented to even the largest of networks. This is the inner loop of the outer epochal training loop. Dividing inputs into parallelizable batches incrementally updates parameters, informing, in architecture-specific ways, what is learned by the model. Different network architectures implement different learning strategies and as a result have different modes of historicity, registered in the temporal relation among the stream of inputs.

When used in what we might think of as a read-write mode, a mode characterized by the iterative processing mentioned above during initial training or during the modification of that training in a process known as fine-tuning, deep learning networks continually modify their parameters. Deployed in read-only mode for inference or prediction/generation, these parameters are no longer subject to updating, although the models may learn from patterns provided as input, filtered through what has already been learned and fixed in the model's internal representations. As the number of inputs scales—in large language models the size of the input is referred to as a context window—learning from these inputs becomes increasingly possible and the distinction between learning that takes place during training and during inference from inputs becomes less clear (Min et al.). There are important differences to be found in the operation of models and in the training procedures used in the various neural network architectures, especially considering that these are venerable technologies with almost seventy years of history. In order to characterize neural networks as lacking historicity, one would thus need to make a clear distinction between training and inference and between learning and prediction. Such claims generally result from abstract rather than concrete and technical accounts of their operation.

Strangely enough, it is in an essay on the form of neural networks that Matthew Kirschenbaum connects the formal attributes of neural networks to what he presents as their ahistoricity. In Kirschenbaum's "Spec Acts: Reading Form in Recurrent Neural Networks," Kirschenbaum argues that neural networks are speculative instruments that are "inimical to historicist thinking" (Kirschenbaum 364). Neural networks "defy the imperative to always historicize," he argues, because these networks are "progressive and unidirectional" (364). Kirschenbaum links the problem of determining the history of the network, the sequence of prior inputs that have been seen in the process of training the network, to the form of the network, but that form itself never emerges as a concrete object of analysis. The network of concern to Kirschenbaum is the operation and meaning of a particular kind or subcategory of recurrent neural network (RNN) known as a long short-term memory (LSTM) network. In place of an analysis of LSTM networks, Kirschenbaum produces generalizations about neural networks in the abstract. The LSTM network is of special interest to Kirschenbaum because it was used by Ross Goodwin in 2017 to process a variety of inputs ("temporal, vox, locative, or pictorial" [361]) extracted from Goodwin's car and a network of sensors (supplemented and powered by a set of pre-trained models and data) and produced while driving from Brooklyn, New York to New Orleans, Louisiana. The outputs of this LSTM would become a generated text that Goodwin titled *1 the Road.* While Kirschenbaum acknowledges that he is shifting from specifics to the general—he writes, after mentioning OpenAI's transformer-based GPT-3 generative network, that "not all of these natural language processing technologies are neural networks of the precise type Goodwin is using, but all of them rely on some broadly shared principles of what is generally known as machine learning or deep learning" (362)—he leaves out the particular affordances of the LSTM network and instead focuses on the description of highly abstract artificial neural networks that forecloses entirely his ability to address the question of form.

Despite the sudden dominance of RNN and LSTM networks in the 2010s (prior to the advent of the transformer architecture), these networks have a surprisingly long history. The LSTM network was first introduced in 1995 by Sepp Hochreiter and Jürgen Schmidhuber (Hochreiter and Schmidhuber). There were two major innovations of LSTM networks: first, a novel network architecture constructed from the base of a RNN with a memory cell for holding stored values along with a gate unit; second, a new learning algorithm capable of addressing varied, unevenly distributed inputs from time-series samples. Hochreiter and Schmidhuber describe the relation of this algorithm to the new architecture as such: "gradient-based algorithm for an architecture enforcing constant (thus, neither exploding nor vanishing) error flow through internal states of special units (provided the gradient computation is truncated at certain architecture-specific points; this does not affect long-term error flow, though)" (Hochreiter and Schmidhuber

1736). As they make clear, there is a tight connection between the learning algorithm and the network architecture; the generative capabilities of LSTM networks are the result of the orchestration between an architecture-specific learning algorithm and the architecture itself, in particular the memory cells that enable the creation of a “long short-term” memory of inputs. While LSTM networks might be used for similar purposes as other neural networks, they have a particular architecture that lends to them the affordances of producing convincing output objects from input samples based on a capacity for potentially extracting information from the history of encoded data stored in their memory cells that made them useful in early generative work (Hochreiter and Schmidhuber 1743–1746).

