

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Perplexity AI, Inc.
Petitioner

v.

Comet ML, Inc.
Patent Owner

Case No. IPR2025-01063

U.S. Patent No. 11,650,968

**DECLARATION OF SCOTT DENNING
IN SUPPORT OF PETITION FOR INTER PARTES REVIEW OF
UNITED STATES PATENT 11,650,968**

CLAIM LISTING

Claim Element	Claim Limitation
1[pre].	A method of training a neural network (NN), the method comprising:
1[a].	over a series of NN training epochs, where in each epoch the NN undergoes training and a loss is computed:
1[b].	determining, using:
1[b.1].	a set of model parameters;
1[b.2].	data describing training loss of the NN; and
1[b.3].	a model that has been trained using training losses of a plurality of NNs other than the NN;
1[c].	a probability of improvement in the loss of the NN; and
1[d].	if the probability is less than a threshold, or a wait value is greater than a wait threshold, stopping training.
2.	The method of claim 1, comprising if the probability is not less than a threshold and a wait value is greater than a wait threshold, continuing training.
3.	The method of claim 1, comprising increasing the wait value if the current loss of the NN is not less than the minimum loss in the loss history for the NN.
4.	The method of claim 1, comprising setting the wait value to zero if the current loss of the NN is less than or equal to the minimum loss in the loss history for the NN.
5.	The method of claim 1, wherein determining using a model a probability of improvement in the loss of the NN comprises obtaining from a leaf of at least one tree data structure data relevant to the NN.
6.	The method of claim 1, wherein determining an expected probability of improvement comprises determining a mean expected training loss and a variance.
7.	The method of claim 1, wherein model parameters comprise hyperparameters.
8[pre].	A system of training a neural network (NN), the system comprising:
8[a].	a memory; and
8[b].	a processor configured to:
8[c].	over a series of NN training epochs, where in each epoch the NN undergoes training and a loss is computed:

Claim Element	Claim Limitation
8[d].	determine, using:
8[d.1].	a set of model parameters;
8[d.2].	data describing training loss of the NN; and
8[d.3].	a model that has been trained using training losses of a plurality of NNs other than the NN;
8[d.4].	a probability of improvement in the loss of the NN; and
8[e].	if the probability is less than a threshold, or a wait value is greater than a wait threshold, determine to stop training.
9.	The system of claim 8, wherein the processor is configured to, if the probability is not less than a threshold and a wait value is greater than a wait threshold, determine to continue training.
10.	The system of claim 8, wherein the processor is configured to increase the wait value if the current loss of the NN is not less than the minimum loss in the loss history for the NN.
11.	The system of claim 8, wherein the processor is configured to set the wait value to zero if the current loss of the NN is less than or equal to the minimum loss in the loss history for the NN.
12.	The system of claim 8, wherein determining using a model a probability of improvement in the loss of the NN comprises obtaining from a leaf of at least one tree data structure data relevant to the NN.

I, Scott Denning, declare that:

I. Introduction

1. I have been retained by Petitioner Perplexity AI, Inc. (“Petitioner”), as an independent expert in this *Inter Partes Review* proceeding before the United States Patent and Trademark Office.

2. I understand that this proceeding involves U.S. Patent No. 11,650,968 (“the ’968 patent”), and I have been asked to provide my opinions as to the patentability of claims 1-12 of the ’968 patent based on certain prior-art references. I have been asked to consider the state of the art and prior art available as of May 24, 2019. Based on my experience and expertise, it is my opinion that the prior art renders obvious claims 1-12 of the ’968 patent, as I discuss in detail below.

3. I understand that the ’968 patent is assigned to Comet ML, Inc. (which I may refer to herein as “Patent Owner” or “PO”).

4. I am not, and never have been, an employee of Comet ML, Inc. or Perplexity AI, Inc. I am not receiving compensation for this Declaration beyond my normal hourly fees based on my time actually spent analyzing and documenting my opinions herein on the ’968 patent, the asserted prior art publications cited in this Declaration and in the Petition, and the issues related thereto. My compensation is not related to the outcome of this proceeding, and I will not receive any additional compensation based on the outcome of any IPR or other proceeding involving the

'968 patent.

5. All of my opinions set forth below are based on my own personal knowledge and professional judgment. I am over 18 years of age and, if I am called upon to do so, I would be competent to testify under oath as to the matters set forth in this Declaration.

II. Background & Qualifications

6. All of my opinions stated in this Declaration are based on my own personal knowledge and professional judgment. I believe I am well qualified to serve as a technical expert in this matter based on my educational and work experience in the field of training artificial neural networks. In forming my opinions, I have relied on my training, knowledge, and experience in the relevant art. My *curriculum vitae* (CV) is submitted as Exhibit 1004. I certify that Ex. 1004 is a true and correct copy and sets forth the details of my educational history, work history, and other related professional experience.

7. I am an expert in the field of training artificial neural networks, and my expertise stems from my educational background and years of work experience in this field. I received a Bachelor of Science degree in Computer Science from Southern College of Technology in 1993. I have a Master of Science in Economic Analytics from Clemson University and, with the exception of defending my dissertation, I have completed all requirements for my Ph.D. degree from University

of Colorado with a focus on Machine Learning and Natural Language Processing.

8. I have 30 years of experience working with server software, embedded software, communication protocols, mobile devices, computer networks, and control systems. I also have 9 years of experience building models with artificial neural network (ANN) software libraries. As detailed below, I have contributed architectural design, systems engineering, and many of lines of functional program code for building, training, and deploying ANNs.

9. Much of the research for my MS in Economic Analytics from Clemson between August, 2022 and May, 2024 involved the design of an ANN model using Google's TensorFlow for predicting crop yields based on daily weather data. TensorFlow is a software library for machine learning and artificial intelligence, that relies upon defining the ANN architecture and numerous hyperparameters for training models as referenced by the '968 patent. The ANN model from my thesis has a root mean square error (RMSE) of less than 6% and remains state-of-the-art in crop yield prediction to date. Creating this model required nearly 2 years of research and hundreds of iterations testing various architectures and hyperparameters for optimal model fit.

10. From 2016 to the present, I have worked as a machine learning researcher, a Research Scientist, and Chief Scientist of Deep Learning. In these roles, I have personally created many highly accurate ANN models and led teams that

created the same. This work is highly iterative. I have started, stopped, and aborted many thousands of ANN model builds in order to perform architectural modifications and/or changes to the hyperparameters in pursuit of improvement of either build time or model accuracy.

11. While performing research for a PhD at the University of Colorado between 2016 and 2018, I used PyTorch, TensorFlow, and Python-implemented, scratch built ANNs to create many concept maps of documents (chains of concepts that are independent of the exact words used), Bayesian language models (given a series of words, identify the word that most likely comes next), and word embeddings (similarity between words).

12. From 2018 to 2021 I was Vice President and Chief Scientist of Deep Learning at Ontrak, Inc., a technology-driven behavioral healthcare company providing patient care using digital patient care records. During this time, I led a team of 5 research scientists in creating products and mission-critical applications for identifying and caring for individuals having a high probability of having behavioral health problems including anxiety, depression and substance abuse though these conditions had never been diagnosed. The team also created systems for matching prospective patients with outreach agents and care coaches resulting in a 23% improvement in sign-up and 30% improvement in 1 year retention. These systems rely on models created from personas, publicly available information, and

health records comprising diagnosis and treatment codes, prescriptions, and doctor's notes. And they were created by TensorFlow and other ANN frameworks and libraries.

13. A complete list of my publications and testifying experience is included in my *curriculum vitae*, attached as Exhibit 1004.

III. Materials Considered

14. The opinions contained in this Declaration are based on the documents I reviewed, my professional judgment, as well as my education, knowledge and experience in the relevant art.

15. In forming my opinions, I reviewed the following documents:

Exhibit	Description
Ex. 1001	U.S. Patent No. 11,650,968 (the "'968 patent")
Ex. 1002	Prosecution History for U.S. Patent No. 11,650,968
Ex. 1005	International Publication No. WO 2018/175098 ("Baker")
Ex. 1006	<i>A Comprehensive guide to Bayesian Convolutional Neural Network with Variational Inference</i> ("Shridhar")
Ex. 1007	<i>Bayesian Optimization with Tree-structured Dependencies</i> ("Jenatton")
Ex. 1008	<i>Stopping criteria for boosting automatic experimental design using real-time fMRI with Bayesian optimization</i> ("Lorenz")
Ex. 1009	<i>Speeding up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves</i> ("Domhan")
Ex. 1010	U.S. Pub. No. 2019/0095798 ("Baker '798 Pub.")

Exhibit	Description
Ex. 1011	Declaration of Dr. Sylvia Hall-Ellis

16. In forming my opinions, I have considered the materials listed above and any other documents cited in this Declaration. My opinions are based on the documents I reviewed and my knowledge and professional judgment. My opinions have also been guided by my appreciation of how a person of ordinary skill in the art would have understood the state-of-the-art, the prior art, and the claims and the specification of the '968 patent at the time of the alleged invention.

IV. Legal Standards

17. I am not an attorney, and do not offer opinions on the law in this Declaration.

18. In forming my opinions and considering the subject matter of the '968 patent and its claims in light of the prior art, I am relying on certain legal principles that counsel in this case explained to me. My understanding of these concepts is summarized below.

19. I understand that, in this proceeding, Petitioner has the burden of proving that the challenged claims are invalid by a preponderance of the evidence.

A. Anticipation

20. I have been informed that a claim is invalid as anticipated under 35 U.S.C. § 102 (AIA) if a prior art reference discloses, either expressly or inherently,

all limitations of a claim as those limitations are arranged in the claim.

21. I have been informed that a limitation is disclosed inherently in a reference if it is necessarily present in the process or product described in the prior art reference. I understand that probabilities or possibilities are insufficient to show that a prior art reference inherently discloses something beyond what it discloses explicitly.

B. Obviousness

22. I have been informed that a claim is invalid as obvious under 35 U.S.C. § 103 (AIA) if the differences between the claimed subject matter and the prior art are such that the subject matter as a whole would have been obvious at the time of the invention to a person of ordinary skill in the art. I have been informed that the following matters are relevant to determining whether the claimed invention would have been obvious: (1) the scope and content of the prior art, (2) the difference or differences between the patent claim and the prior art, (3) the level of ordinary skill in the art at the time the invention of the patent, and (4) any secondary considerations or objective indicia of non-obviousness.

23. I have been informed that the combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results. When a claim simply arranges prior art elements with each performing the same function it had been known to perform and yields no more than

one would expect from such an arrangement, then such a combination is obvious. When a patent claims a structure already known in the prior art altered by the mere substitution of one element for another known in the field, the combination is likely to be obvious unless the combination yields an unpredictable result.

24. I have been informed that when a work is available in one field of endeavor, design incentives and other market forces can prompt variations of it, either in the same field or a different one. If a person of ordinary skill in the art can implement a predictable variation, such a variation is likely unpatentable. For the same reason, if a technique has been used to improve one device, and one of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill. One question to consider is whether the improvement is more than predictably using prior art elements according to their established functions.

25. I have been informed that it may often be necessary, in a validity analysis, to consider whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue. This can be accomplished by looking to interrelated teachings of multiple patents or other publications or pieces of prior art; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by one of ordinary skill in the art.

26. I have been informed that a validity analysis need not seek out precise teachings directed to the specific subject matter of the challenged claim; it is appropriate to take account of the inferences and creative steps that a person of ordinary skill in the art would employ. I have been informed that a person of ordinary skill in the art is a person of ordinary creativity, not an automaton.

27. I have been informed that a claim composed of several elements is not proved obvious merely by demonstrating that each element was, independently, known in the prior art. I have been informed that it can be important to identify a reason that would have prompted a person of ordinary skill in the art in the relevant field to combine the elements in the way the claimed new invention does. I am told that one way that subject matter can be proved obvious is by noting there existed at the time of the invention a known problem for which there was an obvious solution encompassed by the patent's claims. I have been informed that any need or problem known in the field of endeavor at the time of the claimed invention and addressed by the patent can provide a reason for combining the elements in the manner claimed.

28. I have been informed that one should not assume that a person of ordinary skill in the art attempting to solve a problem will be led only to those elements of prior art designed to solve the same problem. Instead, I have been informed that since familiar items may have obvious uses beyond their primary purposes, in many cases a person of ordinary skill in the art will be able to fit the

teachings of multiple prior art references together like pieces of a puzzle.

29. I have been informed that, when there is a design need or market pressure to solve a problem and there are a finite number of identified, predictable solutions, persons of ordinary skill in the art have good reason to pursue the known options within their technical grasp. If this leads to the anticipated success, the product was likely not accomplished by innovation but by using ordinary skill and common sense. I have been informed that, in such an instance, the fact that the combination was obvious to try may show that the combination was obvious.

30. I have been informed that, when determining whether a claimed combination would have been obvious, the correct analysis is not whether a person of ordinary skill in the art, writing on a blank slate, would have chosen the particular combination of elements described in the claim. Instead, I have been informed that the correct analysis considers whether one of ordinary skill, facing the wide range of needs created by developments in the field of endeavor, would have seen a benefit to selecting the combination claimed.

31. I have been informed that, when determining whether a claimed invention is obvious, any “secondary considerations” of non-obviousness identified by the patentee should also be considered. These secondary considerations can include:

- commercial success of the invention, causally related to the invention itself rather than to companion factors, such as advertising or attractive packaging;
- the invention taught away from the technical direction followed by those skilled in the art;
- a long-felt but unsatisfied need for the invention while the needed implementing arts and elements had long been available;
- the invention achieves results unexpected to those skilled in the art;
- copying of the invention by competitors as distinguished from their independent development;
- unsuccessful attempts by those skilled in the art to make the invention;
- acquiescence by the industry to the patent's validity by honoring the patent through taking licenses or not infringing the patent, or both; and
- skepticism, disbelief in or incredulity by those skilled in the art that the patentee's approach worked.

32. I have been informed that, for the above information to impact the obviousness of a patent claim, there must be a nexus between the alleged secondary considerations and the claims. In addition, I have been informed that the burden of introducing evidence of secondary considerations generally is on the Patent Owner. If the Patent Owner or its expert should assert secondary considerations of non-obviousness, I reserve the right to provide a Declaration addressing assertions of

non-obviousness due to secondary considerations.

C. Claim Construction

33. I understand that the United States Patent and Trademark Office interprets claim terms in an *inter partes* review proceeding under the same claim construction standard that is used in a United States federal court. I understand that under this standard, the meaning of claim terms is considered from the viewpoint of one of ordinary skill in the art at the time of the earliest alleged priority date.

34. I understand that claim terms are generally given their plain and ordinary meaning as understood by one of ordinary skill in the art in light of the specification and the prosecution history pertaining to the patent. I understand, however, that claim terms are generally not limited by the embodiments described in the specification.

35. I understand that in addition to the claims, specification, and prosecution history, other evidence may be considered to ascertain the meaning of claim terms, including textbooks, encyclopedias, articles, and dictionaries. I have been informed that this other evidence is often less significant and less reliable than the claims, specification, and prosecution history.

36. I have been informed that, in general, a preamble limits the invention if it recites essential structure or steps, or if it is necessary to give life, meaning, and vitality to the claim. I have further been informed that a preamble is not limiting

where a patentee defines a structurally complete invention in the claim body and uses the preamble only to state a purpose or intended use for the invention. I have further been informed that dependence on a particular disputed preamble phrase for antecedent basis may limit claim scope because it indicates a reliance on both the preamble and claim body to define the claimed invention. I have further been informed that clear reliance on the preamble during prosecution to distinguish the claimed invention from the prior art transforms the preamble into a claim limitation because such reliance indicates use of the preamble to define, in part, the claimed invention.

D. Priority Under 35 U.S.C. § 120

37. I understand that the “priority date” or “effective filing date” of a patent is the date on which it is filed, or the date on which an earlier-filed U.S. or international patent application was filed if the patentee claims the benefit of priority to that earlier-filed application.

38. I understand that in the patent application process, the applicant may keep the originally filed claims or change the claims between the time the patent application is first filed and the time a patent is issued. An applicant may amend the claims or add new claims. These changes may narrow or broaden the scope of the claims. I understand that a patent application is entitled to the benefit of the filing date of an earlier-filed application only if the disclosure of the earlier application

provides support for the claims of the later application as required by the “written description” requirement of 35 U.S.C. § 112. I further understand that for a patent application to have an adequate written description of the invention, it must show the inventor was in possession of the invention at the time the patent application was filed.

39. When determining whether a specification of a patent application contains adequate written description of the claimed invention, I understand that one must make an objective inquiry into the four corners of the specification from the perspective of a POSITA. Further, I understand that the standard for written description is stricter than the standard for determining whether or not a patent claim would have been obvious based on the prior art.

40. In particular, the question for purposes of written description is not merely whether the claimed invention is an obvious variant of what the specification discloses. Instead, the application itself must describe the invention, and do so in sufficient detail that a POSITA can clearly conclude that the inventor invented the claimed invention as of the date when the application was filed.

41. Similarly, I understand the issue of written description is separate from the question of “enablement” (i.e., whether a specification allows a POSITA to practice the claimed invention without undue experimentation). A specification may adequately enable a claimed invention yet fail to provide written description support.

42. Finally, I understand that the standard for whether or not a disclosure is sufficient to provide written description support for a claim is different from the standard for whether a disclosure anticipates a claim. To support a claim, the description must be sufficient to convey to a POSITA that the inventors were in possession of the full scope of the claimed subject matter. By contrast, a description anticipates a claim if it discloses even a single species within the scope of the claim. Thus, in the case of “genus” claims covering two or more different species, I understand that there can be situations in which a given disclosure is sufficient to anticipate the claim even though that same disclosure is not adequate to support the claim and entitle the inventors to an earlier priority date.

43. I understand that in order to claim priority to an application through a chain of applications, every single application in the chain (going back to the particular application on which the inventors want to rely) needs to support the claim as a matter of written description. In other words, each application in the chain must allow a person of ordinary skill in the art to discern that the inventors were in possession of the claimed invention at the time of the application. It is not sufficient under this standard for the disclosure in the specification to render the claimed invention obvious.

V. Person of Ordinary Skill in the Art at the Time of the Alleged Invention

44. I have been asked to assume that the '968 patent is entitled to its earliest alleged priority date of May 24, 2019. *see* Ex. 1001, 1 (“Related U.S. Application Data” section). Based on my knowledge and experience, I was a person of ordinary skill in the art (“POSITA”) as of May 24, 2019, and can provide opinions regarding the knowledge of a person of ordinary skill in the art as of that time. My opinions herein are, where appropriate, based on my understandings as to a person of ordinary skill in the art at that time.