In explicating what he characterizes as the “fully activated formalism, form unconstrained by matter” (378) of neural networks, Kirschenbaum does not offer an account of these memory cells and the different ways in which LSTM networks can make use of histories (Hochreiter and Schmidhuber, in a now-familiar formula, assign agency to the network when they write that “the net can use in_j to decide when to keep or override information in memory cell c_j and out_j to decide when to access memory cell c_j and when to prevent other units from being perturbed by c_j.” [Hochreiter and Schmidhuber 1745]). Instead of an analysis of these interesting formal features of the LSTM network, Kirschenbaum relies upon Orit Halpern’s account of abstract neural networks and a description of the operation of neurons within that abstract network that he has extracted from Halpern. He argues that “a neural network knows only prediction and prognostication, never pastness. A neural network never *ever* historicizes. The gearbox is all additive, accumulative” (379). He continues, directly citing from Halpern:

> Media historian Orit Halpern helpfully puts the basic underlying concepts in layperson’s terms: “From within the net, one cannot determine which neurons fired to excite the current situation,” she writes. “From within a net (or network) the boundary between perception and cognition, the separation between interiority and exteriority, and the organization of causal time are indifferentiable.” In other words (again Halpern’s): “the temporality of the net is preemptive, it always operates in the future perfect tense. . . devoid of historical temporalities.” (379)

In shifting from the particular to the general, Kirschenbaum elides key differences in the operations and in the design of these networks. The LSTM network, as we have seen, has multiple possible historical states (using or overriding information stored in memory cells with new inputs). Even more problematic, however, is the fact that the neural networks described by Halpern are not the generic networks posited by Kirschenbaum, but actual existing and historically specific networks. In *Beautiful Data* (2014), the source for the above quotation, Halpern was not describing contemporary deep learning networks but very early implementations of simplified linear learning models (Halpern 157). These are networks without memory cells, softmax layers, convolution layers, backpropagation, stochastic gradient descent algorithms, without a suite of optimizers, multiple stages of training, and reinforcement learning. These are quite different architectures from the LSTM network and contemporary deep learning networks; they implement different temporalities in their training and inference methods, and they had very different affordances for understanding their operation.

The model invoked by Halpern is the best known and most influential mid-twentieth century neural network. This is the Perceptron, initially described by Cornell University psychologist Frank Rosenblatt in 1957. In Rosenblatt's conception of his project, he makes it clear that the Perceptron was not an artificial intelligence device or an algorithm for pattern matching but instead it was a simplified brain model. The visual perception system is difficult to map and understand, even in relatively uncomplicated animal models such as the frog and later cat models that inspired Rosenblatt's design. In his explication of his new model, he refers to the "big mystery" of "how the apparently unintelligible tangle of connections in the association area manages to record the fact that a beam of light or a landscape is actually seen, or a voice heard, and how the impulses from the stimulus are interpreted in such a manner as to enable the organism to select the appropriate response channel, and no other" (Rosenblatt, *Design of an Intelligent Automaton* 6). Unlike today's deep learning networks that prioritize performance over all other considerations, Rosenblatt's artificial neural network was designed to enable human interpretability. He described it as "a man-made system whose 'anatomy' is known to the last detail" (6). This "anatomy," the network architecture, would eventually enable an entire research program and comparative studies between behavioral studies of humans and animals and results from experimentation with machine learning models (Rosenblatt and Keseler). "Although this system now exists only in concept," Rosenblatt continues in this same report, "it has been shown to be capable of the same functions of sensing, recognition, retention, and response selection as its biological counterpart" (6). The ability to instrument the model and trace what has been learned from supplied inputs to eventual response outputs was enabled by its architecture, its anatomy.

It would be worthwhile to briefly review the architecture of Rosenblatt's Perceptron. Despite misconceptions and wide-spread contemporary references to Multi-Layer Perceptrons (MLPs), Rosenblatt's Perceptron was implemented as a multi-layer network. He described the need for three layers, even in his earliest conceptual models (Rosenblatt, *The Perceptron*). That initial photoperceptron network was composed of three systems in three layers. The first he termed the Sensory or "S" System. This is the input layer, and, as the Perceptron was designed for computer vision, Rosenblatt conceptualized this layer "as a set of points in a TV raster, or as a set of photocells" (4). Each "neuron" or element was connected to one or more elements in the next layer, the Association or "A" System. In contemporary terms, this second layer is a "hidden layer." This layer receives inputs from the S System and transmits its output to the next layer. Using fixed parameters for a threshold value, it sends output forward if that threshold was reached from the sum of input values from neurons in the first layer. Rosenblatt writes of the second layer neurons: "The value of an A-unit's output will vary with its history, and acts as a counter, or register for the memory-function of the system" (4). He called the third and final layer the Response or "R" System. In the design for the machine, it "consists typically of a relatively small number of units, which may operate type-bars or signal lights, and which are activated when the mean or net value of the signals received from the A-system exceeds a critical level" (4). The Perceptron's architecture was first imagined as a physical computing machine; it was intended to be an alternative to traditional digital computers, but out of necessity, primarily the costs involved in developing the complex hardware required to implement these multiple layers at a reasonable scale, it was initially simulated on a conventional, general-purpose digital computer—an IBM 704. This foundational network bears the traces of its design as a physical machine—and thus is partly responsible for the name "machine learning." It was, in fact, its architectural limitations that would come to determine the reception and fate of Rosenblatt's Perceptron (Dobson, *Birth of Computer Vision*) and early neural networks in general.