45. I understand that factors that are pertinent to a determination of the level of ordinary skill in the art include: (1) the educational level of the inventor; (2) the type of problems encountered in the art; (3) the prior art solutions to those problems; (4) the rapidity with which innovations are made; (5) the sophistication of the technology; and (6) and the educational level of workers active in the field.

46. Based on the materials and information I have reviewed and based on my experience in the technical areas relevant to the '968 patent, a person of ordinary skill in the art at the time of the alleged invention of the '968 patent would have had a Bachelor of Science degree in Computer Science with two years of related work experience building and training ANNs, including selecting and testing the accuracy of ANNs trained with various hyperparameter combinations. Alternatively, that person could have also had a Master of Science degree in computer science with less

than a year of related work experience in the same discipline. More education can supplement practical experience and *vice versa*.

47. I exceeded this level of skill in the art as of the earliest priority date of the '968 patent.

VI. Technical Background

48. Below, I present a brief overview of certain technical concepts and features that would have been known to a person of ordinary skill in the art prior to and at the time of the earliest alleged priority date of the '968 patent. The technical concepts and features that I describe in this Technical Background section reflect the state of the art that a person of ordinary skill in the art would have had knowledge of and would have understood prior to and at the time of the alleged invention of the '968 patent. I rely on, and incorporate as applicable (even if not expressly mentioned below in Section IX), the following disclosures to support my opinions in this Declaration, including my opinions relating to how the prior art discloses and/or suggests the challenged claims of the '968 patent and how (and why) a person of ordinary skill in the art would have been motivated to consider and combine the disclosures and suggestions from one or more prior art references.

A. Neural Networks and Iterative Training over Mini-Batches and Epochs

49. Prior to the '968 patent's earliest alleged priority date, a person of ordinary skill in the art would have been familiar with neural networks and how the

terms mini-batches (or minibatches) and epochs describe aspects of training neural networks.

50. Neural networks are one example of machine learning models. During a learning process of a neural network, training data consisting of example data is provided to the neural network so that the neural network can self-teach a prediction algorithm. Once trained, the neural network can generate predictions based on a provided input. As an example, a neural network may be trained to classify images based on whether the images show a cat or not. The training data includes labeled images – some images including cats (and labeled as such) and other images that do not include cats (again, with appropriate labels).

51. The Baker '798 Pub (Ex. 1010) demonstrates that, as early as September 28, 2017, a person of ordinary skill in the art would have known the structure of a neural network as “a collection of nodes and directed arcs.” Ex. 1010, ¶[0014]. The Baker '798 Pub further explains that the “nodes in a neural network are often organized into layers,” and, “[i]n a feed-forward neural network, the layers may be numbered from bottom to top.” *Id.* “Each directed arc in a layered feed-forward neural network goes from a source node in a lower layer to a destination node in a higher layer.” *Id.* The Baker '798 Pub depicts a “feed-forward neural network” having “an input layer, an output layer, and three inner layers.” *Id.* An “inner layer in a neural network is also called a ‘hidden’ layer,” and “[e]ach directed

arc is associated with a numerical value called its ‘weight.’” *Id.* The Baker ’798 Pub explains that, “[t]ypically, each node other than an input node is associated with a numerical value called its ‘bias,’” and the “weights and biases of a neural network are called ‘learned’ parameters.” According to the Baker ’798 Pub, “[d]uring training, the values of the learned parameters are adjusted by a computer system,” and other “parameters that control the training process are called hyperparameters.” *Id.* One example of a learning process involves backpropagation. Input data is presented to the NN, which produces an output (during a forward pass), and then the NN is presented with the correct output and generates an error (e.g., a “loss”) and updates the weights assigned to the neurons during backpropagation. The goal of training is to produce a neural network that is highly accurate in generating predictions. Thus, training should continue until the model converges to some best performance level. Often, this is measured in terms of a “loss” or “cost of errors” as I explain in more detail below with reference to the ’968 patent. However, continuing to provide more training data (or continuing to iterate though a set of training data) after the model converges to a best performance does not provide further benefits to the model. In fact, the model will eventually exhibit “overfitting” behavior, in which the model begins to perform best on classifying the training data, but cannot provide predictions for input that is included in the training data set.

52. It is possible to manipulate the training process by changing

characteristics of the training process and/or by changing the structure of the neural network. Changing these settings, known as “hyperparameters,” can impact the possible performance of the neural network. A few examples of these hyperparameters include the number and structure of layers, number of neurons in each layer, and/or the type of functions used by each neuron when calculating an output.

53. The Baker ’798 Pub also explains one example of training, in which a “feed-forward neural network may be trained by the computer system 400 using an iterative process of stochastic gradient descent with one iterative update of the learned parameters for each minibatch.” Ex. 1010, ¶[0014]. The Baker ’798 Pub explains that a “full batch of training data is typically arranged into a set of smaller, disjoint sets called *minibatches*,” and an “*epoch* comprises the computer system 400 doing a stochastic gradient descent update for each *minibatch* contained in the full batch of training data.” Ex. 1010, ¶[0014] (emphasis added). Said differently, the process of performing a stochastic gradient descent on a minibatch constitutes completion of an epoch. Therefore, to the extent that any summary statistics (e.g., a loss calculation) or other information are generated after each minibatch is processed (including the last minibatch in an epoch), these summary statistics are calculated after each epoch, since processing of the minibatch is the same as completion of an epoch.

54. The Baker '798 Pub confirms what a person of ordinary skill in the art would have understood with respect to the use of the terms “epoch” and “minibatch,” including that during an epoch, “[f]or each minibatch, the computer system 400 estimates the gradient of the objective for a training data item by first computing the activation of each node in the network using a feed-forward activation computation.” *Id.* The “partial derivatives of the objective” are then estimated “with respect to the learned parameters using a process called ‘back-propagation,’ which computes the partial derivatives based on the chain rule of calculus, proceeding backwards through the layers of the network.” Ex. 1010, ¶[0014]. The Baker '798 Pub underscores that “[t]he processes of stochastic gradient descent, feed-forward computation, and back-propagation are well-known to those skilled in the art of training neural networks.” Ex. 1010, ¶[0014].

B. Hyperparameter Optimization

55. Prior to the '968 patent's earliest alleged priority date, a person of ordinary skill in the art would have been familiar with the process of hyperparameter optimization.

56. For example, Domhan (Ex. 1009, cited but not relied upon in prosecution), demonstrates that at least as early as July 2015 (*see* Ex. 1011, ¶¶45-48), a person of ordinary skill in the art would have been familiar with the computationally expensive nature of fitting large deep neural networks (DNNs) and

“the time overhead of automated hyperparameter optimization” that “hampers its widespread adoption.” Ex. 1009, 1 (“1 Introduction”). A person of ordinary skill in the art was well aware that DNNs were very sensitive to the setting of their hyperparameters. Ex. 1009, *Abstract*. These hyperparameters dictate aspects of the DNN training process and can cause the training process to produce model parameters/weights having a low error rate and/or cost of errors, if the selected hyperparameters are appropriate for the DNN and training data.

57. Domhan describes known methods to speed up hyperparameter selecting with “predictive models to speed up hyperparameter optimizers.” Ex. 1009, 4 (“3.2 Speeding up Hyperparameter Optimization”).

58. Domhan explains that “while the hyperparameter optimizer is running,” the best performance was tracked. Ex. 1009, 4 (“3.2 Speeding up Hyperparameter Optimization”). “Each time the optimizer queries the performance $l(\lambda)$ of a hyperparameter setting λ ” the DNN is trained using λ as usual except that the training run is terminated early if an “extrapolation model predicts the network to eventually yield worse performance” than the best performance. *Id.* More precisely, “at regular intervals i during SGD training we measure and save validation set performance y_i .” *Id.* According to Domhan, there “are $e_{\max}$ epochs, k such intervals per epoch, and every p epochs, we gather the performance values $y_{1:n}$ of the n intervals so far and run [Markov Chain Monte Carlo inference] MCMC to probabilistically extrapolate

performance to the final step $m = k \times e_{max}$.” *Id.* The predicted probability that the network, after training for m intervals, will exceed the best performance, is then considered. *Id.* If “this probability is above a threshold δ then training continues as usual for the next p epochs.” *Id.* “Otherwise, training is terminated.” *Id.*

VII. Summary of the ’968 Patent

A. The ’968 Patent’s Specification

59. The ’968 patent is titled “Systems and methods for Predictive Early Stopping in Neural Network Training.” Ex. 1001, Cover. The ’968 patent discloses “[s]ystems and methods” that “may train one or more neural networks (NNs) and determine when to stop training to not waste computing or other resources when improvement is not [sic] no longer likely.” Ex. 1001, 3:22-25.

60. In background, the ’968 patent describes a typical NN training process at a high level, noting that training a NN “is a time-consuming process, with a significant computational overhead.” Ex. 1001, 2:5-7. Training involved evaluating “different weights, and hyperparameter configurations, [and] involves waiting for the model’s loss function to converge to some local minima.” Ex. 1001, 2:7-9. The training process itself includes a forward and reverse pass of data through the NN. In the forward pass, “the NN is presented with an input and produces an output, and in the backward pass (backpropagation) the NN is presented with the correct output,

generates an error e.g., a loss, and generates update gradients which are used to alter the weights at the links or edges.” Ex. 1001, 2:10-19.

61. The '968 patent describes the well-known concept of “loss” (or a loss function) as defining “a deviation from an expected or correct output value for a given input, during learning or training. A loss function for a network may represent the difference or inconsistency between the value or values output from the network, and the correct value/values that should be output given the data input to the NN.” Ex. 1001, 2:46-51. Examples of loss functions include “a negative log-likelihood or residual sum of squares but may be computed in another manner.” Ex. 1001, 2:52-54.

62. The '968 patent further discusses how a training process may proceed in terms of batches and epochs.

Learning or training may be organized in batches and epochs. A batch may be a set of multiple training data presented to a NN which results in a number of outputs. The NN weights may be modified after an entire batch is run. A loss may be generated for each batch. An epoch may be a set of batches representing the entire training set; thus an epoch may represent a number of forward/backward passes, each resulting in a weight adjustment. There may be multiple losses per epoch, one for each batch.

Ex. 1001, 2:58-65.

63. In short, training may include running multiple epochs on batches of training data. Ex. 1001, 2:58-3:3. As described, “loss” values may be calculated for each batch. Ex. 1001, 2:58-65.

64. The ’968 patent suggests there are problems with “[p]rior training methods” for neural networks, in that they “typically stop by periodically testing a NN using a holdout data set not included in training data,” and it “is hard or impossible using prior art methods to predict at any given point in training how much improvement can be achieved by training using further epochs.” Ex. 1001, 3:4-18. The ’968 patent alleges that it “would be desirable to determine at what point further improvement beyond a certain amount is not likely, or not worth the extra time and resources for training.” *Id.*

65. The ’968 patent purports to solve these problems by using a secondary machine learning model to predict “when to stop training to not waste computing or other resources when improvement is not [sic] no longer likely.” Ex. 1001, 3:22-25. The secondary model can predict when to stop training during a single training run (e.g., for a predefined set of hyperparameters), when it is unlikely to achieve better performance than a previously established best. The ’968 patent describes this as operating in a “local mode.” *Id.*, 3:36-4:4. In the local mode, the method uses known stopping criteria to ascertain when to stop training: the method “may cause stopping

of training in a ‘local’ mode as soon as a probability of improvement in the loss of the NN is less than a threshold, or a wait value is greater than a wait threshold.” *Id.*, 18:63-66. The “local” mode can stop a training process for which hyperparameters have already been chosen as soon as it is determined that the training process has likely already found the minimum loss achievable by the training process. *Id.*, 5:7-18.

66. The ’968 patent also states that the secondary model can assist in selecting hyperparameters and to stop training on any future hyperparameter sets when it predicts that a previously tested hyperparameter set has achieved a best loss measure. This is considered a “global mode” of operation. Ex. 1001, 5:19-38, 18:26-63. In the global mode, the method uses other known stopping criteria, and “may cause stopping of training in a ‘global’ mode as soon as a probability that the likely best loss of the NN is lower than the best loss of other NNs which have been trained with other hyperparameters is lower than a threshold, or when a wait or patience value exceeds a threshold. *Id.*, 18:30-35.

67. The ’968 patent summarizes these two modes, stating “[a]fter training period (e.g., an epoch) for a NN, a model trained using training data from other NNs may return a probability of improvement in the loss of the NN or a probability that the likely best loss of the NN is lower than the best loss of the other NNs for which hyperparameters have been chosen.” Ex. 1001, 3:28-35; 16:13-17:9; Table 3. As

mentioned in background, the loss “may define a deviation from an expected or correct output value for a given input.” *Id.*, 2:46-48. The ’968 patent explains that multiple models are used for calculating the probability of improvement of the training loss in some embodiments: one model may determine a mean expected best loss for the NN, and another model may determine the variance of that mean. Ex. 1001, 13:36-42. Thereafter, “calculations” may be used to determine the probability, such as by “determining the variance for the mean, calculating the square root of the variance, and using a function such as the known cumulative distribution function (CDF) having as inputs the mean, the square root of the variance, and the best loss seen so far.” Ex. 1001, 13:43-54. Then, “[t]raining may be stopped if the probability is less than a threshold, or a wait value is greater than a wait threshold.” *Id.* The ’968 patent describes benefits provided by these two embodiments, stating they “may improve prior NN training by making training or learning more efficient, by stopping training at a point where little or no further improvement is expected, or less than a certain amount of improvement is expected.” *Id.*, 26:19-23. The embodiments may assist in selecting hyperparameters for training a NN “by stopping training of a NN with a set of hyperparameters at a point where it is unlikely that the current NN can be trained to have a loss better than the best seen so far for other NNs.” *Id.*, 26:23-28.

68. The '968 patent discloses embodiments “that may stop training of a NN if it is determined that likelihood of improvement in loss of a best loss seen is below a threshold, or the training has proceeded to the mean expected or final value of training (typically after a certain number of initial training cycles or epochs), or if training has ‘stalled’ and does not improve (e.g., over a historic best value) over a certain number of training cycles or epochs.” Ex. 1001, 5:7-18.

69. Embodiments of the '968 patent “may be used for choosing hyperparameters of a NN,” for example, “hyperparameters may be chosen for a NN as is known in the art, and the NN may be trained, the training process determining if to stop training the NN if the likelihood of improvement (e.g. to a final loss) of the NN beyond a best loss seen for other NNs with other sets of hyperparameters is less than a threshold.” Ex. 1001, 5:19-25. After “hyperparameters for a NN are decided, the NN may be trained by a process which stops training when the likelihood of improvement (e.g. to a final loss) of the NN beyond a best loss seen before for this NN is less than a threshold.” Ex. 1001, 5:26-30; 13:36-42. Training may otherwise be stopped “based on a comparison of the current loss for the NN to a predicted best loss (e.g. final best loss) for the NN.” Ex. 1001, 5:30-38. The '968 patent provides an example in which, “if an expected or predicted training loss (e.g. that returned by a model) is greater than or equal to the actual training loss of the target NN, or if

patience has been exhausted (e.g., if a wait value, which is incremented if no improvement is seen, is greater than a wait threshold) training may be stopped.” *Id.*

70. In the ’968 patent, “one or more models, possibly including sub-models, may be built or trained using training data of many other NNs (e.g. collected in the past, before training the target NN, or before choosing hyperparameters for a set of NNs to be compared), some having different structures and uses than the target NN being trained, and the model may be used to return an expected loss at a certain time or training period for the target NN.” Ex. 1001, 5:39-51. A “model may take as input data describing the NN being trained such as model parameters (e.g. hyperparameters) and loss data such as a loss history and a current loss (which may be included in the loss history) and return one or more values describing a training loss.” *Id.*

71. The ’968 patent also explains that, “[t]ypically, there is a delay before determining whether or not to stop to prevent stopping from occurring at the very beginning of training in an early training cycle, period or epoch (e.g. the first or second epoch).” Ex. 1001, 5:52-55. As an example, “a process may wait for a certain number of iterations or epochs, e.g. 10 epochs, for a target NN being trained, before making a prediction and determining if training should be stopped.” Ex. 1001, 5:56-58. The “prediction may be adjusted for every new epoch of data received, and thus

the mean predicted loss value may rise or fall based on loss values being reported.”

Ex. 1001, 5:59-61.

72. In the '968 patent, a predictive model may be incorporated “into an early stopping algorithm that is responsible for stopping a NN training process at the point at or as soon as the minimum predicted validation loss is achieved,” and training may be terminated “based on a model, such as a prediction provided by a learning curve model.” Ex. 1001, 5:62-67.

73. A model of the '968 patent may be updated “using data gained from training the particular or target NN,” and the “comparison of expected and actual loss may be modified by a threshold or range such that if the difference is less than the threshold or within the range it can be deemed that stopping should occur.” Ex. 1001, 6:1-6.

74. The '968 patent also explains that some embodiments “may be built on top of a model estimating the minimum value of a validation metric for a given experiment configuration,” and that a “large amount of loss function data, or training losses, may be collected, from multiple different NNs (typically different NNs in a variety of different contexts, with a variety of different architectures, performing a variety of different analysis tasks, having been trained for different numbers of epochs), along with hyperparameter and possibly other data, e.g. via an experiment or learning management platform, to build or train a predictive model that can

estimate an interval in which an evaluation metric is likely to fall.” Ex. 1001, 6:7-29.

75. According to the ’968 patent, some “embodiments may, unlike some prior art methods, not partition data based on specific NN architectures,” and some “embodiments do not need to use information regarding architecture type, or architecture specific parameters, such as the number of convolution filters, to predict the minimum value of the learning curve.” Ex. 1001, 6:7-29.

76. Because the models of the ’968 patent “have been trained or developed using training losses of multiple NNs, a particular NN when applied to the model to determine stopping gets the benefit of a model trained on data gathered from many NNs including those other than that particular NN being analyzed for stopping.” Ex. 1001, 6:30-35.

77. The ’968 patent provides 2 flowcharts – FIGS. 4A and 4B that describe the embodiments in detail. As described therein, “the operations shown in FIG. 4A in one embodiment may cause early stopping of NN training for a NN of a known configuration, and the embodiment shown in FIG. 4B may allow choosing among the best NN structure.” Ex. 1001, 18:35-39. These figures are shown below.

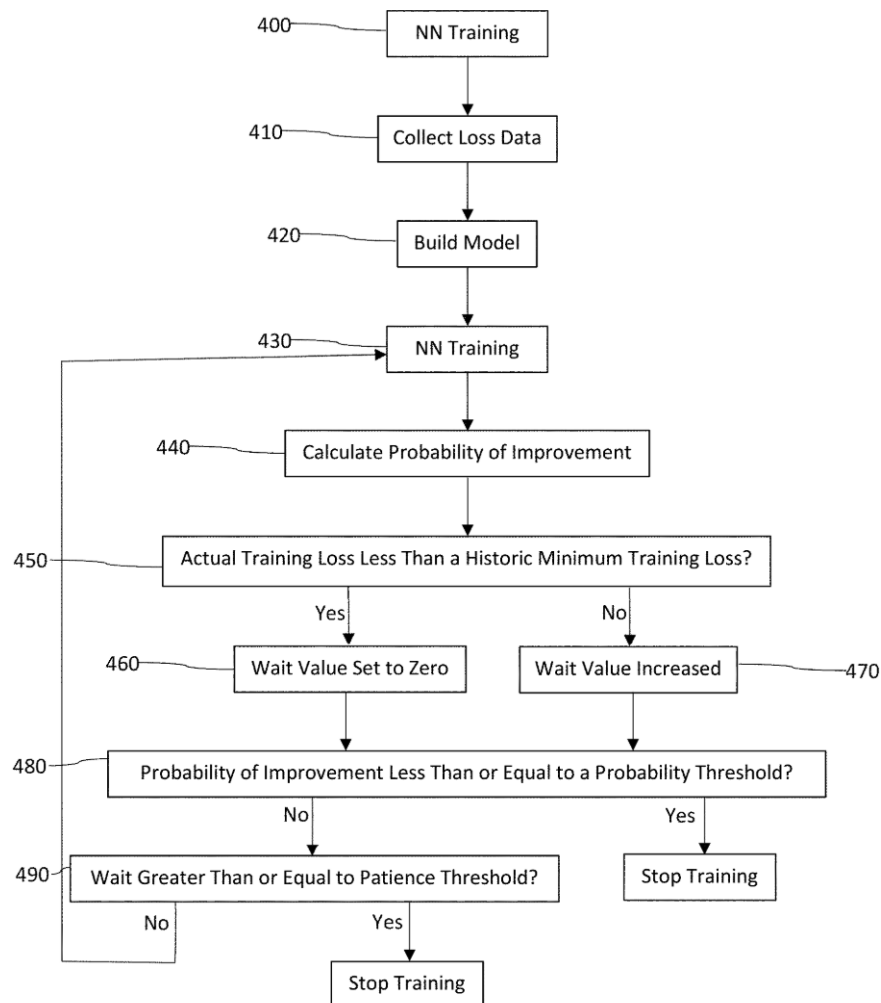
Ex. 1001, FIG. 4A

Fig. 4A

78. Operations 400-420 describe the process of training models that “determine an efficient time to stop training” another model. Ex. 1001, 12:38-13:9. These models may be tree-based models that may, “when presented with input data such as a particular NN’s loss data and hyperparameters, provide estimates of the mean or average expected change in the loss curve, and a variance.” *Id.*, 13:1-9.

79. Operations 430-490 describe the process for training a NN and determining whether an early stopping criterion applies to stop the training process, using the models built in operations 400-420. Ex. 1001, 13:10-15:21. Training begins (or continues for a next training interval or epoch if training has already begun) at operation 430. *Id.*, 13:10-35. The '968 patent states that when training first begins, the early stopping criteria may be delayed “to prevent early stopping if stopping conditions are met in an early iteration or epoch (e.g., epoch 5).” *Id.*, 13:24-27.

80. In Operation 440, a probability of improvement over a best loss seen for this NN may be calculated. Ex. 1001, 13:36-14:22. This calculation considers the mean expected best loss for the NN and the variance of that mean to determine probability. *Id.* The '968 patent points to Table 3 (reproduced below) as showing pseudo-code for calculating this probability.

Ex. 1001, 22:18-23:10 (TABLE 3)**TABLE 3**

Algorithm 3 Predict**Input:**

current_loss_history = $[y_0, y_1, \dots, y_{t-1}]$ // For NN being evaluated
 Φ = hyperparameters
best_metric = $R > 0$ // if “local” may be best loss seen for specific NN; if global may be best loss across other NNs trained, typically with different hyperparameters

Output:

prediction = $R > 0$ // predicted loss history, expressed as mean below
interval = $[R > 0; R > 0]$
 $\text{prob}_i = R \in [0, 1]$ // likelihood, between 0 and 1, that this NN’s loss will ever be better than input best_metric

1: procedure PREDICT(loss_history, Φ , best_metric)

2: origin = y_0 // origin, used to normalize or re-scale, is set to the first loss recorded for the NN

3: current_min = min(loss_history)

4: mean = origin $\times (1.0 - m_\mu(\text{loss_history}, \Phi))$ // input loss_history and hyperparameters to model to receive from model mean, the un-normalized likely best loss predicted to be achieved for this NN

5: var = origin $\times (m_\sigma^2(\text{loss_history}, \Phi))$ // Input data to model to receive from model predicted variance

TABLE 3-continued

6: std = square root of var

7: max = mean + $1.96 \times \text{std}$

8: min = mean - $1.96 \times \text{std}$

9: $\text{prob}_i = \Phi_{\text{mean,std}}(\text{best_metric})$ // Probability of improvement from a normal CDF (cumulative distribution function) parameterized by the mean and standard deviation; $\Phi_{\text{mean,std}}$ is CDF, which expresses a probability distribution

10: return mean, [min, max], prob_i // [min, max] is pair of values returned as interval

81. In Operation 450, the method determines if the actual training loss (as calculated) is less than a historic minimum training loss. Ex. 1001, 14:23-24.

82. In Operations 460 and 470, the method increases or resets the wait value, depending on the result of Operation 450. Ex. 1001, 14:25-61. If the current or actual training loss is less than a historic minimum, the wait value is not increased and may be set to zero. Ex. 1001, 14:25-29. If the current or actual training loss is not less than a historic minimum training loss, the wait value may be incremented,

to indicate that another epoch (or other training interval) has passed without achieving an improved training loss. Ex. 1001, 14:39-53.

83. In Operation 480, the method determines if the probability of improvement over a best loss for this NN (e.g., with the current set of training parameters) is less than or equal to a probability threshold. Ex. 1001, 14:63-67. If so, then training is stopped. If not, in Operation 490, the method determines if the wait value (that was incremented in Operations 460-470) is greater than or equal to a threshold, known as a “patience” value. If so, training is stopped. Ex. 1001, 15:1-21. If not, the method returns to Operation 430 to continue training for a next training interval. *Id.*

84. FIG. 4B, reproduced below, is a method of assisting in selecting among a plurality of NN structures. Ex. 1001, 18:38-39.

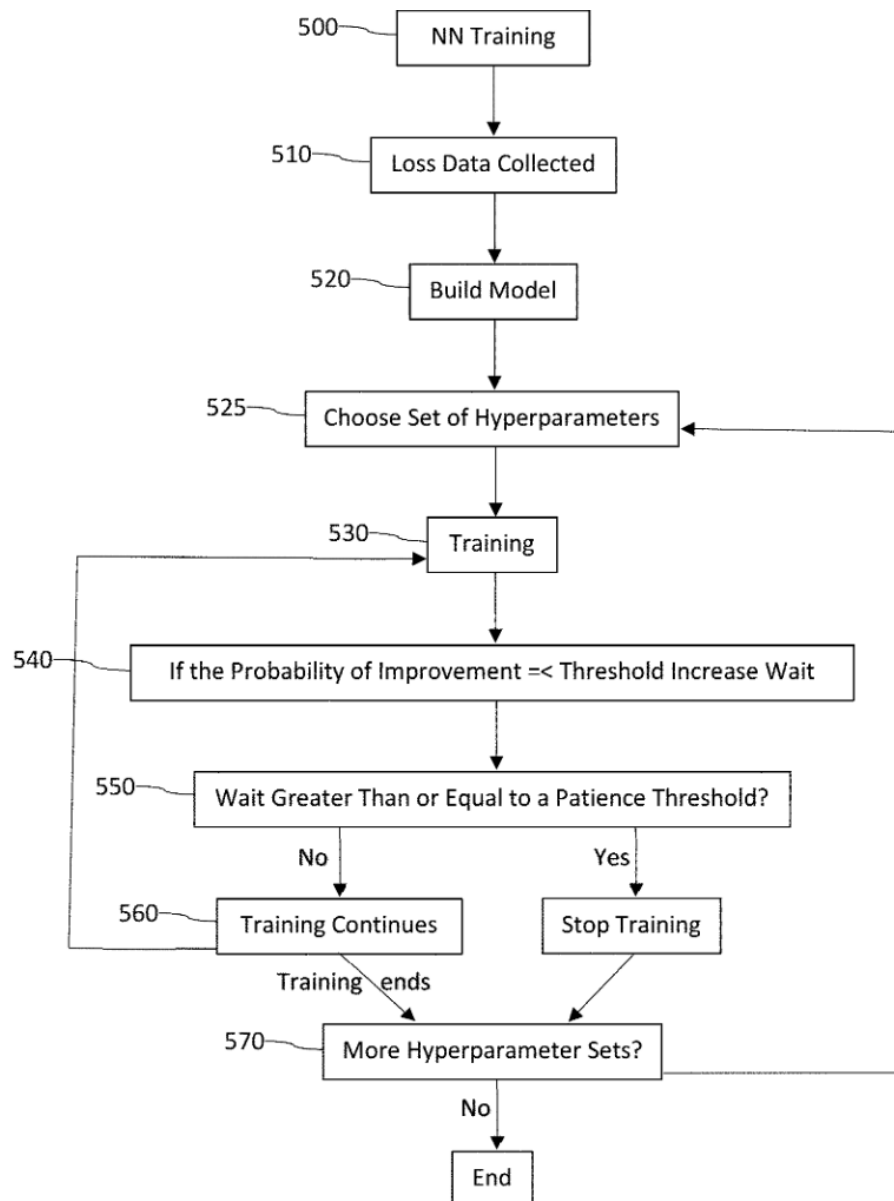
Ex. 1001, FIG. 4B

Fig. 4B

85. Operations 500-520 relate to building models used to assist in selecting hyperparameter values for training a NN. Ex. 1001, 15:23-36.

86. In Operation 525, a set of hyperparameters may be chosen for a NN, for example “by traversing or randomly choosing an entry in a grid, each grid entry including a set of hyperparameters.” Ex. 1001, 15:37-40.

87. In Operation 530, training of the NN begins (or continues for the next training interval, if training has already begun). This training process “of a specific NN having a specific or unique set of hyperparameters chosen in operation 525 may be called an ‘experiment’, and a number of experiments may take place. The resulting best loss for each NN in an experiment may be recorded.” Ex. 1001, 15:41-53. In some embodiments of Operation 530, “one best loss value over all different NN-hyperparameter combinations seen thus far may be recorded (e.g. as *best_metric*); when a new best loss occurs, it may replace the best loss value. *Id.*

88. In Operations 540-560, various early stopping criteria are evaluated. In Operation 540, the method determines “if the probability of improvement over a best loss seen for other NNs is less than or equal to a probability threshold” if so, a “wait value may be increased.” Ex. 1001, 15:54-57. In Operation 550, if a wait value is greater than or equal to a threshold (a “patience value or threshold”), then training of this particular NN may be stopped. *Id.*, 15:58-62. If training continues in accordance with Operation 560, the training continues until some stopping criterion is reached, or the end of the training process is reached. *Id.*, 15:63-15:3. Once the training process reaches some stopping criteria, the process determines if there are

more hyperparameter sets to evaluate, as indicated at Operation 570. *Id.*, 16:4-11. If so, the process returns to Operation 525 to choose a new set of hyperparameters.

89. The '968 patent provides pseudo-code for the local” and “global” modes of operation in Tables 1 and 2, reproduced below. Ex. 1001, 19:8-20.

Ex. 1001, 20:9-21:7 (TABLE 1)

TABLE 1

Algorithm 1 Early Stopping (Local)
Global Variables:
 wait, initialized to 0 // wait is a wait parameter to be incremented if no improvement
Input:
 loss_history = $[y_0, y_1, \dots, y_{t-1}]$ // Historical loss values for the NN to be analyzed for stopping training. Typically each y is collected at the end of each sequential epoch or other period of time, but other ways of organizing a series of loss data may be used. Data is for epochs or periods labelled starting at zero. In one embodiment normalization and conditioning are performed in the model used in PREDICT, shown in Table 3. loss_history excludes the current loss value y_t
 current = y_t // the current loss value for the NN, e.g. value subsequent to y_{t-1}
 Φ = hyperparameters for the NN
 patience = $Z > 0$ // threshold for wait
 threshold = $R \in [0, 1]$ // threshold for probability, range 0-1
Output:
 True/False // True tells calling procedure to stop training; False=keep training
1: procedure EARLY STOPPING (loss_history, current, Φ , threshold, patience) // call procedure for local stopping with input
2: current_min = min(loss_history) // current_min is the lowest value in loss_history for NN
3: current_best = min(current, current_min) // minimum from loss history and current loss
4: prediction, interval, prob_{*t*} = Predict(loss_history, Φ , current_best) // Example of predict is described in Table 3; interval and prediction may not be used, but may be used in some embodiments
5: if current < current_min then // If there is improvement (current NN loss better than historic), reset the wait value
6: wait = 0
7: else // If no improvement is seen, increment the wait value
8: wait = wait + 1
9: if prob_{*t*} ≤ threshold then // If probability of improvement less than threshold, stop training;
10: return True // True is a determination to stop training; signals the procedure calling EARLY STOPPING to stop training the NN

TABLE 1-continued

11: if wait ≥ patience then // If the wait value is greater than or equal to patience value there has been no improvement over some time, so stop training
12: return True
13: return False // signal to calling procedure to continue training NN; if any True condition is not met, return False; False is a determination to continue training;

TABLE 2

Algorithm 2 Early Stopping (Global)

Global Variables:

wait = 0

Input:

current_loss_history = $[y_0, y_1, \dots, y_{t-1}]$

best_metric = $\min([\min(\text{loss_history}_0); \min(\text{loss_history}_1); \dots \min(\text{loss_history}_n)])$ //

loss_history_x is the loss history from NN_x trained NNs having hyperparameters different from

NN being trained currently; best_metric is lowest loss found among loss histories for all other

NNs; alternately one “best seen so far” variable may be used

patience = $Z > 0$

Φ = hyperparameters for the NN

threshold = $R \in [0, 1]$ // threshold for probability, range 0-1

Output:

True/False // True tells calling procedure to stop training; False=keep training

1: procedure EARLY STOPPING(loss_history, current_loss_history, Φ , start, interval, threshold, patience)

2: prediction, interval, prob_t = Predict(loss_history, Φ , best_metric)

3: if prob_t $\leq$ threshold then // If the probability this NN will be trained to a loss better

than best seen with other NNs is less than a probability threshold, increase wait else if probability $>$ threshold, return false below and continue training

4: wait = wait + 1

5: if wait $\geq$ patience then

6: return True // Stop early; no probability seen over a certain period that this NN will have better loss than others, thus stop training this NN; True is a determination to stop training

7: return False // If the probability NN will be trained to a loss better than best seen with other NNs is greater than threshold, do not stop training; False is a determination not to stop training

B. The '968 Patent's Prosecution History

90. Below is a brief summary of the '968 patent's prosecution history. This summary is not intended to address or summarize the '968 patent's entire prosecution history. Rather, it is intended to summarize the parts that may be relevant

to my analysis below.

91. U.S. Patent Application No. 16/707,265 (the “’265 application”) was filed December 9, 2019. Ex. 1001, Cover. Ex. 1002, 105-168.

92. A Non-Final Office Action was issued September 27, 2022, in which a restriction was required between an invention to a “method for training a neural network including a probability that the likely best loss of the NN is lower than the best loss of the other NN” (or original claims 1-6 and 14-17) and an invention to a “method for training a neural network including a probability of improvement in the loss of the NN” (or original claims 7-13 and 18-22). Ex. 1002, 59-61. The Non-Final Office Action also rejected original claims 7, 13, and 18 under 35 U.S.C. §103 over Prechelt (NPL titled: Automatic early stopping using cross validation: quantifying the criteria) and Nixon (U.S. 2020/0104678), claims 8 and 19 under 35 U.S.C. §103 over Prechelt, Nixon, and Albon (NPL titled: Neural Network Early Stopping), and claim 12 under 35 U.S.C. §103 over Prechelt, Nixon, and Reisser (U.S. 2019/0354865). Ex. 1002, 62-67. Original claims 9-11 and 20-22 were identified as being directed to allowable subject matter. Ex. 1002, 67-68. The Examiner indicated that “[n]one of the prior art of record teaches or fairly suggest:” “comprising increasing the wait value if the current loss of the NN is not less than the minimum loss in the loss history for the NN,” “setting the wait value to zero if the current loss of the NN is less than or equal to the minimum loss in the loss history for the NN,”

and “wherein determining using a model a probability of improvement in the loss of the NN comprises obtaining from a leaf of at least one tree data structure data relevant to the NN.” Ex. 1002, 67-68.

93. The Applicant responded to the Office Action on January 23, 2023, electing to prosecute claims 7-13 and 18-22 without traverse. Ex. 1002, 47. The Applicant also argued that Prechelt did not teach or suggest the claimed probability of improvement found in claim 7. Ex. 1002, 48. Specifically, the Applicant explained that independent claim 7 requires “determining...a probability of improvement in the loss of the NN,” and that, “[b]ased on this probability, claim 7 determines whether or not to stop training: ‘if the probability is less than a threshold, or a wait value is greater than a wait threshold, stopping training.’” *Id.* The Applicant noted that the Office Action cited to Prechelt’s “generalization loss” for “the limitation of a probability,” and argued that “Prechelt’s generalization loss is the ‘relative increase of the validation error over the minimum-so-far.’” Ex. 1002, 48 (citing Prechelt, p. 4). The Applicant argued that “Prechelt teaches a generalization loss ... which measures increase of error,” and that “[t]his does not teach or suggest the claimed probability of improvement; rather Prechelt measures an absolute measure of increase.” Ex. 1002, 48. The Applicant stated that, “[w]hile each of Prechelt’s measure and the claimed probability may be used to determine when to stop training, each goes about it in a different manner.” Ex. 1002, 48.