Architecture, in short, matters. One of the now canonical papers in the history of deep learning, the "Going Deeper with Convolutions" conference paper that announced the Inception convolutional neural network says as much (Szegedy et al.). Despite the emphasis on depth in "going deeper," what we learn is that it is not just the addition of complexity, the adding of more layers, that has produced the improvements in this new model, but instead the organization of the network—in short, its architecture. In this paper, the authors write of their state-of-the-art benchmark crushing network:

> Our GoogLeNet submission to ILSVRC 2014 actually uses 12 times fewer parameters than the winning architecture...from two years ago, while being significantly more accurate. On the object detection front, the biggest gains have not come from naive application of bigger and bigger deep networks, but from the synergy of deep architectures and classical computer vision, like the R-CNN algorithm. (Szegedy et al. 1)

The title riffs on the text "we need to go deeper" from a popular meme connected with 2010 film *Inception*. The authors explain that "the word 'deep' is used in two different meanings: first of all, in the sense that we introduce a new level of organization in the form of the 'Inception module' and also in the more direct sense of increased network depth" (Szegedy et al. 1). It is the architectural feature of recursivity found in a network with subnetworks or modules that characterizes GoogLeNet and the convolutional layers that organize its image ontology (Dobson, "Objective Vision"). The classifications made by a convolutional neural network like GoogLeNet/Inception cannot be separated from that image ontology that makes for no distinction, for example, between background and foreground.

Without any doubt, today's most important machine learning architecture is the transformer. While "attention is all you need," as it was stated in the title of a conference paper announcing this new architecture, suggests that this new paradigm is a simplification, a reduction of complexity, it is the addition of an architectural feature that makes these networks distinct from previous techniques (Vaswani). Unlike earlier artificial neural network architectures, including Rosenblatt's Perceptron and CNNs, transformers were not designed to replicate even rudimentary human or other animal cognitive processes. Discarding the convolutional layers of previous models, transformers implement a new feature, self-attention, along with fully connected feed-forward layers. Attention is a key and value mapping between outputs and queries that results in weighted values that register the significance of the relations between these keys and queries. Multi-head attention allows for greater parallelization and potentially specialization of these units. "As side benefit," the authors write, "self-attention could yield more interpretable models. We inspect attention distributions from our models and present and discuss examples in the appendix. Not only do individual attention heads clearly learn to perform different tasks, many appear to exhibit behavior related to the syntactic and semantic structure of the sentences" (Vaswani et al. 7). This same insight, that architecture-specific components are observed to be specialized for some syntactic relations and other linguistic functions, has also been made on behalf of other transformer-based architectures including BERT (Rogers et al.).

In contemporary machine learning it has become trivial to load, modify, and swap models using different neural networks to analyze or model the same inputs. Machine learning pipelines encourage experimentation to select better performing networks and to construct ensemble models from multiple models and networks. This high degree of modularity obscures differences among the potential architectures. That said, even casual users of deep learning models are increasingly aware of some major architectural features, for example, the number of parameters, layers, inputs, and outputs. At the same time, the network graph has become central to much thinking about deep learning. Neural networks, as Ranjodh Singh Dhaliwal, Théo Lepage-Richer, and Lucy Suchman argue, are best understood "not as being created, discovered, found, generated, or even studied" but as rendered, which is to say broken down, disassembled, and made again (Dhaliwal et al. 13). Thinking in terms of these complex graphs has enabled the construction of stacked networks and sub-networks. In commonly used packages such as Pytorch (Paszke et al.) and Tensorflow (Abadi et al.), networks are manipulated in graph form. The model state saved after modification, for example, preserves the graph and its organization and when loading a saved model, contemporary machine learning frameworks traverse or walk through and verify the integrity of the architecture through its graph. New networks are designed as graphs. They are discussed and debated as graphs. Depicting neural networks visually as graphs foregrounds architectural differences and enables them to be more effectively analyzed, compared, and revised. These graphs depict network architecture, and their architectural features are key to understanding the flow of information through the network.

The Multiple Stages of Neural Network Training

Not only do the selected and implemented architectures determine the capabilities and meaning of the outputs of neural networks, but so too does the historicity involved in the recent paradigm shift in model training. This historicity is fundamentally different from that found in earlier paradigms, in which a fixed dataset determined the horizon of possibilities for meaning at the moment of training. The advent of self-supervised learning helped make the transition to this paradigm possible. The term self-supervised describes model training procedures characterized by minimal operator involvement; the initial or pre-training of transformer-based large language models typically does not include labeled information, only perhaps a generic task like next token or sentence prediction. While previous generations of neural language models could be trained, fitted, and incrementally retrained, the formalized division of training in transformer models is what has given these models their impressive capabilities. This division of training has also rendered them acceptable for interaction in the form of chatbots. Unsupervised pretraining on massive datasets is what makes a large language model both "large" and a language model. Researchers speculate that this unsupervised training on mountains of language samples provides the models with some degree of generalized specialization, which is to say a sense of language features drawn from a statistical model. Architecture-defined and determined components such as individual neurons or specific attention heads in the networks of pretrained large language models might learn to recognize specific parts of speech or sets of tokens assumed to share some features, like numerals or punctuation. While these base models might encode information needed to predict the answer to a math question in the form of next token predictions that returns the correct answer with a high degree of probability, they are quite limited in predictive power and restricted in their responses.