94. The Applicant also argued that Nixon “trains a NN using data from the same NN in a distributed manner, which also cannot teach or suggest training a model which determines a probability that the likely best loss of the NN is lower than the best loss of the other NNs,” and that Nixon does not “use the loss information as input to a model; rather Nixon para. [0028] describes that loss of a NN measures how well NN 120 performs after being trained.” Ex. 1002, 48-49. The Applicant concluded that “Nixon’s loss is merely signal of a model’s performance at predicting parameters for a target network.” *Id.*

95. A Notice of Allowance was mailed February 17, 2023. Ex. 1002, 28. Claims 7-13 and 18-22 were allowed and renumbered 1-12. Ex. 1002, 33. The Examiner stated that “Prechelt does not teach a model that has been trained using training losses of a plurality of NNs other than the NN.” Ex. 1002, 34. The Examiner also stated that “the value disclose [sic] by Prechelt for determining the loss is an absolute value and not a probability and Nixon’s optimizer does not exclude the other than the NN.” Ex. 1002, 34-35.

C. Claim Construction of the ‘968 Patent

96. I have been informed that claim terms are typically given their plain and ordinary meanings, as would have been understood by a person of ordinary skill in the art at the time of the earliest alleged priority date. I have further been informed that when considering the meaning of any terms in the challenged claims of the ‘968

patent, I should apply the plain and ordinary meaning of those terms. I have further been informed that in considering the meaning of the claims, one must consider the language of the claims, the specification, and the prosecution history of record.

97. I have been informed that in general, a preamble limits the invention if it recites essential structure or steps, or if it is necessary to give life, meaning, and vitality to the claim. I have further been informed that a preamble is not limiting where a patentee defines a structurally complete invention in the claim body and uses the preamble only to state a purpose or intended use for the invention. I have further been informed that dependence on a particular disputed preamble phrase for antecedent basis may limit claim scope because it indicates a reliance on both the preamble and claim body to define the claimed invention. I have further been informed that clear reliance on the preamble during prosecution to distinguish the claimed invention from the prior art transforms the preamble into a claim limitation because such reliance indicates use of the preamble to define, in part, the claimed invention.

98. For the purposes of this Declaration and the arguments made herein, I have relied on the ordinary and customary meanings of the claim terms. I reserve the right to address constructions of the claim terms later in this proceeding, if needed.

VIII. Summary of the Prior Art

A. International Publication No. WO 2018/175098 to Baker (Ex. 1005)

99. Baker is a printed publication of PCT application PCT/US2018/020887, filed March 5, 2018, and published September 27, 2018. Ex. 1005, 1. I understand that Baker has been attached as Ex. 1005. *See* Ex. 1005. Baker was not cited and was not considered during prosecution.

100. Baker describes the use of a “coach” machine learning system to help build a “student” machine learning system that minimizes a loss or some measure of the cost of loss. Ex. 1005, ¶¶[0004], [0007]. According to Baker, “by monitoring a student machine learning system, the coach machine learning system can learn (through machine learning techniques) ‘hyperparameters’ for the student machine learning system that control the machine learning process for the student learning system.” *Id.* The coach machine learning system (i.e., a secondary machine learning system) can help build and train student machine learning systems having the same or different type of machine learning architecture. Ex. 1005, ¶[0005]. Baker’s training process involves well-known backpropagation techniques and continues until a stopping criterion is reached; Baker’s learning coach is trained using data from the current student machine learning model and other student machine learning models.

101. The learning coach machine learning system can be configured to modify the student learning system, such as modifying the hyperparameters and/or architecture of the student learning system (e.g., making “additions” to the student

learning system) to improve the capability of the student learning system while guaranteeing that there will be no degradation in performance. Ex. 1005, ¶¶[0007], [0044]. Baker describes the “performance” of a student machine learning system in terms of “error rate or some other measure of the cost of the errors.” *Id.*

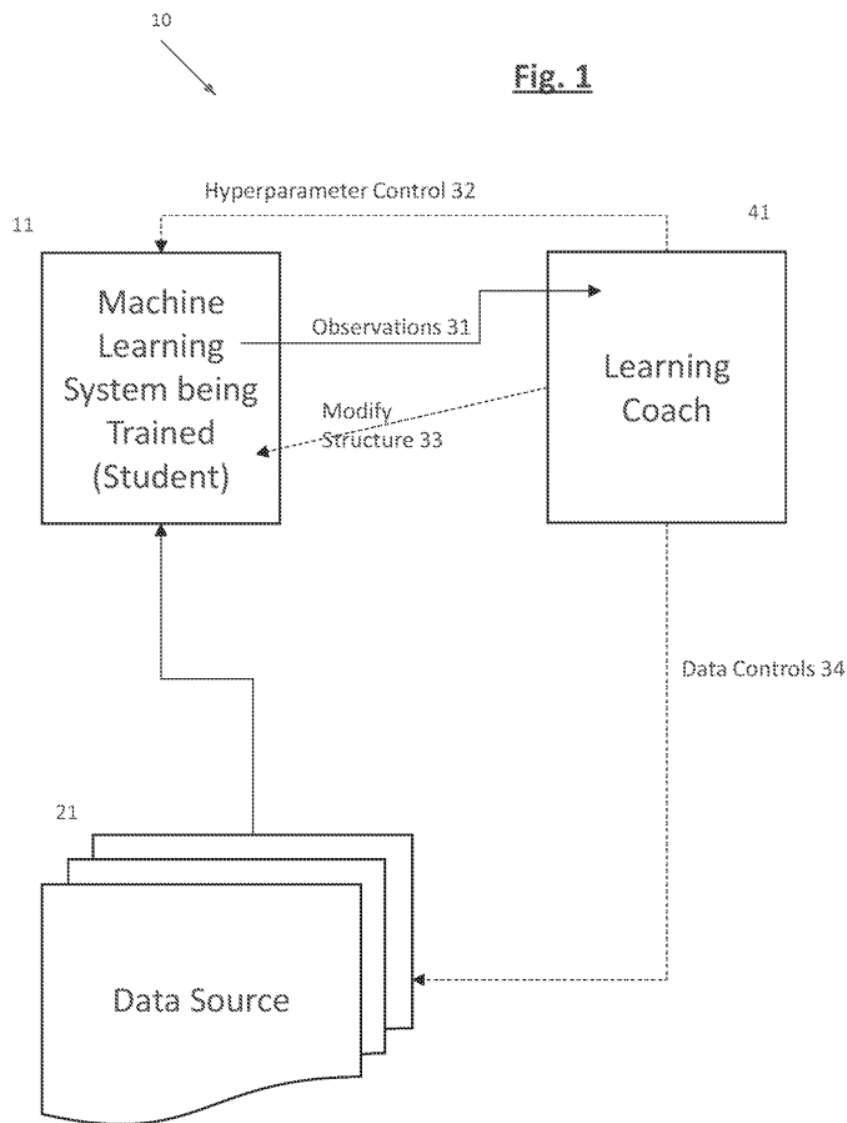
102. The learning coach improves the capability of the student learning system using objectives that are specific to the learning coach, such as to “optimize some combination of the cost of errors and the cost of performing the computation.” *Id.*

103. Baker describes some embodiments in which the system also includes a “computerized reference system for generating a set of training data for the student system.” Ex. 1005, ¶[0008]. The computerized reference system can be controlled by a learning experimental system, which can determine a cost function for the learning coach based on observations from the reference system and the student. *Id.*

104. Figure 1 of Baker, reproduced below, shows a computer system 10 including the student machine learning system 11 and a learning coach machine learning system 41 “that controls some aspects of the learning process and aids the student machine learning system 11 in its process of learning the patterns.” Ex. 1005, ¶[0025]. The learning coach can be a machine learning system, as an example, “a pattern recognition system that recognizes patterns of learning performance of a ML

system, such as the student 11 or some other ML system.” Ex. 1005, ¶¶[0025]-[0026].

Ex. 1005, FIG. 1



105. Baker explains that the student and learning coach are trained for different purposes. The student learns to recognize patterns in data supplied by a data source so that it can later be deployed for some intended use. Ex. 1005, ¶¶[0025]-

[0026] The goal of the learning process for the student machine learning system is to minimize error rate or some other measure of the cost of errors. Ex. 1005, ¶¶[0007], [0044]. The learning coach, on the other hand, does not try to recognize the patterns that the student learning system is learning to recognize. Ex. 1005, ¶[0026]. The learning coach learns from and processes a different set of data from the student machine learning system. *Id.* “The primary data processed by the learning coach 41 are observations 31 of the internal state of the student machine learning system 11 and data that student system 11 computes during its training process.” Ex. 1005, ¶[0026]. When deployed, the learning coach controls the learning process of the student machine learning model and the structure of the student machine learning model to optimize some combination of the cost of errors (by the student machine learning model) and the cost of performing the computation of the learning process for the student machine learning model. Ex. 1005, ¶¶[0007], [0044]. The learning coach controls the learning process of the student machine learning model by manipulating different characteristics of the learning process and the student machine learning system, such as by selecting the training data for the student, by “controlling hyperparameters 32 of the student 11 and changing the internal structure 33 of the student machine learning system.” *Id.* For example, the learning coach can change the internal structure of the student machine learning system embodied as a DNN by changing the number of layers in the DNN, add or delete connections

between nodes, and/or add or delete nodes in one or more layers of the DNN. Ex. 1005, ¶[0043].

106. For the sake of explanation, Baker describes two types of parameters. “Trainable parameters” are features of large complex machine learning systems (such as a student learning system 11 embodied as a large, deep feed-forward neural network (a DNN)) “for which good values need to be found to try and minimize some measure of the cost of errors in the pattern recognition process.” Ex. 1005, ¶¶[0027], [0029]. The process of “learning” in a machine learning system includes finding these values. *Id.* “Hyperparameters” are control parameters “that tune the learning process to work better on the kind of data for a particular task.” *Id.* Hyperparameters are not “learned” by the machine learning system (e.g., the student machine learning system), but are typically specified by a system designer to control the learning process of the machine learning system. *Id.* The learning process is typically iterative, including testing how different hyperparameters effect different trainable parameters, and determining error costs for each tested set of trainable parameters caused by a hyperparameter selection. *See id.*, ¶[0030]. Baker describes that the learning coach 41 controls some or all of these hyperparameters, instead of the system designer. Ex. 1005, ¶[0028].

107. Baker recognizes that the use of a learning coach machine learning system to control some or all of the hyperparameters of training a student machine

learning system can work in many different implementations. However, as an example, Baker describes the use of hyperparameters for training deep feed forward neural networks. Ex. 1005, ¶[0030]. Baker describes a specific training method as “an iterative process of stochastic gradient descent on a log-likelihood error cost function, with a softmax classification for the output nodes, the gradients for each training example being estimated by back propagation of the partial derivative of the error cost function, the estimates being accumulated over minibatches, with one update of the parameter for each minibatch. In the illustrative method, the training process is enhanced by Nesterov’s momentum, and smoothed by L2 regularization.” *Id.* However, Baker notes that “[o]ther well-known training methods could be used in a similar way” and this process “can be applied to any complex machine learning system, including support vector machines (SVMs), random forests, CART analysis, hidden stochastic process models, dynamic Bayesian networks, Boltzmann machines, ensembles of any of these and even heterogeneous ensembles.” *Id.*

108. Baker provides pseudo-code for the student machine learning training process described throughout the disclosure at paragraph [0031], reproduced below, which is repeated until a stopping criterion is reached. However, to the extent it is argued that Baker does not explicitly describe what stopping criterion is used, a POSITA would have been aware of and looked to known stopping criteria to use with Baker’s training process.

Ex. 1005, ¶[0031]

Initialize each weight;

Do until a stopping criterion is reached: {

1. **Input a set (minibatch) of training examples**
2. **For each training example m , set $a_{0,i}(m)$ and perform the following steps:**
 - a. **Feedforward (with softmax output):** For each $l = 1, 2, \dots, L - 1$ compute
$$z_{l,j}(m) = \sum_{i=0}^{n_l} w_{l-1,i,j} a_{l-1,i}(m), \quad a_{l,j}(m) = \sigma(z_{l,j}(m)), \quad a_{L,k} = e^{z_{L,k}} / \sum_j e^{z_{L,j}}$$
 - b. **Compute output error gradient $\delta_{L,j}(m)$:**
$$\delta_{L,j}(m) = -(y_j(m) - a_{L,j}(m)) / n_L$$
 - c. **Backpropagate error gradient:** For each $l = L-1, L-2, \dots, 2, 1$ compute
$$\delta_{l-1,i}(m) = a_{l-1,i}(m)(1 - a_{l-1,i}(m)) \sum_{j=1}^{n_l} w_{l-1,i,j} \delta_{l,j}(m)$$
3. **Gradient descent:** For each $l = L-1, L-2, \dots, 2, 1$ update the weights
$$\widetilde{v_{l,i,j}} = v_{l,i,j}$$
$$v_{l,i,j} \leftarrow \mu v_{l,i,j} - \eta \sum_{m=1}^M a_{l,i}(m) \delta_{l+1,j}(m) / M$$
$$w_{l,i,j} \leftarrow w_{l,i,j} (1 - \frac{\eta \lambda}{n}) - \mu \widetilde{v_{l,i,j}} + (1 + \mu) v_{l,i,j}$$

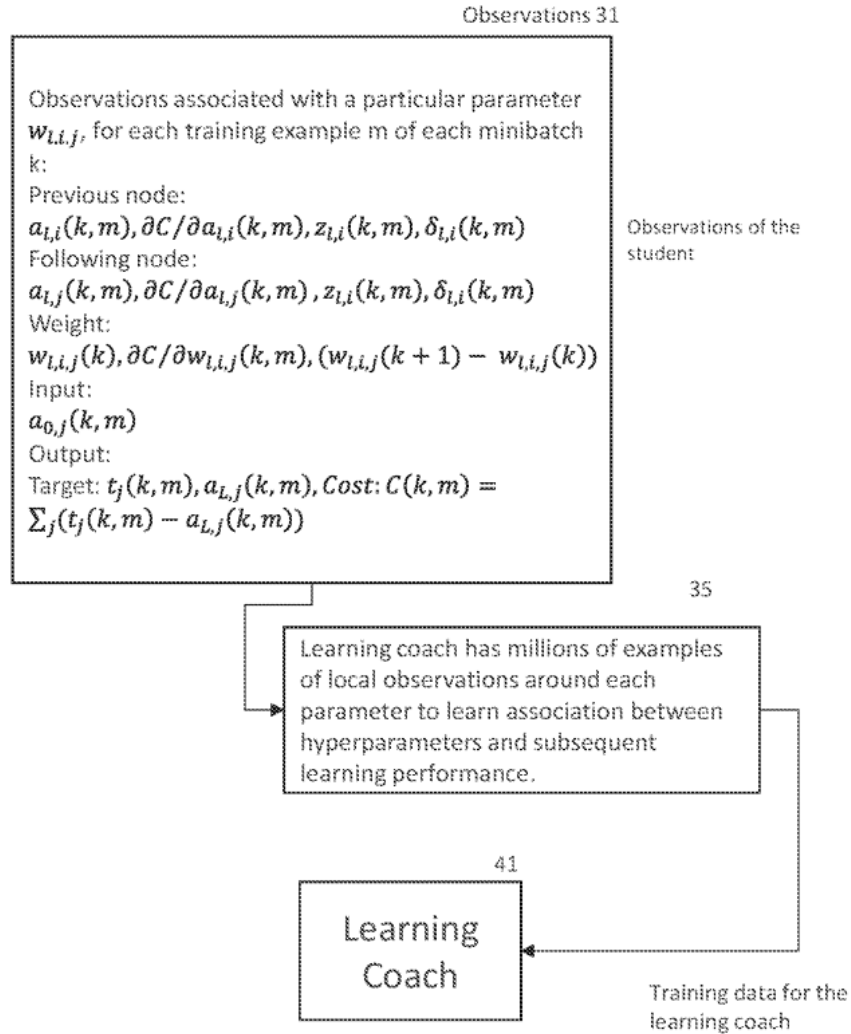
}

109. Baker explains that the learning coach machine learning model learns to optimize hyperparameters (such as M (the size of the minibatch), η (the learning rate), λ (the regularization parameter), and μ (the momentum parameter) of the pseudo-code above) from its own learning process involving observations of the learning process of the student learning system. Where these hyperparameters are

continuous values, the learning coach can be a neural network (*id.*), that models the performance of the student learning system as a regression and may estimate an optimum value of a hyperparameter as a regression function of observations made during the current or previous steps of training of the student learning system. Ex. 1005, ¶[0032]. For categorical hyperparameters, the learning coach may estimate optimum selections as a classification task. *Id.*

110. As mentioned previously, the learning coach is designed to optimize the learning process for the student machine learning model to balance the cost of errors by the student machine learning model with the cost of performing further computations for training the student machine learning model. Ex. 1005, ¶¶[0007], [0044]. Historically, training the student machine learning model involved setting hyperparameters by trial-and-error, which required a test run of the full training procedure on a large amount of training data and then testing the trained student machine learning model on a separate validation set. Ex. 1005, ¶[0033]. To minimize the iterative and resource-intensive learning process for training the student machine learning model, the learning coach is trained using observations of the results of training a student machine learning system using various values for hyperparameters. Ex. 1005, ¶[0039]. The learning process for the learning coach observing updates to parameter weights is illustrated in Figure 3, reproduced below.

Fig. 3



111. As illustrated in Figure 3, as each connection weight parameter is updated, the learning coach obtains an example of the effect of a set of hyperparameter values on the future learning performance of the student learning system (e.g., the resulting error rate or other measure of the cost of the errors). Ex.

1005, ¶¶[0039]-[0040]. Through training, the learning coach models the effect of a set of hyperparameter values on learning performance of the student learning system as a function of the hyperparameters and of the context, (e.g., the set of observations surrounding the connection for each example in the preceding minibatch or multiple preceding minibatches). *Id.* This training data for the learning coach can be include observations recorded during training of multiple student machine learning systems (including “other” student learning systems) (*id.*, ¶[0041]), so that the learning coach can have “millions of training examples from which it can learn to predict the best hyperparameter values for each customized situation” (*id.*, ¶¶[0040]-[0041]), such as to optimize the cost of errors and the cost of performing the computation, such as the cost to further train the student learning system. Ex. 1005, ¶¶[0007], [0040]-[0041], [0044].

112. While the example discussed in reference to Figure 3 describes how the learning coach is trained to manipulate hyperparameters to optimize the cost of errors, Baker describes a similar example for training the learning coach to manipulate the structure of the student machine learning system. Paragraph [0045] provides an example of this learning process (*see also id.*, ¶¶[0132]-[0134]):

[0045] In one illustrative embodiment, learning coach 41 trains itself to make decisions about structural changes in student learning system 11 by running experiments. In this illustrative embodiment, learning coach 41 implements two versions of student learning system 11, one with the

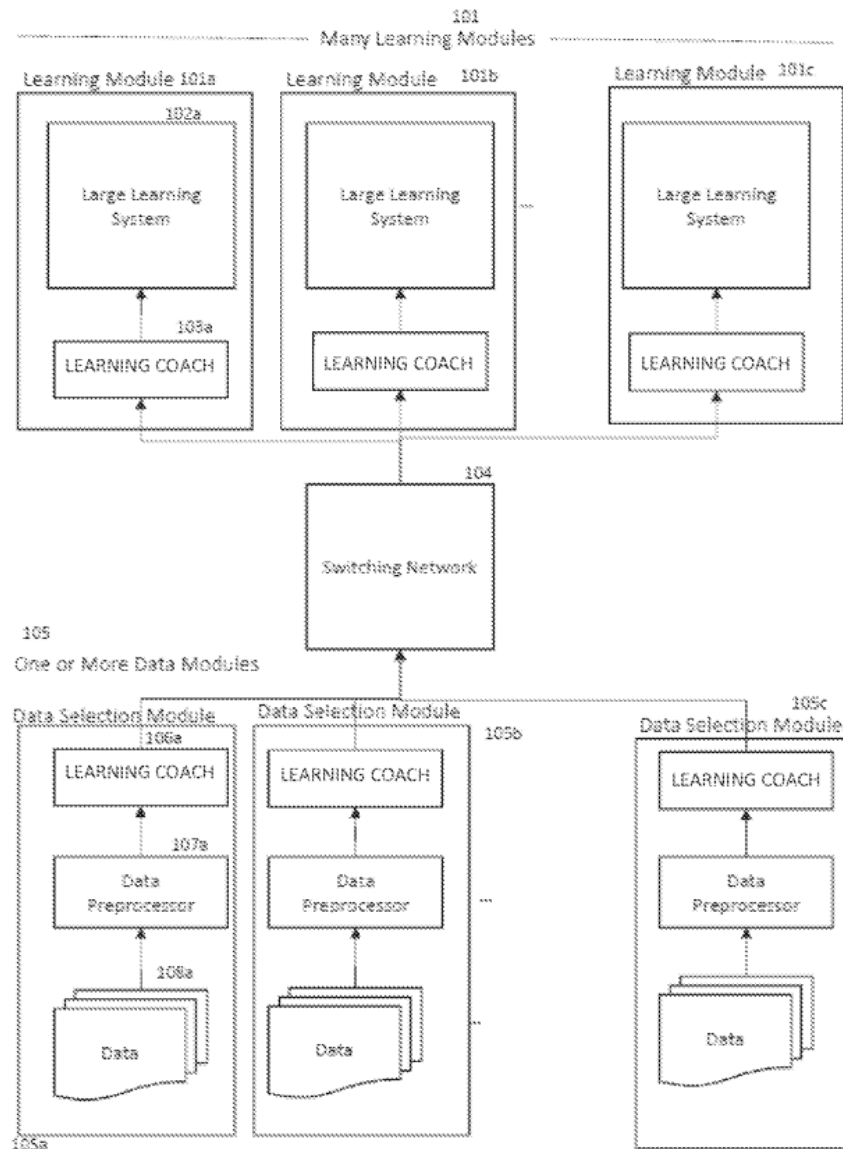
structural change and one without. Learning coach trains both versions of student learning system 11 and evaluates their comparative performance on an independent evaluation set. It also measures or estimates the comparative computational requirements for the two versions. This trial-and-error experimentation, by which learning coach 41 learns to make decisions about structural changes in student learning system 11, is similar to the trial-and-error process of estimating customized hyperparameters illustrated in Figures 2 and 3.

113. Figure 4, reproduced below, shows a system including multiple student machine learning systems (embodied as the “large learning systems” 102 within the “learning modules” 101 and the “data preprocessor 107” within the “data selection modules” 105). Figure 4 is presented as a more complex example for using multiple learning coaches during training of multiple student machine learning models. Each of these student machine learning systems is paired with a respective learning coach (103 and 106). Such an example is relevant where each learning module “is trained to specialize, performing best on particular problems or on particular kinds of data.” Ex. 1005, ¶[0047]. Those student machine learning models – specifically the “large learning systems” and “data preprocessors” may be the same or different types of models. Ex. 1005, ¶[0050]. As an example, “the large learning systems 102 may be deep neural networks, while the data preprocessors 107 may be hidden Markov process models, random forests, or simple k-means classifiers. In some embodiments, either the large learning systems 102 or the data preprocessors 107

may comprise a mixture of multiple types of pattern recognition systems.” Ex. 1005, ¶[0050].

Ex. 1005, FIG. 4

Fig. 4



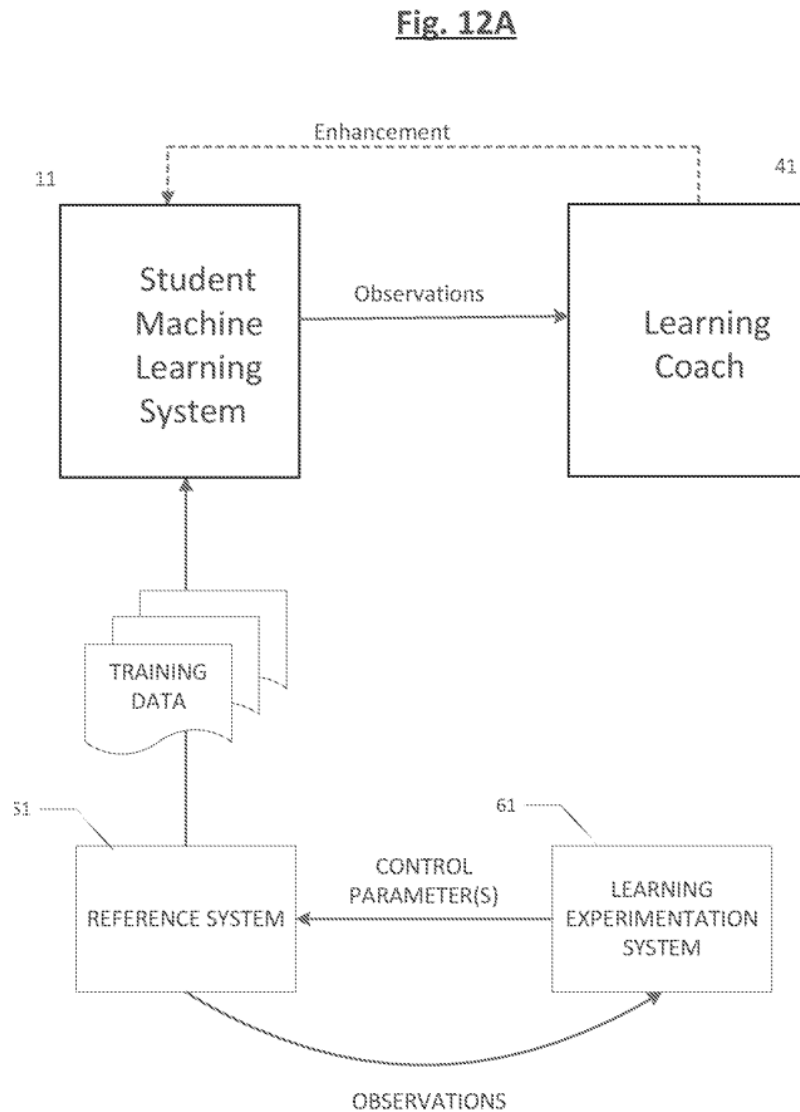
114. Each of the learning coaches 103, 106 can “change the architecture of an associated machine learning system 102, 107,” can “control the learning process

of the learning systems 102, 107,” and/or “may also impose a regularization or change the weight in a regularization of their associated student learning system 102, 107.” Ex. 1005, ¶[0049]. Moreover, if there is more than one associated student machine learning system, “one or more learning coaches 103, 106 may supervise and control the combining of knowledge of the machine learning systems or may combine the output for a particular data item.” *Id.* The learning coaches can even change the number of machine learning systems working together on a task. *Id.* While the discussion of Figure 4 does not describe the learning coaches 103, 107, Baker does state that the “learning coach and the student(s) may be the same or different types of machine learning architectures.” Ex. 1005, ¶[0006].

115. Baker provides additional examples of using one or more learning coaches to change a structure of a student machine learning model between paragraphs [0071]-[0110] (referencing Figures 6-11). For example, the learning coach may use a separate support system with explicit knowledge (which may itself be a machine learning system) to determine when to insert explicit knowledge into certain layers (or nodes or edges) of the student machine learning system. Ex. 1005, ¶[0074]. For example, “the learning coach 303 may present knowledge to the learning network 102 in the form of connection weights between particular nodes.” *Id.*

116. Baker describes another embodiment in reference to Figure 12A, reproduced below, in which training data for the student machine learning system is generated by a reference system that is controlled by a learning experimentation system. The learning coach observes the student machine learning system as it is trained. Ex. 1005, ¶[0111].

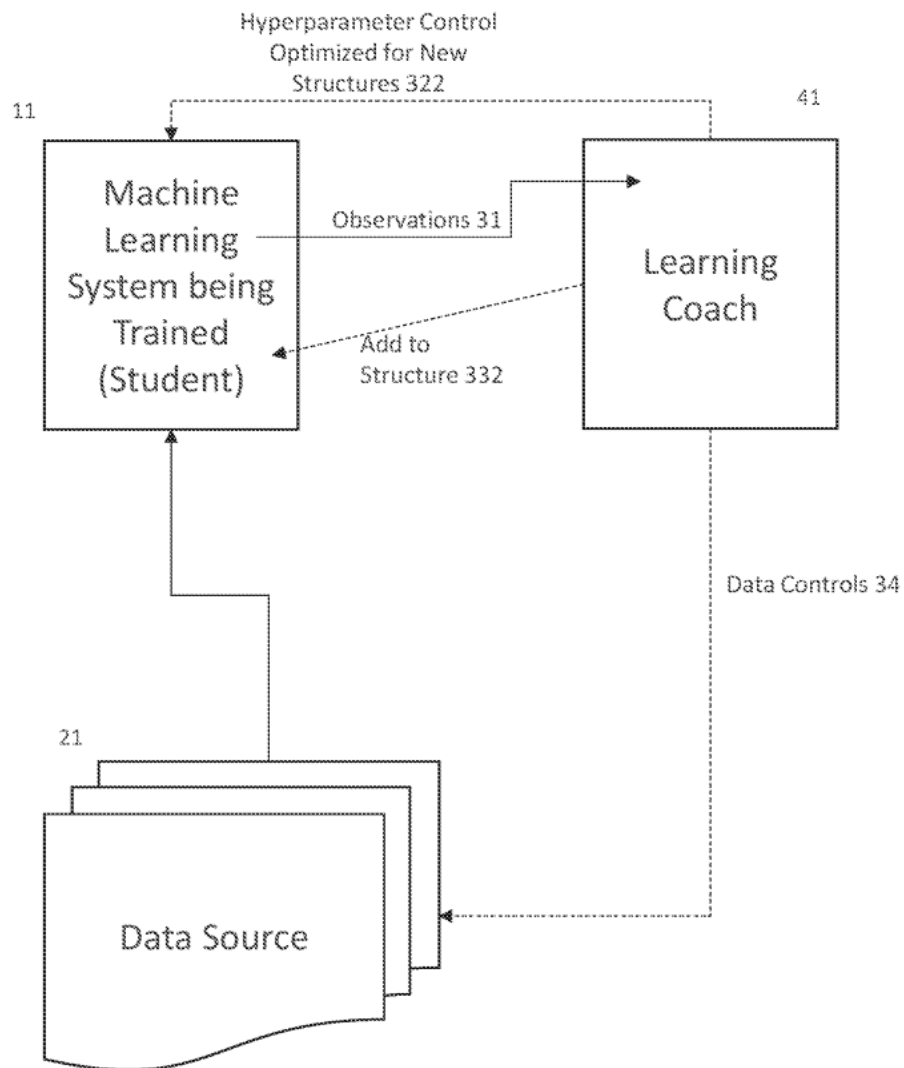
Ex. 1005, FIG. 12A



117. Under the configuration of Figure 12A, the learning experimentation system 61 can assist in training the learning coach by “provid[ing] tasks for the student system 11 such that the learning coach 41 can learn from observing the learning behavior of student system 11 in specially designed situations that will be instructive for the learning coach 41.” *Id.* In the embodiment of Figure 12A, the reference system may be a classifier (for classifying training data), a synthesizer/generator (for generating training data), or a combination of the two. Ex. 1005, ¶[0113]. The reference system can be the same type, or a different type, of machine learning system as the student machine learning system. For example, the reference system can be a neural network, “a support vector machine (SVM) or a random forest.” *Id.* The learning experimentation system 61 can be configured to design experiments such that the student machine learning system learns to imitate the reference system. Ex. 1005, ¶[0118].

118. Figures 15-16 (shown below) are illustrations showing a learning coach 41 that combines its ability to control hyperparameters with its ability to modify the structure of the student machine learning system 11 (a DNN in the examples of Figures 15-16). Ex. 1005, ¶[0147].

Fig. 15

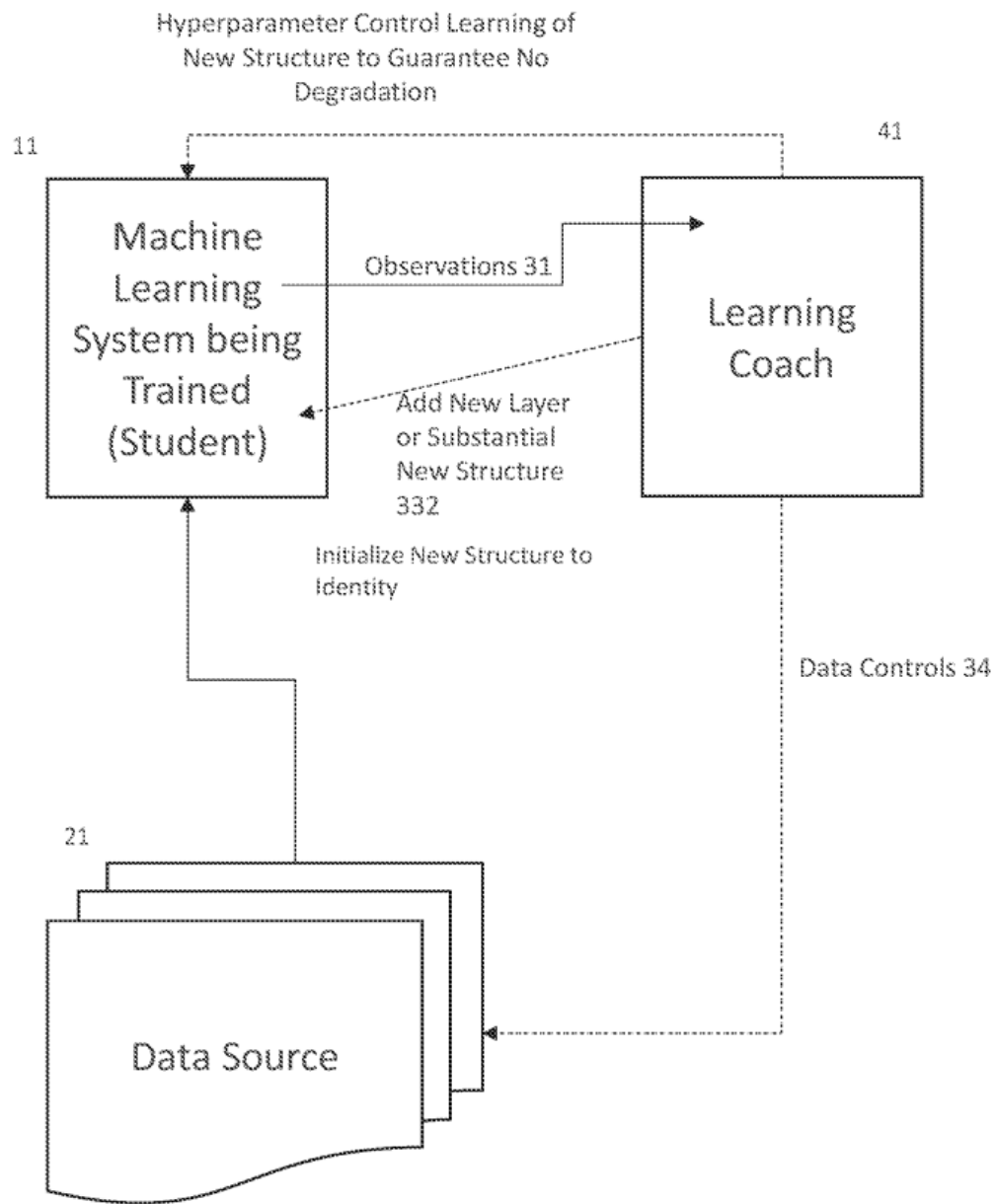


119. Baker explains that the learning coach 41 specifically focuses on controlling the learning rate of different portions of the student machine learning system network – older portions of the network may need a different learning rate from new parts of the network. *Id.* Specifically, Baker states that “this ability is

focused on the issues that are important in this situation by supplying examples of the student learning system being trained in similar situations. That is, the learning coach 41 learns to optimize the relative learning rate of the new additions to the structure of the student system 11 by observing other examples in which nodes and arcs have been added to a student system 11.” Ex. 1005, ¶[0148].

120. Figure 16 is described as an “important special case of the embodiment illustrated in Figure 15 [having] the addition of an entire layer of new nodes and their associated arcs to the network of student learning system 11.” Ex. 1005, ¶[0149]. The described system ensures that “the performance of the larger network at least matches the performance of the smaller network.” *Id.*

Fig. 16



121. The learning coach 41 of Figure 16 initializes weights and biases of the added structure to be an identity mapping, and can add additional structure, as needed. Ex. 1005, ¶[0149]. The learning coach 41 then “manages the further learning

to guarantee that the performance of the now-larger student learning system network 11 on training data is always at least as good on the training data as previous performance of the smaller system.” *Id.* As an example, “whenever a gradient descent step produces a degradation in performance, the learning coach 41 can decrease the step size.” *Id.* Similarly, the learning coach can add or modify other substantial structures of the student system 11 without degradation in performance. Ex. 1005, ¶[0151].

a) System Structure

122. Baker states that the “student learning system 11 and learning coach 41 can be implemented with computer hardware or software.” Ex. 1005, ¶[0046]. The systems can be a part of an integrated computer system that has multiple processing CPU cores. *Id.* One set of processing cores could execute program instructions for the student learning system and another set of processing cores could execute program instructions for the learning coach. *Id.* The processing instructions can be stored in a computer memory that is accessible by the processing cores. *Id.* Baker provides additional examples, including using graphical processing unit (GPU) cores to process the program instructions for the student learning system 11 and/or learning coach 41. *Id.*

B. Shridhar, Kumar, *et al.* “A Comprehensive Guide to Bayesian Convolutional Neural Network with Variational Interference” (Shridhar) (Ex. 1006)

123. Shridhar is a research paper that was published online through multiple publication databases by January 8, 2019. Ex. 1011, 35-27. I understand that Shridhar has been attached as Ex. 1006. *See* Ex. 1006. Shridhar was not cited and was not considered during prosecution.