In "Poetry Will Not Optimize; or, What is Literature to AI," Michelle Elam examines the possibilities of literary experimentation with OpenAI's transformer-based GPT-3 (Elam). Like Kirschenbaum, Elam is interested in reading the texts produced through generative uses of deep learning models, what is now commonly called generative AI. She also interrogates the historicity of generative models. Reading Gwern Branwen and Shawn Presser's prompting of GPT-3 with the first four lines of Maya Angelou's "Still I Rise" poem enables Elam to critique the "flattening intergenerational significations" (Elam 287) produced by GPT-3 from samples of what may be potential Black texts across time. The text produced to complete Angelou's poem is incoherent, "a cringeworthy jumble of blues, Black power, racial uplift, and Ole Man River minstrel" (Elam 288). This temporal collapse of text fragments motivates Elam's development of "algorithmic ahistoricity," a conceptualization of the way in which, as she argues, large language models may freeze their inputs and treat all text samples as if they were produced in the same historical moment. Another dimension of this same problem, she argues, is found even in experiments that would be restricted to training on the oeuvre of a single author. Elam provides the example of August Wilson's Century Cycle, a set of ten plays, each of which takes up the representation of Black life in a different decade of the twentieth century. Wilson works both within and against his own periodizing by drawing on linguistic resources that are "not rigidly specific to any particular time and place" (288). Elam's critique of AI is rooted in the problem of machine learning's difficulty in modeling dialect, diachronic language, and idiolect, which is to say language's change over time and place as well as those objects that self-consciously trouble their own purported historicity.

These are important problems that large language modeling did not entirely address in the shift from static to contextual embeddings. The previous generation of neural language models used static embeddings in which a single vector was assigned to each word or subword (i.e., token) regardless of the context in which that word was used. To preserve different historical uses of language, strategies were developed to create and compare separate models that were trained on periodized historical sources (Hamilton et al.). Contextual embeddings, such as those produced by transformer-based networks, encoded the same word or token with different vector values depending upon the token's position in the fragment passed through the network and these neighboring tokens. Models can be trained in ways that enable historical differentiation of training data and inputs. This approach to period-specific training typically involves the creation of multiple models or checkpoints on data drawn from sources published during the period in question. After these models are created, inference can be performed on the individual models and comparisons can be made that might lead to insights about discursive changes and historical drift through the embedding space. Nonetheless, there are some problems introduced with this model of historicity. The largest being that such training generally assumes an even distribution of discourse throughout the training data. Also, while iterative and periodized training can provide a more restricted semantic space for modeling and text generation, it does not solve the other problems raised by Elam, that of idiolectic language patterns or texts that play with synchronic or anachronistic language.

While Elam is right to say that the training sets, the large samples of language provided as input to contemporary machine learning models, are dehistoricized within the individual stages of training, the resulting model's treatment of these data are historicized. While this might seem like a fine distinction, that historicity has important consequences for how meaning is made from these dehistoricized data and how history enters into machine learning. Elam describes work like the above of periodized training as limited by the network's treatment of the resulting data:

> To be clear, of course one can train an algorithm on historically accurate data—that is not my point. Rather, the challenge lies with what gets counted as usable data in the first place: the historical information for training sets is necessarily treated as a set of static points—information already reduced and rendered interpretable as usable data. One can add new or different data but data itself are treated as ahistorical for the purposes of programming. (Elam 286)

Because Elam is interested in the generative capacity of recent language models, there are some architectural specifics that inform and alter the historicity of the model following its initial pretraining stage. The advent of fine-tuning, the second stage of the now-normalized two-stage training process for transformers, modifies the information gleaned from the pretraining with some degree of supervision. That supervision can be rather heavy handed. To take a widely used and well-known example, classifiers built on transformer models that used labeled input samples of language to fine tune the model to recognize the difference between positive or negative movie reviews. These procedures modify the models through architecture-defined features. There are also light-weight versions of supervised fine tuning. Reinforcement Learning from Human Feedback or RLHF, would be one of these. OpenAI has popularized and improved RLHF techniques in their quest to make ChatGPT more friendly, helpful, and most importantly, inoffensive (Ouyang et al.). RLHF fine tuning of ChatGPT takes place through OpenAI-created directives and preferences held and decisions made by evaluators interacting with the model. In modifying the pretrained language model to respond according to human preferences for instruction-based interaction, OpenAI reinscribes aspects of the model and alters its historicity. That same process can be seen in the now many open-source LLMs available in either their foundation or their initial pre-trained form (potentially several checkpoints during the iterative training process) as well as in the form of an instruction fine-tuned model using a variety of supervised fine-tuning processes including RLHF and Direct Preference Optimization (DPO). Such fine-tuning procedures alter the model. They do so in ways that are presently hard to understand. The use of fine tuning introduces an additional layer—a metaphorical layer—of complexity to the model and its interpretability. By not being able to distinguish what has been learned in the form of pretraining and what has been learned from fine tuning, people interacting with these models are left to guess the source of predictions. The widespread use of such models, without watermarking or even a relatively simple hash of the model to guarantee that we are in fact using "GPT-4o" or "gpt-4-turbo" as delivered by the now almost completely closed OpenAI, exposes users to model manipulation, unexpected ideological shifts, and other nefarious possibilities.