124. Shridhar discusses a Bayesian Convolutional Neural Network (BayesCNN) using Variational Interference. Ex. 1006, *Abstract*. Shridhar discusses some known early stopping criteria, and proposes other structures to address problems of overfitting on small datasets by adding a measure of uncertainty and regularization to predictions generated by BayesCNN during inference. Ex. 1006, 2. Shridhar recognizes that inferring model posterior in a Bayesian NN is difficult and proposes to approximate the intractable true posterior distributions with variational probability distributions. *Id.* To do this, Shridhar explains how to apply two convolutional operations, one for the mean and one for the variance.

125. In background, Shridhar provides an overview of NN structures, generally. In a NN, a “large number of highly interconnected processing elements (neurons) working together makes a Neural Network solve complex problems.” Ex. 1006, 4. Neural Networks consist of 3 layers “input layer to feed the data to the model to learn representation, hidden layer that learns the representation and the output layer that outputs the results or predictions.” *Id.*

126. Shridhar describes Convolutional Neural Networks (CNNs) in reference to hierarchy models. Ex. 1006, 4. Starting layers of the NN learn simple features of images, such as the location of edges and corners, and subsequent layers learn more complex features such as colors, textures, and other features of an image. *Id.* The neurons in each layer share weights. *Id.* As an example in image processing, neurons in a CNN have weights for a small number of pixels in an image, instead of weights for every pixel in the image. The weights are shared in progressive layers of the CNN, so the CNN successively sees larger portions of the image since subsequent layers combine information about increasingly larger subsets of the image. *Id.* The final layer makes a prediction for the output class. *Id.*

127. Shridhar also describes probabilistic machine learning by walking through an example. Ex. 1006, 6. Bayesian inference uses a prior distribution that is used to represent a prior belief as to which functions are likely to have generated data in a function of interest. *Id.* This probability can be approximated using a predictive distribution function designed to minimize the Kullback-Leibler (KL) divergence between the predictive distribution function and intractable prediction functions that more directly measure the likelihood that a given observation is generated. *Id.* Shridhar explains that “[m]aximizing the KL divergence between the posterior and the prior over a set of random variables w will result in a variational distribution that learns a good representation from data (obtained from log

likelihood) and is closer to the prior distribution. In other words, it can prevent overfitting.” *Id.*

128. Shridhar continues its background discussion by explaining two sources of uncertainty for NN models – each providing a measure of how certain the model is with its prediction. Ex. 1006, 7. *Aleatoric* uncertainty is caused by noise inherent in the observations that make up training data and real-time data, and this cannot be reduced with the collection of additional data. *Epistemic* uncertainty represents uncertainty caused by the model itself, and it can be reduced by collecting additional data. *Id.*

129. Shridhar next summarizes backpropagation, the most commonly used method for training neural networks. Ex. 1006, 8. “Backpropagation is a technique to compute the gradient of the loss in terms of the network weights.” *Id.* First, “input features through the network propagates in the forward direction to compute the function output and thereby the loss associated with the parameters.” *Id.* Then, “the derivatives of the training loss with respect to the weights are propagated back from the output layer towards the input layers. These computed derivatives are further used to update the weights of the network.” *Id.* This happens continuously, and weights are updated during each iteration. *Id.* However, even with backpropagation, many hyperparameters must be tuned through other means, and the time required to optimize those hyperparameters is proportional to the size of the training data set.

Id. Bayesian methods account for uncertainty in parameter estimates and can propagate this uncertainty into predictions that can be used to minimize the resources needed to optimize the hyperparameter selection. *Id.*

130. Shridhar goes on to explain the crux of the paper’s contribution, including applying two sequential convolutional operations to calculate mean and variance for a model output. Shridhar compares the proposed BayesCNN configuration against existing model training structures, including for example, applying an L1 norm. Ex. 1006, 22. These comparative training methods included known early stopping techniques to conserve resources when further training would not yield additional helpful results. Shridhar explains one example of early stopping, when further training is unlikely to yield any changes in validation accuracy (model performance): “[w]e trained a model to some epochs (number of epochs differs across datasets as we applied early stopping when validation accuracy remains unchanged for 5 epochs).” *Id.*

C. Jenatton, Rodolphe et al., “Bayesian Optimization with Tree-structured Dependencies” (Jenatton) (Ex. 1007)

131. Jenatton is a conference paper that was presented during the 34th International Conference on Machine Learning was held on August 6-11, 2017, in Sydney, New South Wales, Australia. I understand that Jenatton was published by August 6, 2017. Ex. 1011, 38-40. I understand that Jenatton has been attached as Ex.

1006. *See* Ex. 1006. Jenatton was not cited and was not considered during prosecution.

132. Jenatton describes a Bayesian optimization technique that can be applied to “complex black-box functions whose evaluations are expensive.” Ex. 1007, *Abstract*. Jenatton focuses on training complex machine learning models but recognizes that the described technique could be used in other applications where Bayesian optimization has been used in the past. Jenatton describes a technique that leverages a known dependency structure in an output domain to explore a search space characterizing the output domain more efficiently. *Id.* In other words, the described method decreases the number of candidate outputs that must be evaluated, using the tree-structure to inform decisions about what outputs should be evaluated in depth. Ex. 1007, 1. Jenatton proposes to generate a surrogate model combining knowledge from an independent Gaussian process with a linear model that encodes a tree-based dependency structure that can transfer information between overlapping decision sequences.

133. Jenatton states that the described method is useful for searching for hyperparameter combinations. For example, where different hyperparameter choices share some common characteristic (e.g., two logistic regression penalties may share a single learning rate; or hyperparameters of a deep neural network may be hierarchical in nature, where different layer sizes, activation function choices, or

dropout fraction may depend on the number of layers in the network), Jenatton's method can increase the efficiency of searching for useful hyperparameter combinations. Ex. 1007, 2.

134. Jenatton describes that to leverage conditional dependencies between hyperparameters, a tree-structured surrogate model is built, with separate Gaussian Processes (GPs) at the leaf nodes, and random linear (or constant) functions at the inner nodes. Ex. 1007, 2. This structure enables information transfer between leaves that share nodes on their respective paths, thereby increasing efficiency when searching the output space. Second, Jenatton describes an acquisition function which exploits the tree structure and relies on expected improvement functions. *Id.* The function focuses on the most promising leaf node to score, then optimizes over all possible anchor points in this region of this most promising leaf node. *Id.*

135. Jenatton explains that acquisition models can take one of a handful of known structures, all defined in terms of expectations over a surrogate model posterior. Those choices include Thompson sampling, probability of improvement (PI), and expected improvement (EI). Ex. 1007, 4. Jenatton specifically focuses on EI, and demonstrates that organizing hyperparameter decisions using a tree-structure speeds-up selection of optimal hyperparameter combinations, despite the fact that the acquisition function does not explore all possibilities (i.e., does not explore all leafs of the decision tree). Ex. 1007, 6-8.

D. Lorenz, Romy, et al. “Stopping Criteria for Boosting Automatic Experimental Design Using Real-Time fMRI with Bayesian Optimization” (Ex. 1008)

136. Lorenz is a conference paper published on November 24, 2015, as part of the event titled *5th NIPS Workshop on Machine Learning and Interpretation in Neuroimaging: Beyond the Scanner – Proceedings* that was held on December 11-12, 2015. Ex. 1011, 41-44. I understand that Lorenz has been attached as Ex. 1008. *See* Ex. 1008. Lorenz was not cited and was not considered during prosecution.

137. Lorenz describes how typical Bayesian optimization stopping criteria can be applied outside of the context of training a machine learning model. Specifically, Lorenz recognizes that Bayesian optimization techniques originally developed for machine learning can be applied to design experiments to evoke a desired target brain pattern. Ex. 1008, *Abstract*. Lorenz states that functional magnetic resonance imaging (fMRI) experiments are high-cost and therefore experiment design must have an accurate and reliable stopping criteria to end experiments that are unlikely to achieve desirable results. *Id.*

138. Lorenz points out that most research into Bayesian optimization has focused on “online learning,” while citing to an example publication that describes the use of Bayesian optimization in machine learning. Ex. 1008, 4 (*citing* Hoffman, M.D., Shahriari, B., de Freitas, N.: On correlation and budget constraints in model-based bandit optimization with application to automatic machine learning. Proc. of

the 17th Int Conf on Artificial Intelligence and Statistics (AISTATS) (2014)). Lorenz describes a combination of “two popular acquisition functions” to establish a stopping criterion. *Id.* Those two acquisition functions are “expected improvement (EI)” and “probability of improvement (PI).” Ex. 1008, 4-5. EI acquisition functions look to same candidate stimuli that have a high posterior variance in the Gaussian Process (GP), thereby favoring exploitation in the face of uncertainty. PI acquisition functions provide a probability of observing an improvement in a latent objective function. *Id.*

139. According to Lorenz, the EI function has the following form:

$$EI(x) = (m(x) - f(x^+))\Phi(z) + \sigma(x)\phi(z)$$

140. In the above, Φ and ϕ are, respectively, the cumulative and probability density functions for a standard Gaussian random variable. z is equivalent to the PI function, shown below. $m(x)$ is the mean function, $\sigma(x)$ is the variance function, and $f(x^+)$ is the maximum observed activation function.

141. And Lorenz defines the PI function as:

$$z = \frac{m(x) - f(x^+)}{\sigma(x)}$$

142. The included functions are a subset of those in the EI function.

143. Lorenz describes a method of combining these two acquisition functions, by calculating a PI of each new candidate proposed by an EI function and

terminating the Bayesian optimization algorithm when the PI of the next new candidate point is less than a threshold significance level. *Id.* Said differently, when the next proposed combination of experimental parameters has a low probability of providing improved results (less than the threshold significance level), then the optimization algorithm is terminated. *Id.*

IX. The Prior Art Renders Obvious the Challenged Claims of the '968 Patent

144. The '968 patent includes 12 claims, of which claims 1 and 8 are independent. Claims 1 and 8 are similar – Claim 1 is directed to a method of training a neural network (NN), and claim 8 is directed to a system of training a neural network (NN). The limitations in each of these claims are identical.

A. Ground 1: Baker and Lorenz Render Obvious Claims 1 and 6-8 of the '968 patent.

145. Baker and Lorenz render obvious claims 1 and 6-8 of the '968 patent.

Motivation to Combine Baker and Lorenz

146. There are numerous reasons a person of ordinary skill in the art would have combined Baker and Lorenz.

147. Baker is generally directed to the field of machine learning and optimization of the same through automated design of experiments and/or test runs for selecting hyperparameters. Ex. 1005, Abstract. That is, Baker describes a learning coach machine learning model that assists in training a student machine learning model, such as a NN. *See e.g.*, Ex. 1005, *abstract*, ¶¶[0004], [0007], [0025]-

[0029]. Baker's learning coach controls hyperparameters and "makes structural modifications to the student learning system to optimize some combination of the cost of errors and the cost of performing the computation." Ex. 1005, ¶[0025]. According to Baker, "by monitoring a student machine learning system, the coach machine learning system can learn (through machine learning techniques) 'hyperparameters' for the student machine learning system that control the machine learning process for the student learning system." *Id.* The coach machine learning system can help build and train student machine learning systems having the same or a different type of machine learning architecture. Ex. 1005, ¶[0005]. The training process uses the error cost function to proceed iteratively through "minibatches" of training data, and updates trainable parameters for each minibatch. *Id.* The training process proceeds "until a stopping criteria is reached." Ex. 1005, ¶[0031].

148. To the extent it is argued that Baker does not explicitly list what stopping criteria may be used, a POSITA would understand that Baker is referring to well-known stopping criteria that can be used to stop a training process. Lorenz describes multiple well-known stopping criteria, including those discussed in "Bayesian optimization literature" referencing machine learning based literature. Ex. 1008, 4 n[8]. Lorenz is generally directed to optimization for tuning parameters of an experimental test run. Ex. 1008, *Abstract*. While Lorenz discusses these experimental values in reference to an fMRI testing method, Lorenz specifically

points to several reference papers all relating to optimization methods used in machine learning training (*see e.g., id.*, 7-8 (see titles of references 6, 8, 11, as examples)). A POSITA looking for literature relating to experimental stopping criteria, particularly relating to Bayesian Optimization methodologies, would look at publications discussing the general principles of well-known experimental stopping criteria, even if outside of the intended application. For example, a POSITA looking for machine-learning training optimization approaches would look to Lorenz, even though it is specifically directed to fMRI experimental design, at least because Lorenz extensively discusses that the same early stopping criteria have been used extensively in ML training algorithms.

149. Lorenz discusses two well-known and reliable stopping criteria, as well as ways of combining those two stopping criteria. As Lorenz states, these stopping criteria have been discussed at length in ML training-related literature. Ex. 1008, 7-8. Lorenz specifically discusses Probability of Improvement as one example stopping criteria, that may be combined with Expected Improvement (a second well-known stopping criteria).

150. In view of the explicit teaching in Baker of optimizing the cost of performing the computation of the learning process for the student machine learning model (*see* Ex. 1005, ¶¶[0007], [0044]), a person of ordinary skill in the art would have turned to Lorenz which expressly discloses providing reliable stopping criteria

to end experiments that are unlikely to achieve desirable results, based on a metric indicative of the performance of the model. Ex. 1008, *Abstract*, 4-5.

151. A person of ordinary skill in the art would have combined these references to provide the system of Baker with a known stopping criterion, such as by using the probability of improvement as disclosed in Lorenz. *See id.*, 3-5, n. [6], [8], and [11].

152. The combination of Baker and Lorenz is obvious because the combination relies on a known solution that has been implemented in machine learning and other areas of experimental research. Lorenz discloses using a probability of improvement of a calculated metric as a known stopping criteria. Lorenz describes that this stopping criteria can improve machine learning training systems, as discussed in Baker in the same way the solution would improve other known systems—saving on resources by providing a stopping criteria for training a model.

153. A person of ordinary skill in the art would have had a reasonable expectation of success in the combination of Baker and Lorenz based on the explicit teachings of the references and the knowledge of the person of ordinary skill in the art, including those concepts discussed in Section VI herein. Baker explains that a training process continues “until a stopping criterion is reached.” Ex. 1005, ¶[0031]. Lorenz explains that a probability of improvement calculation is a known stopping

criteria, demonstrates that the use of probability of improvement criteria are usable in machine learning training methods, and provides details about how to use the probability of improvement calculation as a stopping criteria, when the calculated probability of improvement is less than a threshold significance level. See e.g., Ex. 1008, 4-5.

154. A person of ordinary skill in the art reviewing Baker would have recognized that training the student machine learning system remains a highly iterative process, even with the use of a learning coach machine learning model that considers model parameters (e.g., hyperparameters). Ex. 1005, ¶ [0033]. While the learning coach machine learning system may propose optimal hyperparameter values for the student machine learning system to minimize the cost of errors of the student machine learning system, those proposed values must still be tested by executing a test run on a large amount of training data. *Id.* Testing the proposed “optimal” hyperparameter values only proves that those values are, in fact, optimal, if other hyperparameter values are also tested and a training loss is calculated (e.g., other hyperparameter values proposed by the learning coach machine learning system), so that the learning process for the student machine learning system is iterative. To minimize costs of testing hyperparameter values that are unlikely to yield optimal performance and minimize the cost of errors by the student machine learning system, a POSITA would have sought to use a known early stopping criteria

to stop non-optimal training runs. Lorenz discloses a well-known early stopping criteria based on probability of improvement of a performance metric, and Lorenz discloses that this process is well-known and used for machine learning training. Ex. 1008, 4-5. To end a training run that is unlikely to result in parameters for the student machine learning model that minimizes cost of errors, a POSITA would implement the probability of improvement criteria to determine a probability that a particular test run is likely to yield an improvement in the cost of errors, using the parameters of the current training run, the calculated cost of errors of the current training run, and the learning coach machine learning model to determine when to early stop the test run. This is a known stopping criteria, and it would minimize the cost of performing the computations of the learning process (Ex. 1005, ¶[0007]), while enabling the learning coach to determine an optimal set of hyperparameters that minimizes the cost of errors. A POSITA would have had a reasonable likelihood of successfully implementing this combination, because Baker explains that a stopping criterion is used to end training, and Lorenz describes the early stopping algorithm in detail, including calculating a mean and variance of the performance metric. A POSITA would be able to implement the algorithm as described in Lorenz, while considering the probability of improvement in cost of errors (i.e., by calculating a mean cost of errors and a variance in the cost of errors), as the cost of errors are already “observed” performance metrics by Baker’s learning coach.

***Claim 1[pre]: A method of training a neural network (NN),
the method comprising:***

155. To the extent element 1[pre] is limiting, Baker renders obvious element 1[pre]. Baker describes a machine learning (ML) system includes a student ML system and a learning coach ML system that “learns to make an enhancement to the student ML system or to its learning process, such as updated hyperparameter or a network structural change, based on the training of the student ML system.” Ex. 1005, *Abstract*. In short, Baker describes a method of training the “student” machine learning system to minimize the error rate or some other measure of the cost of the errors. Ex. 1005, ¶¶[0004], [0007], [0025]. The “student” machine learning system can be “a large, deep feed-forward neural network (a DNN),” although Baker explains that the student can be other machine learning structures, including “support vector machines (SVMs), random forests, CART analysis, hidden stochastic process models, dynamic Bayesian networks, Boltzmann machines, ensembles of any of these and even heterogenous ensembles.” Ex. 1005, ¶¶[0029], [0030], [0050].

156. Baker explains that the learning process of the student machine learning system may be controlled by hyperparameters. Ex. 1005, ¶[0030]. Baker describes an example “well-known” training method as “an iterative process of stochastic gradient descent on a log-likelihood error cost function, with a softmax classification for the output nodes, the gradients for each training example being estimated by back propagation of the partial derivate of the error cost function, the estimates being

accumulated over minibatches, with one update of the parameters for each minibatch.” *Id.* The “iterative process” of training the student machine learning model is performed to find values of “parameters” of the student machine learning model to minimize some measure of the cost of errors of the student machine learning model. Ex. 1005, ¶¶[0027]. There may be millions or billions of possible parameter combinations for a single student machine learning model, and so the learning process can be very time consuming and expensive. *Id.* Therefore, the described “well-known” learning process is performed “until a stopping criterion is reached.” Ex. 1005, ¶¶[0031], [0037].

157. Baker explains that hyperparameters control this learning process, such as M (the size of the minibatch), η (the learning rate), λ (the regularization parameter), and μ (the momentum parameter). Ex. 1005, ¶¶[0032]-[0033]. The “learning coach” machine learning model described in Baker is trained to optimize selection of those hyperparameters (or of other characteristics of the learning process for the student machine learning model or the structure of the student machine learning model) to “optimize some combination of the cost of errors and the cost of performing the computation.” Ex. 1005, ¶¶[0007], [0032]-[0033].

158. The ’968 patent’s discussion of “training a neural network” is consistent with Baker’s discussion of the training process for the student machine learning model. As mentioned above, the ’968 patent discusses that training

proceeds over a series of training intervals, such as “batches” and/or “epochs” and one or more training losses can be calculated (e.g., representing a difference or inconsistency between the value or values output from the network and the correct value/values) for each training interval. Ex. 1001, 2:46-66. The training process itself proceeds in accordance with hyperparameters chosen for the training process, and the training process includes “iteratively adjusting model parameters to optimize an objective function, also known as a loss function.” *Id.*, 1:39-67, 5:19-51, 13:10-35, 15:41-53.

Claim 1[a]: over a series of NN training epochs, where in each epoch the NN undergoes training and a loss is computed:

159. Baker renders obvious element 1[a].

160. Element 1[a] recites a typical training process for a NN. The '968 patent describes a “training epoch” as “a set of batches representing the entire training set; thus an epoch may represent a number of forward/backward passes, each resulting in a weight adjustment.” Ex. 1001, 2:59-65. Each “batch may be a set of multiple training data presented to a NN which results in a number of outputs.” *Id.* The training process may iterate over multiple training epochs, and the trainable weights of the NN may be updated and a loss may be computed after each epoch.

161. The '968 patent also describes a “loss or function may define a deviation from an expected or correct output value for a given input, during learning or training.” *Id.*, 2:46-57. The '968 patent gives examples of loss functions,

including “a negative log-likelihood or residual sum of squares.” *Id*

162. Baker describes the same typical training process for a NN (the student machine learning model), which can be optimized using a secondary machine learning model—the learning coach. Baker explains that the student machine learning model undergoes learning (also known as “training;” *see* Ex. 1005, ¶[0008]) over a series of minibatches. Ex. 1005, ¶[0039]. Baker explains that the learning coach observes “the results of the forward activation of the deep neural network and the backpropagation of the partial derivatives” for each example in a minibatch *and* as summary statistics for each update in the minibatch gradient descent. *Id*. The learning coach considers the effect of a particular set of hyperparameters on the resulting student machine learning model parameters (connection weights in a neural network) and the future learning performance of the student learning system. Ex. 1005, ¶[0040]. As mentioned, Baker explains that the student learning system has an objective to “minimize the error rate or some other measure of the cost of the errors” and the learning coach can have an objective to “optimize some combination of the cost of errors and the cost of performing the computation.” Ex. 1005, ¶[0007]. Given Baker’s mention of an “objective” of the student learning system to “minimize the error rate or some other measure of the cost of the errors,” the “performance” of the student learning system refers to the student learning system’s progress toward these objectives, i.e., the progress of the student learning system to minimize the

error rate of the student learning system as it undergoes the learning process.¹

163. Completion of training through a series of Baker’s minibatches (which are analogous to batches of training data) is the same as completing a “training epoch” as recited in 1[a]. Training using minibatches, each of which constitute one forward pass and one backpropagation pass, is well-known to those of skill in the art as completing a batch or an “epoch.” *See e.g.*, Ex. 1010, ¶[0013]. Baker explains that the training process is an iterative process and trainable parameters may be updated once for each minibatch, and the training process continues “until a stopping criterion is reached.” Ex. 1005, ¶¶[0030]-[0031]. Iterating through updates to the

¹ The ’968 patent is ambiguous about how many “losses” are calculated during an epoch. Claim element 1[a] recites that “a loss is computed” during an epoch, whereas the specification states that a “loss” is generated when processing each “batch” within a complete training set, and “[a]n epoch may be a set of batches representing the entire training set” so that multiple losses are computed per epoch. A person of ordinary skill in the art would have understood that this description relates to the general process of performing forward pass and backpropagation of multiple batches, which can be completed one or more times during a training process. *See* Ex. 1001, 2:58-66. A person of ordinary skill in the art would thus have understood that Baker discloses this same training process.

trainable parameters after training on multiple minibatches until a stopping criterion is reached constitutes training “over a series of NN training epochs,” as recited in element 1[a].

164. Baker uses different terminology, but it describes the concept of calculating a “loss” during training, in the same way that the ’968 patent describes it. Baker describes computing a “log-likelihood error cost function” during training on each minibatch. Ex. 1005, ¶¶[0030]-[0031]. This is the same as the example “negative log-likelihood” loss function used by the ’968 patent for calculating a loss. Ex. 1001, 2:46-57. A person of ordinary skill in the art would recognize that the terms “loss” and a “cost of errors” are synonymous when discussing a NN training process. *See also*, Ex. 1006, 14 (discussing logarithmic formulas for likelihood functions); Ex. 1007, 3.

165. Moreover, a person of ordinary skill in the art would understand that calculating (and minimizing) a “measure of the cost of the errors” is synonymous to calculating a “loss” as discussed in the ’968 patent – both of which describe the deviation of a model’s predictions from a true prediction.

166. Therefore, Baker teaches 1[a].

Claim 1[b.1]: [determining, using] a set of model parameters;

167. I understand that each of claim elements 1[b.1]-1[b.3] are recitations of the factors used for determining a probability of improvement in the loss of the NN,

as recited in claim element 1[c]. As discussed in greater detail below, the “probability of improvement” is a computation used for determining whether some further training process is likely to yield better performance by a neural network, as compared with a current measure of performance (i.e., a “cost of error” also known as a “loss”). The probability of improvement is computed to optimize the training process of a model. Baker renders obvious element 1[b.1].

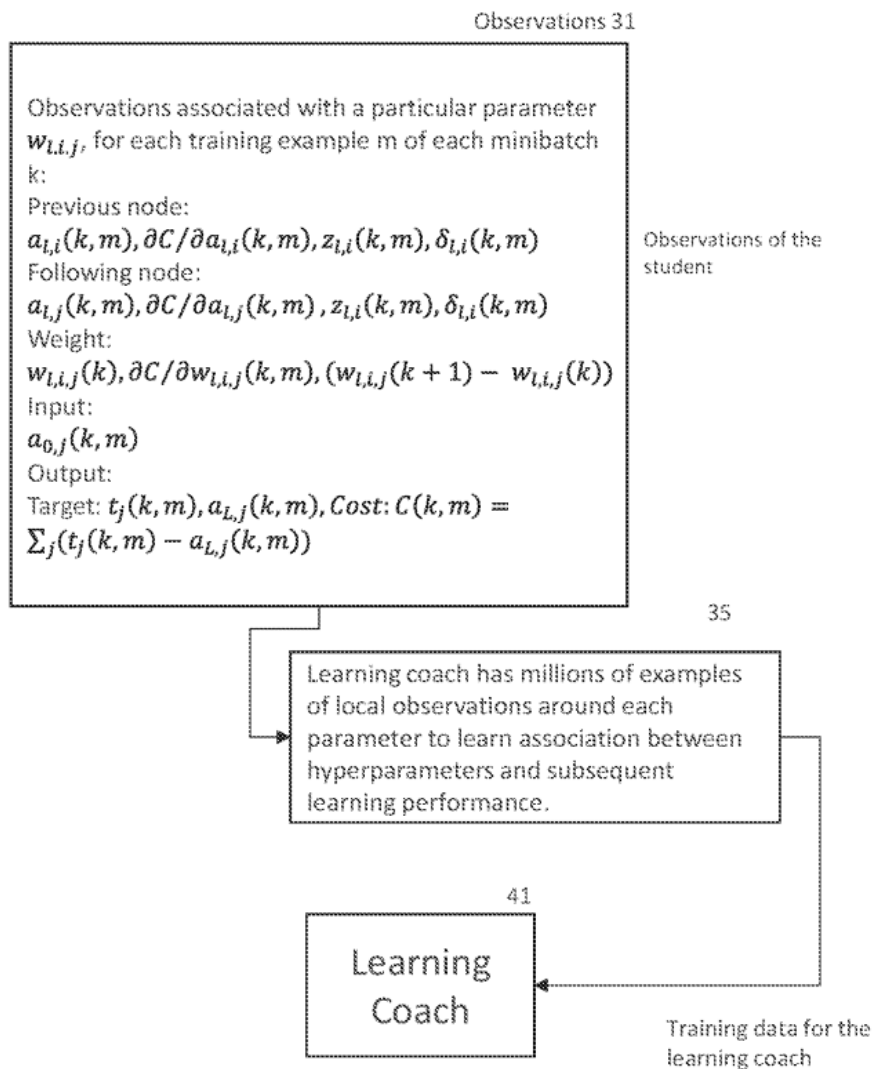
168. The '968 patent describes “parameters” as including “weights of links or functions that link layers of a NN, which may be adjusted during learning or training” *and* “[h]yperparameters may be parameters that cannot be directly optimized as a function of the model’s loss, and may be determined using a part of the data that has not been used for training.” Ex. 1001, 1:39-2:4. The '968 patent explains that hyperparameters “may describe an overall architecture of the NN such as number and structure of layers, number of neurons in each layer or overall, the type of functions neurons use when computing output, and variables which determine how the network is trained, such as the learning rate.” *Id.* Those hyperparameters “may include parameters other than those describing a NN architectures, for example NN training parameters such as batch size, number of training parameters, number of training samples, number of link weights, number of layers, number of examples in a training set, etc. Hyperparameters may include for example, a number of layers, a kernel size for a layer (e.g. 3×3 , 7×7), a batch size

(e.g. 32, 64, 128 or 256), a learning rate (e.g. 0.0001 to 0.1) or other parameters. Hyperparameters may include layer configuration, activation function, or other data.” *Id.*

169. Baker teaches using a set of model parameters (hyperparameters)² to optimize the training process of a model. Baker explains that a learning coach embodied as a machine learning model is used to select hyperparameters that are likely to yield a high-performance student learning model. Ex. 1005, ¶¶[0039]-[0041]. The learning coach “uses” the hyperparameters of a learning process for a student machine learning model by learning how sets of hyperparameter values effect the future learning performance of a student learning system. These hyperparameters are included in training data for the learning coach, as shown in FIG. 3, reproduced below, so that the learning coach “uses” these hyperparameters during training, which impacts prediction outputs of candidate values for future hyperparameter values used to train the student learning system. *Id.* Therefore, Baker renders obvious element 1[b.1].

² I understand from counsel that “parameters” as used in claim 1 can include “hyperparameters” based on the content of claim 7, which is dependent from claim 1 and states “wherein model parameters comprise hyperparameters.”

Fig. 3



Claim 1[b.2]: [determining, using] data describing training loss of the NN; and

170. Like element 1[b.1], I understand that element 1[b.2] recites a second factor used for determining a probability of improvement in the loss of the NN, as

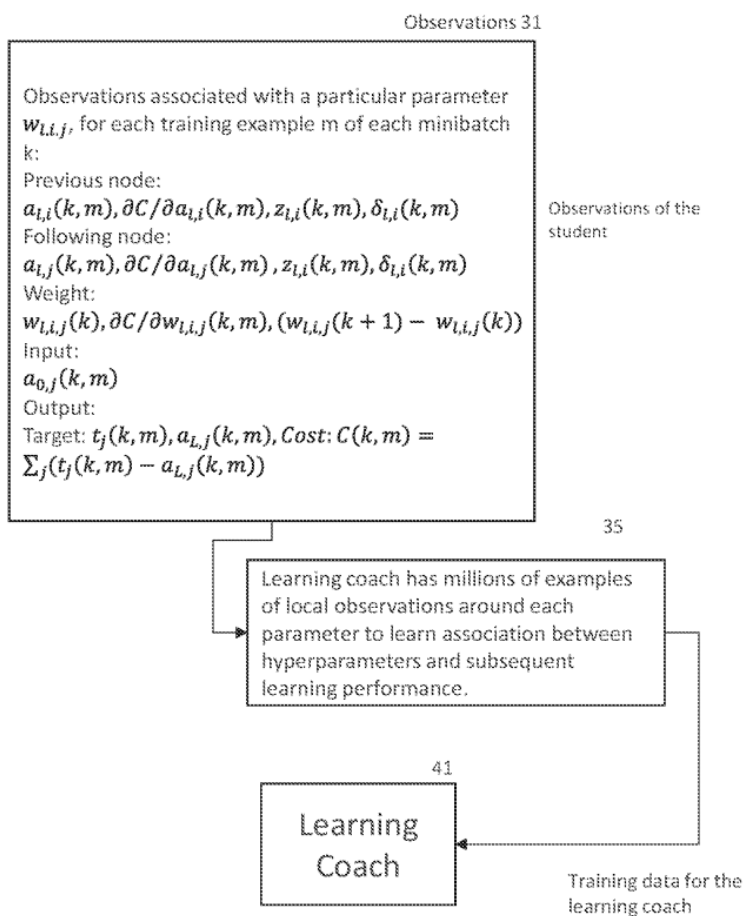
recited in claim element 1[c]. Baker renders obvious element 1[b.2].

171. As mentioned in reference to 1[a], Baker’s “cost of errors” is the same as the “loss” discussed in the ’968 patent.

172. Baker discloses that the learning coach “has millions of training examples from which it can learn to predict the best hyperparameter values for each customized situation.” Ex. 1005, ¶[0040]. The learning coach’s objective for selecting a set of “best hyperparameter values” can include choosing a set of hyperparameter values that optimizes a combination of the cost of errors and the cost of performing a computation during training the student learning model. Ex. 1005, ¶[0007]. A person of ordinary skill in the art would have recognized that the training data for the learning coach would have included a computation of a cost of error function (also known as a “loss” function) to enable the learning coach to predict a set of hyperparameters that optimizes the combination of the cost of errors and cost of performing a computation. Including the computed cost of error in the learning coach’s training data, paired with the hyperparameter values that led to the computed cost of error (and other contextual data), enables the learning coach to identify relationships between the hyperparameter values and the resulting cost of errors. In turn, this enables the learning coach to attempt to identify hyperparameter values that optimize the cost of errors. A POSITA would recognize that a “cost of error function” or more simply an “error cost function” as discussed in Baker is the same

as a “loss” or “loss function” as described in the ’968 Patent. Baker and the ’968 Patent describe identical examples of their respective “error cost function” and “loss function,” respectively. Baker explains that the training process can be “an iterative process of stochastic gradient descent on a log-likelihood error cost function, with a softmax classification for the output nodes, the gradients for each training example being estimated by back propagation of the partial derivative of the error cost function, the estimates being accumulated over minibatches, with one update of the parameters for each minibatch.” Ex. 1005, ¶[0030]. Similarly, the ’968 patent describes an example loss function as “a negative log-likelihood,” and a “loss” may be generated for each batch. Ex. 1001, 2:46-62. These terms define the same, known mathematical calculation to define a difference (or error) between an expected output, and the output actually generated by the NN, using a logarithmic scale. *See also*, Ex. 1006, 14 (discussing logarithmic formulas for likelihood functions); Ex. 1007, 3. Figure 3, reproduced below, shows that a “cost” is calculated for a target output included in the training data. By including this cost of error data in the training data for the learning coach, the learning coach “uses” the cost of error data during training to generate prediction outputs of candidate values for future hyperparameter values based in part on the cost of error data that is included in the training data used to train the student learning system. *Id.* Therefore, Baker teaches element 1[b.2].

Fig. 3



173. The '968 patent provides a similar discussion of using training losses. Training losses are computed during training of a NN (Ex. 1001, 2:46-66, 13:10-23, 15:41-53). The '968 patent may also include data reflecting historical training losses from the current NN and/or other NNs in training data for a model used to facilitate training of the NN. *See id.*, 5:7-18, 5:39-51, 6:7-61, 12:40-13:9, 15:32-40.

Claim 1[b.3]: [determining, using] a model that has been trained using training losses of a plurality of NNs other than the NN;

174. Like elements 1[b.1] and 1[b.2], I understand that element 1[b.3] recites a third factor used for determining a probability of improvement in the loss of the NN, as recited in claim element 1[c]. Baker renders obvious element 1[b.3].

175. Baker's learning coach, like the "model" discussed in the '968 patent, assists in training a NN (e.g., Baker's student machine learning model). *See* Ex. 1001, 5:19-20, 26:23-29; Ex. 1005, ¶[0007], [0025]-[0026].

176. Baker's learning coach is a "model that has been trained using training losses of a plurality of NNs other than the NN." Baker's learning coach is a machine learning system. Ex. 1005, ¶[0025], Figure 3. The learning coach is trained using training data including "millions of training examples from which it can learn to predict the best hyperparameter values for each customized situation." Ex. 1005, ¶[0040]. As mentioned above, those training examples include training losses (also known as "cost of errors"), as shown in Figure 3. By including the cost of error data in the training data for the learning coach, the learning coach is enabled to identify relationships between certain hyperparameter values and the resulting cost of errors (considering other contextual data included in the training data).

177. Moreover, the training data includes training data "from observing the learning process of many different examples of student learning systems 11. That is,

learning coach 41 learns from prior experience of coaching other student learning systems 11, not just from the current example.” Ex. 1005, ¶[0041]. As discussed herein, the “student learning system” may be a DNN. Ex. 1005, ¶[0029]. Those “other student learning systems 11” are “a plurality of NNs other than the NN,” and the training data observed for these “other” student learning systems includes the same training data that was observed for the current student learning system. Ex. 1005, ¶[0041]. Therefore, Baker renders obvious element 1[b.3].

178. The '968 patent also describes models that assist in “choos[ing] hyperparameters of an NN.” Ex. 1001, 5:19-20, 26:23-29. These models are built using data reflecting the performance of past NNs with various hyperparameters, such as “loss function data, or training losses, [that] may be collected, from multiple different NNs (typically different NNs in a variety of different contexts, with a variety of different architectures, performing a variety of different analysis tasks, having been trained for different numbers of epochs), along with hyperparameter and possible other data, e.g. via an experiment or learning management platform, to build or train a predictive model that can estimate an interval in which an evaluation metric is likely to fall.” *Id.*, 6:7-35; *see also id.*, 12:31-13:9, 15:32-40, 18:26-63. This discussion of a “model that has been trained using training losses of a plurality of NNs other than the NN” includes the same features that provide the same functions as Baker’s learning coach.