While we are presently unsure how to differentiate, in the examination of models and their outputs, pretrained from fine-tuned models, there are other temporally significant aspects to such models. There are, for example, important distinctions in the representations provided by models. Consider that the difference between "in-context" and "in weights" embeddings. The term "in weights" has been created to name embeddings, the representations, provided by the model, either as the result of learning during pretraining or from a fine-tuning process. These representations are different from those termed "in-context" (Brown et al.). These are learned representations created during model inference. In generative models, the prompt is embedded as part of the "in-context" learning. In Elam's example of Branwen and Presser's generated poem, the first four lines of Angelou's poem are passed in this manner (along with other potential inputs found in the present history of prompts). Additional information that alters the model's response can be provided at this point. Retrieval Augmented Generation (RAG) is a generic name for a pipeline built on combining information retrieval techniques with generation to improve the quality of responses and reference to sources by extracting relevant context to append to prompted queries as input for generation (Lewis et al.). OpenAI's GPT-4 has the capacity of 128k tokens to be provided as part of this in-context learning. While these do not modify the model's parameters, they will change the model's behavior. Typically, this additional context is provided to improve performance on tasks. In-context learning is linked to prompt tuning or prompt engineering; this allows for out-of-training examples can be used to modify predictions. This has opened up new avenues of critique and also difficulties in interpreting models. Open to injection, something akin to SQL injection in which instructions are added to data for processing by a database, the prompt is a complex input. The use of in-context learning adds another site for historicizing the model and produces a complex temporality by combining two moments from the past (pretrained + fine-tuned) with the present (inference of in-context inputs). Even read-only models—models no longer in training mode—can be prompted with large context windows to add additional information, examples, or instructions. Such modes of inference without updating, which is to say gradient updates of the model, makes for a complex historicization of inference from inputs as this activity brings together multiple distinct moments of historicity.

Interpretation: Mechanistic and Humanistic

While the previous analysis concerned the historicization of specific network architectures and their pipelines, an important additional site of analysis for contesting the ahistoricity of neural networks is found in model features, such as attention heads in the case of transformers, and in the parameters, which is to say the weights and biases, which now number in the billions and trillions. The most promising method of interpreting and explaining the operation of large language models at present is known as mechanistic interpretation (Saphra and Wiegreffe). Research in the area of mechanistic interpretability is especially concerned with the operation of individual components within machine learning models and targets specific features. This is a highly architecture-specific mode of analysis that seeks to examine and instrument components of neural network architectures (Sakarvadia et al.). Instrumentation sometimes takes place through the attaching of probes, in the form of programmatic hooks, to methods (e.g., feed forward or back propagation functions). Some researchers propose the monitoring of specific "neurons" or dropping or ablating these elements from the network (Gurnee et al.) When Kirschenbaum writes of the formal dimensions of neural networks in his essay, he means to invoke the idea of form without material, which may have explanatory power for certain modes of reading, but when he argues that neural networks cannot be probed, he takes what might, perhaps, be a local feature of a particular kind of network as a general attribute of neural networks as such. Kirschenbaum writes:

> What we read when reading neural networks, I want to argue, is a kind of fully activated formalism, form unconstrained by matter, form whose manifestations have no necessary base in a prior substance or substrate. This is the particular poetry of vector space. Neither the input nor the output is 'immaterial'; but the transactions that give rise to form are, for all intents and purposes, inapproachable. There is no getting underneath the proverbial hood to probe mechanism or engine. (Kirschenbaum 378–379)

Contra Kirschenbaum, there are indeed several possible probes that one can attach to neural networks to observe and instrument their operation. These are not only standard features of contemporary machine learning frameworks but some of these instruments are built directly into key data types (tensors) used to construct the networks, which is to say that features exist to record aspects of the history of the network within the network.

Research in model interpretability for deep learning networks is moving incredibly fast, but several key strategies have been in existence since Rosenblatt's initial investigation of Perceptrons. Guided backpropagation, network dissection, concept detectors, these are all mature tools in the toolbox for instrumenting and observing neural networks (Thampi). There is an expanding set of tools for probing models, especially the highly popular transformer networks. Humanist strategies, such as Jill Walker Rettberg's analysis of "algorithmic failure" provide insight into the opaquest of models (Rettberg). Various methods now exist that borrow from the neurosciences an interest in functional localization and have provided rough but usable analogs for single-cell recordings, sub-network activations, mapping and testing mechanisms, and other strategies for understanding the organization and operation of deep neural networks. Modeling has identified how architectures and architectural features appear to be differently specialized. We've learned, for example, that across application domains, for both images and text, higher layers are specialized for context-specific representations. The advent of adversarial techniques has led to the modification of networks to test theories about the storage of information in the models. New ways of reading and interpreting neural networks as cultural artifacts are being developed that attend to the formal dimensions of what Fabian Offert terms "a new paradigm of postsymbolic computation" (Offert 425). This area has been especially interesting to watch and there is an expanding array of techniques to probe and interpret text-based deep learning models (Dobson, "On Reading and Interpreting Black Box Deep Neural Networks"). Several researchers modifying networks argue that the networks have some sort of memory and that one can distinguish facts from other kinds of stored knowledge. These researchers have also discovered the exceptionally brittle nature of the information held within the model and major issues connected to reliability and trustworthiness in the models through their demonstration of editing large language models to alter these stored memories, these stored "facts" (Meng et al.).

Conclusion

I am not particularly invested in the opposition between formalism and historicism; I want to historicize computational methods, but as a generation of literary and cultural critics have demonstrated, historicization does not necessarily involve a rejection of formal concerns. Addressing neural networks in the abstract is to assume that they share the same form, the same architecture. Doing so ignores the fact that these networks are containers as well as functions. They produce a series of transformations; transformations that are increasingly constructed from the outputs of other transformations within the network. The form, which is to say, the architecture of such networks is important for understanding its outputs. The question of the historicity of neural networks and their outputs is key to my argument because this question cannot begin to be answered without formal analysis of network architecture.

When discussing complex technical objects, it is all too easy to ignore their forms. It is also the case that these forms are quite historically situated, far more so than the objects that attract attention from most humanists. As digital objects, neural networks are instantiated or concretized—as Yuk

Hui, who builds on Gilbert Simondon's earlier, non-digital conception of technological concretization, might put it, in a particular historical, cultural, and technological milieu (Hui). The architecture, the network graph, performs similar ontological work as descriptive metadata in Hui's reading of digital data objects. Orchestrated from already existing components and called into being without data, neural networks are digital objects marked by their genesis. The imaginary networks of Pitts and McCulloch, Hebb, Hubel and Wiesel, or the multiple actually implemented mechanical and simulated networks of Frank Rosenblatt have undergone some important transformations over time: they have been implemented in different programming languages, using different frameworks and paradigms, on different hardware and in relation to different inputs and outputs. Newer architectures are continuing to be developed as well as novel assemblages produced from existing models. Multimodal models, switch transformers, mixture of experts: these and other novel paradigms will complicate our existing understanding of the temporality of deep learning and how meaning is made through the process of inputs passing through the architecture of neural networks.

The stakes of these differences might best be seen in a critical account of recent machine learning architectures. Leif Weatherby and Brian Justie make much of the distinction between the mappable feature space and unequivocal representations found in earlier networks and the learned patterns of those produced since the 1990s. Weatherby and Justie take up several recent architectures including convolutional neural networks (CNNs) as used in computer vision and transformers found in language models as examples of what they call "indexical AI." "Indexical AI," they argue, "contrasts with the symbolic AI that dominated artificial intelligence research before 2000" (Weatherby and Justie 382). If symbolic AI names the category of shallower networks in which features (pixels, tokens, etc) could be more readily identified as representing aspects of the modeled object, the deep networks Weatherby and Justie term indexical AI leaves the would-be-interpreter with only non-representational pointers and pathways. These pointers and pathways work within the space defined by the pre-existing network, which they define as a "complex function with a concrete shape" (Weatherby and Justie 393). Given this, Weatherby and Justie propose as the only strategy available for reading the operation of contemporary networks, a concrete analysis that follows the pointers of "complex architecture of indexical pathways" (Weatherby and Justie 384). They propose, in short, formal analysis of the work of the network within the form instantiated by the network. There are other formal strategies for interpreting the large number of units and components found within contemporary deep learning networks, as well as a set of post-hoc methods for analyzing predictions and decisions made by these networks (Dobson, "On Reading and Interpreting Black Box Deep Neural Networks").

In this essay I have attempted to demonstrate, contrary to persistent claims otherwise, that neural networks have a historicity beyond their material origins and that their operations and their outputs do have at least the possibility of interpretability. The degree to which they are interpretable and important aspects of their historicity is a function of their form. That form is primarily registered in their architecture. Thus, any discussion of the form of a neural network and its output needs to take into account its particular architecture. Once the architectural form of a network is defined, from the relatively simple two or three-layer Perceptron to a multi-layered convolutional neural network to a transformer with attention heads, the historicity of that architecture becomes much more obvious. Faced with a model, one might ask why was this particular architecture selected? What problem does it solve? What were the alternatives? The biggest challenges for would-be-interpreters of neural networks are not found in their opacity or the degree to which their transactions might be inapproachable, but are the choices made by model developers to obscure their products and the details of their construction. The GPT-4 Technical Report, published on March 15, 2023, includes the following disclaimers "[GPT-4] it is not fully reliable (e.g. can suffer from "hallucinations"), has a limited context window, and does not learn from experience.... care should be taken when using the outputs of GPT-4, particularly in contexts where reliability is important" (OpenAI, et al. 1–2). OpenAI released this model without any description of its training sources or details about its architecture despite these claims. These limitations have also not prevented OpenAI from working partners, supporting and enabling all sorts of plugins, and signing up numerous API users. The "Limitations" section of this same report begins with the following claim:

> Despite its capabilities, GPT-4 has similar limitations as earlier GPT models. Most importantly, it still is not fully reliable (it "hallucinates" facts and makes reasoning errors). Great care should be taken when using language model outputs, particularly in high-stakes contexts, with the exact protocol (such as human review, grounding with additional context, or avoiding high-stakes uses altogether) matching the needs of specific applications. (OpenAI, et al. 10)

These limitations and a host of other concerns—including, but not limited to black-boxed or restricted access to the model, uncertainty about training sources, the cost of token inference, toxicity in the model, environmental concerns connected to training, etc—make OpenAI's models inappropriate for academic use. The stack of computational work in any scholarly field should contain open models and highly interpretable methods as well as critical inquiry into methods. The opacity of closed and proprietary models like GPT-4 combined with a disinterest in critique and the historicity of computation make for a problematic and dangerous environment. While the different kinds of opacity (i.e., intentional, technical illiteracy, intrinsic to complexity) found in machine learning, as Jenna Burrell argues, prompt different interpretive strategies (Burrell 1–2), multiple forms of opacity can easily frustrate attempts to understand the pipeline and its outputs. The combination of intentional and complex opacity has also made possible the amplification of unsubstantiated claims about these models—from fantasies of artificial general intelligence to sentience—and much of the mystification surrounding them in the present. Computational literary studies, and indeed any efforts toward the study of language models that call themselves humanistic, need to remain attentive to the forms of computation and the technological and social milieu in which digital objects are created and manipulated.

Acknowledgments

I would like to thank the members of the Computational Formalism workshop, sponsored by the Neukom Institute at Dartmouth College, for their comments and feedback on this work, and extend special thanks to the workshop organizers, Tess McNulty and Laura Chapot.

Submitted: August 02, 2024 EDT. Accepted: December 03, 2024 EDT. Published: September 10, 2025 EDT.

WORKS CITED

Abadi, Martín, et al. "Tensorflow: A System for Large-Scale Machine Learning." *12th USENIX Symposium on Operating Systems Design and Implementation*, 2016, pp. 265–83.

Brown, Tom B., et al. "Language Models Are Few-Shot Learners." *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Process*, 2020.

Burrell, Jenna. "How the Machine 'Thinks': Understanding Opacity in Machine Learning Algorithms." *Big Data & Society*, vol. 3, no. 1, 2016, pp. 1–12, https://doi.org/10.1177/2053951715622512.

Crawford, Kate. *Atlas of AI: Power, Politics, and the Planetary Costs of Artificial Intelligence*. Yale University Press, 2022, https://doi.org/10.12987/9780300252392.

Dhaliwal, Ranjodh Singh, et al. *Neural Networks*. University of Minnesota Press, 2024.

Dobson, James E. "Objective Vision: Confusing the Subject of Computer Vision." *Social Text*, vol. 41, no. 3, 2023, pp. 35–55, https://doi.org/10.1215/01642472-10613653.

---. "On Reading and Interpreting Black Box Deep Neural Networks." *International Journal of Digital Humanities*, vol. 5, 2023, pp. 431–49, https://doi.org/10.1007/s42803-023-00075-w.

---. *The Birth of Computer Vision*. University of Minnesota Press, 2023.

---. "Vector Hermeneutics: On the Interpretation of Vector Space Models of Text." *Digital Scholarship in the Humanities*, vol. 37, no. 1, 2022, pp. 81–93, https://doi.org/10.1093/llc/fqab079.

Elam, Michele. "Poetry Will Not Optimize; or, What Is Literature to AI?" *American Literature*, vol. 95, no. 2, June 2023, pp. 281–303, https://doi.org/10.1215/00029831-10575077.

Eve, Martin Paul. *Close Reading with Computers: Textual Scholarship, Computational Formalism, and David Mitchell's Cloud Atlas*. Stanford University Press, 2019, https://doi.org/10.21627/9781503609372.

Frye, Northrop. "Literary and Mechanical Models." *Research in Humanities Computing 1: Selected Papers from the ALLC/ACH Conference, Toronto, June 1989*, edited by Ian Lancashire, Oxford University Press, 1991.

Gurnee, Wes, et al. "Finding Neurons in a Haystack: Case Studies with Sparse Probing." *arXiv:2305.01610, arXiv*, 2 June 2023. *arXiv.org*, http://arxiv.org/abs/2305.01610.

Halpern, Orit. *Beautiful Data: A History of Vision and Reason since 1945*. Duke University Press, 2014, https://doi.org/10.1215/9780822376323.

Hamilton, William L., et al. "Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change." *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, 2016, pp. 1489–501, https://doi.org/10.18653/v1/P16-1141.

Hochreiter, Sepp, and Jürgen Schmidhuber. "Long Short-Term Memory." *Neural Computation*, vol. 9, 1997, pp. 1735–80, https://doi.org/10.1162/neco.1997.9.8.1735.

Hui, Yuk. *On the Existence of Digital Objects*. University of Minnesota Press, 2016, https://doi.org/10.5749/minnesota/9780816698905.001.0001.

Kirschenbaum, Matthew. "Spec Acts: Reading Form in Recurrent Neural Networks." *ELH*, vol. 88, no. 2, 2021, pp. 361–86, https://doi.org/10.1353/elh.2021.0010.

Lewis, Patrick, et al. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks." *Proceedings of the 34th International Conference on Neural Information Processing Systems*, vol. 33, 2020, pp. 9459–74.

Lindgren, Simon. *Critical Theory of AI*. Polity Press, 2024.

Meng, Kevin, et al. “Locating and Editing Factual Associations in GPT.” *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 17359–72.

Min, S. Sewon, et al. “Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?” *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022, pp. 11048–64, https://doi.org/10.18653/v1/2022.emnlp-main.759.

Offert, Fabian. “Can We Read Neural Networks? Epistemic Implications of Two Historical Computer Science Papers.” *American Literature*, vol. 95, no. 2, June 2023, pp. 423–28, https://doi.org/10.1215/00029831-10575218.

Offert, Fabian, and Ranjodh Singh Dhaliwal. “The Method of Critical AI Studies, A Propaedeutic.” *arXiv*, 28 Nov. 2024, https://doi.org/10.48550/arXiv.2411.18833.

OpenAI, et al. “GPT-4 Technical Report.” *arXiv:2303.08774, arXiv*, 18 Dec. 2023. *arXiv.org*, http://arxiv.org/abs/2303.08774.

Ouyang, Long, et al. “Training Language Models to Follow Instructions with Human Feedback.” *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 27730–44.

Pasquinelli, Matteo. *The Eye of the Master: A Social History of Artificial Intelligence*. Verso, 2023.

Paszke, Adam, et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library.” *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 8024–35.

Rettberg, Jill Walker. “Algorithmic Failure as a Humanities Methodology: Machine Learning’s Mispredictions Identify Rich Cases for Qualitative Analysis.” *Big Data & Society*, vol. 9, no. 2, July 2022, pp. 1–6, https://doi.org/10.1177/20539517221131290.

Rogers, Anna, et al. “A Primer in BERTology: What We Know About How BERT Works.” *Transactions of the Association for Computational Linguistics*, vol. 8, Dec. 2020, pp. 842–66, https://doi.org/10.1162/tacl_a_00349.

Rosenblatt, Frank. *Design of an Intelligent Automaton*. Research Reviews, Office of Naval Research, Oct. 1958, pp. 5–13.

---. *The Perceptron: A Perceiving and Recognizing Automaton (Project PARA)*. Report, 85-460–1, Cornell Aeronautical Laboratory, 1957.

Rosenblatt, Frank, and Carl Keseler. “Further Simulation Experiments on Series-Coupled Perceptrons.” *Collected Technical Papers: Volume 2. Report 5*, edited by Frank Rosenblatt, Cornell University, 1963, pp. 73–98.

Sakarvadia, Mansi, et al. “Attention Lens: A Tool for Mechanistically Interpreting the Attention Head Information Retrieval Mechanism.” *arXiv:2310.16270, arXiv*, 24 Oct. 2023. *arXiv.org*, http://arxiv.org/abs/2310.16270.

Saphra, Naomi, and Sarah Wiegreffe. “Mechanistic?” *arXiv*, 7 Oct. 2024, https://doi.org/10.48550/arXiv.2410.09087.

Shmueli, Galit. “To Explain or to Predict?” *Statistical Science*, vol. 25, no. 3, Aug. 2010, https://doi.org/10.1214/10-STS330.

Szegedy, Christian, et al. “Going Deeper with Convolutions.” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2015, pp. 1–9, https://doi.org/10.1109/CVPR.2015.7298594.

Thamp, Ajay. *Interpretable AI: Building Explainable Machine Learning Systems*. Manning Publications, 2022.

Vaswani, Ashish. “Attention Is All You Need.” *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 5998–6008.

Wasielewski, Amanda. *Computational Formalism: Art History and Machine Learning*. MIT Press, 2023, https://doi.org/10.7551/mitpress/14268.001.0001.

Weatherby, Leif, and Brian Justie. “Indexical AI.” *Critical Inquiry*, vol. 48, no. 2, 2022, pp. 381–415, https://doi.org/10.1086/717312.

Wharton, Annabel Jane. “Defining Models.” *Modelwork: Material Culture of Making and Knowing*, edited by Martin Brückner et al., University of Minnesota Press, 2021, pp. 1–17, https://doi.org/10.5749/j.ctv1z9n20d.4.