Claim 1[c]: [determining, using 1[b.1-3]] a probability of improvement in the loss of the NN; and

179. Claim element 1[c] recites determining a probability of improvement in the loss of the NN, using the elements recited in 1[b.1-b.3]. Baker renders obvious 1[c]. However, to the extent it is argued that Baker does not explicitly disclose 1[c], Baker and Lorenz render obvious 1[c].

180. As mentioned above, the “probability of improvement” is a computation used for determining whether some further training process is likely to yield better performance by a neural network. The probability of improvement can be compared with a current measure of performance (i.e., a “cost of error,” also known as a training loss). The probability of improvement is computed to optimize the training process of a model.

181. Baker teaches using the elements in 1[b.1-b.3] to determine optimal hyperparameter values, which involves predicting which hyperparameter values will yield the lowest cost of errors/training loss. Ex. 1005, ¶¶[0007], [0032]-[0033]. Because Baker’s student machine learning model learning process is designed to minimize error or some measure of the cost of errors (Ex. 1005, ¶[0007]), using a learning coach machine learning model (a “model” as recited in Element 1[b.3]) that assigns hyperparameters (“parameters” as recited in Element 1[b.1]) during the learning process) and iterates through the learning process (*Id.*, ¶[0032]) to determine an optimal set of hyperparameters having a best error rate or measure of

cost of errors (a “loss” computed for each test run, as recited in Element 1[b.2]), the learning process “uses” each of these features when training the student machine learning model.

182. Baker explains that the student learning system has an objective to “minimize the error rate or some other measure of the cost of the errors” and the learning coach can have an objective to “optimize some combination of the cost of errors and the cost of performing the computation.” Ex. 1005, ¶[0007]. The “performance” of the student learning system, as determined by the learning coach machine learning model, refers to the student learning system’s progress toward these objectives, i.e., the progress of the student learning system to minimize the error rate of the student learning system as it undergoes the learning process. Baker explains that the learning coach may “model the learning performance of student learning system 11” (Ex 1005, ¶[0032]) as a part of generating the optimal hyperparameters. A person of ordinary skill in the art would understand that modeling the learning performance of student learning system would have been representative of “the error rate or some other measure of the cost of errors.” Ex. 1005, ¶[0007]. Baker teaches using the set of model parameters to optimize the training process of a model. Baker explains that a learning coach embodied as a machine learning model is used to select hyperparameters that are likely to yield a high-performance student learning model. Ex. 1005, ¶¶[0039]-[0041]. The cost/loss

values in Baker are determined after each minibatch or epoch. Ex. 1005, ¶[0031].

183. Moreover, Baker explains that training a NN with a selected set of hyperparameters continues “until a stopping criterion is reached.” Ex. 1005, ¶[0031]. A person of ordinary skill in the art would have recognized that Baker’s learning coach “can learn to predict the best hyperparameter values for each customized situation” for training a student machine learning model (Ex. 1005, ¶[0040]), and so any “stopping criterion” would have been implemented by the learning coach as a part of the prediction of whether a hyperparameter set is the “best” for training the student machine learning model based on the calculated “error or some other measure of the cost of errors,” as a performance indicator to stop training when the performance of the student machine learning model meets a predicted best performance for the hyperparameter values used for training. A person of ordinary skill in the art would have understood that “a stopping criterion” would have been a well-known stopping criterion, such as a calculation of a probability of improvement of the cost of errors and the stopping criterion can be executed by the learning coach (e.g., using the predicted best hyperparameter values, as determined by the learning coach) to “optimize some combination of the cost of errors and the cost of performing the computation.” Ex. 1005, ¶[0007]. Baker renders obvious 1[c].

184. To the extent it is argued that Baker does not explicitly disclose 1[c], a person of ordinary skill in the art would have turned to Lorenz for examples of

known “stopping criteria” that could be implemented by Baker’s training process. Lorenz discloses one of those known stopping criteria could be a probability of improvement criteria. Lorenz describes that the probability of improvement criteria can be used alone or in combination with an expected loss stopping criterion. A POSITA would have modified Baker to include one of the stopping criteria discussed in Lorenz.

185. Lorenz describes two well-known Bayesian optimization stopping criteria for early stopping experiments. Lorenz states that most research into Bayesian optimization has focused on “online learning,” while citing to an example publication that describes the use of Bayesian optimization in machine learning. Ex. 1008, 4. Lorenz explains that Bayesian optimization criteria used for machine learning processes are also useful for experimental design in other contexts. *Id.*, *abstract* (discussing Bayesian optimization for fMRI experiment design). Lorenz describes a combination of “two popular acquisition functions” to establish a stopping criterion. Ex. 1008, 4. Those two acquisition functions are “expected improvement (EI)” and “probability of improvement (PI)” of a performance metric. Ex. 1008, 4-5. Each of these acquisition functions are well known, and use equations to establish what constitutes an “expected improvement” value or a “probability of improvement” value. EI acquisition functions look to same candidate stimuli that have a high posterior variance in the Gaussian Process (GP), thereby favoring

exploitation in the face of uncertainty. PI acquisition functions provide a probability of observing an improvement in a latent objective function. *Id.* The basis for each of these equations, i.e., what the “expected improvement” or “probability of improvement” is compared against (the answer to the question “the improvement over what value?”) is provided as a part of the equations. A person of ordinary skill in the art would have recognized that these equations can be used to determine whether a particular test run has a probability of providing an improvement over a historically established “best” test run (e.g., will the current test run provide performance better than all other, historical test runs?), because these equations consider data generated during all prior runs. Specifically, Lorenz describes a method of combining these two acquisition functions to identify whether a test run is likely to provide a best performance so far, by calculating a PI of each new candidate proposed by an EI function and terminating the Bayesian optimization algorithm when the PI of the next new candidate point is less than a threshold significance level. *Id.* Said differently, when the next proposed combination of experimental parameters has a low probability of providing improved results (less than the threshold significance level), then the optimization algorithm is terminated. *Id.* The ’968 patent also describes probability of improvement functions using mean predicted loss and variance data to determine a likelihood that the NN is unlikely to

improve over a historically determined “best” loss value. Ex. 1001, 5:7-18, 6:62-65, 12:50-13:454, 14:62-67, 16:29-39, Table 3.

186. Baker explains that the student learning system has an objective to “minimize the error rate or some other measure of the cost of the errors” and the learning coach can have an objective to “optimize some combination of the cost of errors and the cost of performing the computation.” Ex. 1005, ¶[0007]. The “performance” of the student learning system refers to the student learning system’s progress toward these objectives, i.e., the progress of the student learning system to minimize the error rate of the student learning system as it undergoes the learning process. Baker explains that the learning coach may “model the learning performance of student learning system 11,” using “millions of training examples from which it can learn to predict the best hyperparameter values for each customized situation.”. Ex. 1005, ¶¶[0007], [0032], [0040]. A person of ordinary skill in the art would have understood that modeling the learning performance of student learning system to predict the best hyperparameter values would have been representative of “the error rate or some other measure of the cost of errors” for possible hyperparameter value combinations. *See id.*, ¶[0007]. Baker teaches using the set of model parameters to optimize the training process of a model. Baker explains that a learning coach embodied as a machine learning model is used to select hyperparameters that are likely to yield a high-performance student learning

model. Ex. 1005, ¶¶[0039]-[0041]. The cost/loss values in Baker are determined after each minibatch or epoch. Ex. 1005, ¶[0031].

187. A person of ordinary skill in the art would have been motivated to combine Baker and Lorenz for the reasons set forth in the section herein titled *Motivation to Combine Baker and Lorenz*. See also Section VI (*supra*). Because Baker uses a cost of errors as a performance metric for the learning coach machine learning model to optimize during training of the student machine learning model, and because Baker uses parameters, a calculated training loss, and a model (the learning coach) for optimizing some measure of the cost of errors, a person of ordinary skill in the art would have combined Baker and Lorenz to calculate a probability of improvement (PI) in a cost of errors, to be used as a well-known stopping criterion. Moreover, as a result of the combination of Baker and Lorenz, where Baker's learning coach computes and uses Lorenz's probability of improvement when optimizing some measure of the cost of errors of the student machine learning system, the determination of the probability of improvement would "use" the parameters, loss calculation, and learning coach model during determination of the probability of improvement.

188. Baker and Lorenz render obvious 1[c].

Claim 1[d]: if the probability is less than a threshold, or a wait value is greater than a wait threshold, stopping training.

189. Baker and Lorenz renders obvious 1[d].

190. Baker describes a “well-known training process” in which a set of operations is performed “until a stopping criterion is reached.” Ex. 1005, ¶[0031]. To be clear, Baker describes training and then stopping training once a stopping criterion is reached. *Id.*; *see also* Ex. 1005, ¶[0037] (“[d]o until a stopping criterion is reached.”). By way of further example, Baker describes “training of the feature nodes and clustering [that] is done iteratively.” Ex. 1005, ¶[0095]. Baker explains that “[t]his iterative process is repeated until...some other stopping criterion is met.” *Id.* And by way of another example, Baker describes an iterative process in which a shared validation data set is used to “retrain [a] preliminary classifier” and “the process is repeated until convergence or some other stopping criterion is met.” Ex. 1005, ¶[0145]. *See also* Section VI (*supra*).

191. As discussed in reference to element 1[c], a person of ordinary skill in the art would have used a well-known stopping criterion for Baker’s training process, such as a stopping criterion based on the probability of improvement in a cost of errors. As discussed in Lorenz, using probability of improvement as a stopping criterion would have included comparing a calculated probability of improvement of the cost of errors for the tested model against a probability threshold, and stopping the training if the probability of improving the cost of errors is below the probability threshold. Lorenz’s probability of improvement stopping criteria is the same as the stopping criteria described in the ’968 patent, which

explains “systems and methods that may stop training of a NN if it is determined that likelihood of improvement in loss over a best loss seen is below a threshold.” Ex. 1001, 5:7-10. “Loss data may be computed and/or returned periodically, for example after each forward and backward pass, or in or after each epoch.” *Id.*, 13:10-23. The probability of improvement over a best loss seen for this NN is then calculated, for example, using a mean expected best loss for the NN and a variance of that mean.” *Id.*, 13:36-54.

192. A person of ordinary skill in the art would have combined Baker and Lorenz for the reasons set forth in the section herein titled *Motivation to Combine Baker and Lorenz*. See also Section VI (*supra*). Therefore, Baker and Lorenz renders obvious 1[d].

Claim 6: The method of claim 1, wherein determining an expected probability of improvement comprises determining a mean expected training loss and a variance.

193. Baker and Lorenz render obvious claim 6.

194. As discussed in reference to element 1[c] and 1[d], Baker and Lorenz teach calculating a probability of improvement in the loss of the NN using the same concepts of a mean expected training loss and a variance. *Supra*; Ex. 1001, 5:7-10, 13:10-23, 13:36-54. Lorenz further explains that a probability of improvement function takes the following form:

$$z = \frac{m(x) - f(x^+)}{\sigma(x)}$$

195. In the above, $f(x^+)$ is a function describing a maximum observed activation, $m(x)$ is a mean, and $\sigma(x)$ is a variance function. Ex. 1008, 4-5. Because Lorenz's explanation of calculating a probability of improvement includes determining a mean and variance, a person of ordinary skill in the art would have understood that applying Lorenz's disclosed probability of improvement function to determining a probability of improving a training loss, as taught by Baker and Lorenz, would have included calculating a mean training loss and a variance. Therefore, Baker and Lorenz render obvious claim 6.

Claim 7: The method of claim 1, wherein model parameters comprise hyperparameters.

196. Baker renders obvious claim 7.

197. As discussed in reference to element 1[b.1], the '968 patent describes "hyperparameters" as a type of "parameters." Ex. 1001, 1:39-2:21.

198. Baker teaches using hyperparameters for a model to optimize the training process of the model. Baker explains that a learning coach embodied as a machine learning model is used to select hyperparameters that are likely to yield a high-performance student learning model. Ex. 1005, ¶¶[0039]-[0041]. The learning coach "uses" the hyperparameters of a learning process for a student machine learning model by learning how sets of hyperparameter values effect the future learning performance of a student learning system. *Id.* In turn, these hyperparameters are used for calculating a probability of improvement in the loss of the student

machine learning model (NN), as discussed in reference to element 1[c], to ensure that the learning process does not run redundant tests and to teach which sets of hyperparameters yielded the best performance (e.g., the best training loss). Moreover, these hyperparameters are included in training data for the learning coach, as shown in FIG. 3, reproduced above, so that the learning coach “uses” these hyperparameters during training, which impacts prediction outputs of candidate values for future hyperparameter values used to train the student learning system. *Id.* Therefore, Baker renders obvious claim 7.

Claim 8[pre]: A system of training a neural network (NN), the system comprising:

199. See 1[pre] *supra*.

Claim 8[a]: a memory; and

200. Baker teaches element 8[a].

201. Baker states that the “student learning system 11 and learning coach 41 can be implemented with computer hardware or software.” Ex. 1005, ¶[0046]. The systems can be a part of an integrated computer system that has multiple processing CPU cores. *Id.* One set of processing cores could execute program instructions for the student learning system and another set of processing cores could execute program instructions for the learning coach. *Id.* The processing instructions can be stored in a computer memory that is accessible by the processing cores. *Id.* The learning coach is a part of the system to train the student machine learning system

(NN), and so Baker's system for training the student machine learning model comprises a memory, as discussed at least in paragraph [0046].

Claim 8[b]: a processor configured to:

202. Baker renders obvious element 8[b].

203. As discussed in reference to element 8[a], Baker teaches that the system for training the student machine learning system (NN) comprises one or more processors, in the form of processing cores that can execute the steps discussed herein. Ex. 1005, ¶[0046]

Claim 8[c]: over a series of NN training epochs, where in each epoch the NN undergoes training and a loss is computed:

204. See 1[a] *supra*.

Claim 8[d.1]: [determine, using:] a set of model parameters;

205. See 1[b.1] *supra*.

Claim 8[d.2]: [determine, using] data describing training loss of the NN; and

206. See 1[b.2] *supra*.

Claim 8[d.3]: [determine, using] a model that has been trained using training losses of a plurality of NNs other than the NN;

207. See 1[b.3] *supra*.

Claim 8[e]: [determine, using 8[d.1-d.3]] a probability of improvement in the loss of the NN; and

208. See 1[c] *supra*.

Claim 8[f]: if the probability is less than a threshold, or a wait value is greater than a wait threshold, determine to stop training.

209. See 1[d] *supra*.

B. Ground 2: Baker, Lorenz, and Shridhar Render Obvious Claims 2-4 and 9-11 of the '968 patent.

210. Baker, Lorenz, and Shridhar render obvious claims 2-4 and 9-11 of the '968 patent.

Motivation to Combine Baker, Lorenz, and Shridhar

211. There are numerous reasons a person of ordinary skill in the art would have combined Baker, Lorenz, and Shridhar, in addition to the reasons set forth above for a person of ordinary skill in the art to have combined Baker and Lorenz.

212. A person of ordinary skill in the art would have had a reasonable expectation of success in the combination of the references based on the explicit teachings of the references and the knowledge of the person of ordinary skill in the art, including those concepts discussed in Section VI herein.

213. Baker describes a learning coach machine learning model that assists in training a student machine learning model, such as a NN. *See e.g.*, Ex. 1005, *abstract*, ¶¶[0004], [0007], [0025]-[0029]. Baker's learning coach controls hyperparameters and "makes structural modifications to the student learning system to optimize some combination of the cost of errors and the cost of performing the computation." Ex. 1005, ¶[0025]. Baker's learning coach uses the error cost

function, specifically a log-likelihood error cost function calculated during a training process to determine the performance of the student machine learning model. Ex. 1005, ¶[0030]. The training process uses the error cost function to proceed iteratively through “minibatches” of training data, and updates trainable parameters for each minibatch. *Id.* The training process proceeds “until a stopping criteria is reached.” Ex. 1005, ¶[0031].

214. A person of ordinary skill in the art would have used a known stopping criteria, such as those disclosed in Lorenz and Shridhar, to stop the training process disclosed in Baker.

215. In view of the teaching in Baker of optimizing the cost of errors and the cost of performing the computation of the learning process, a person of ordinary skill in the art would have known to use multiple stopping criteria to balance the cost of errors and the cost of performing the computation of training. *See* Ex. 1005, ¶¶[0007], [0032]-[0033]. For example, comparing a probability of improvement against a threshold would demonstrate whether the cost of errors (loss) has been optimized for the student machine learning system, and using a “wait threshold” (i.e., observing some characteristic of the training process until it does not improve for some defined period of time and/or number of iterations) would demonstrate whether the cost of performing the computation is yielding an improvement in the performance of the student machine learning system.

216. A person of ordinary skill in the art would have combined Baker, Lorenz, and Shridhar, such as by using stopping criteria proposed by Lorenz and Shridhar to improve the cost of computation of Baker's training system.

217. The combination of Baker, Lorenz, and Shridhar would have been obvious because Baker explains that training proceeds "until a stopping criterion is reached," and Baker's learning coach seeks to optimize a cost of errors (as could be determined using the probability of improvement stopping criterion described in Lorenz) with a cost of performing the calculation of the learning process (as could be determined using the wait threshold described in Shridhar). The combination relies on known solutions. Lorenz discloses using a probability of improvement of a calculated metric as a known stopping criteria, and Shridhar discloses using a wait threshold as another example known stopping criteria. These known stopping criteria can improve machine learning training systems, as discussed in Baker in the same way the solution would improve other known systems—saving on resources by providing an early stopping criteria for training a model based on performance metrics (cost of errors) that are already observed.

218. A person of ordinary skill in the art would have had a reasonable expectation of success in combining Baker, Lorenz, and Shridhar based on the explicit teachings of the references and the knowledge of the person of ordinary skill in the art, including those concepts discussed in Section VI herein. Baker explains

that a training process continues “until a stopping criterion is reached.” Ex. 1005, ¶[0031]. Baker’s learning coach could implement the multiple stopping criteria to “optimize some combination of the cost of errors and the cost of performing the computation,” where the stopping criteria address the cost of errors and the cost of performing the computation, respectively. Ex. 1005, ¶[0007]. Lorenz and Shridhar describe known stopping criteria (comparing a probability of improvement against a threshold and using a wait value to determine if a performance characteristic has not changed over multiple training intervals) and provide details about how to use the respective stopping criteria.

219. A person of ordinary skill in the art had good reason to pursue the known options within his or her technical grasp such that it would lead to anticipated success—a reduction in computation costs associated with training a neural network and optimizing hyperparameters by eliminating training iterations on sub-optimal hyperparameter values. Providing multiple early stopping criteria would minimize the number of suboptimal training iterations that proceed to completion. Accordingly, a person of ordinary skill in the art had a reasonable expectation of success in combining Baker, Lorenz, and Shridhar.

Claim 2: The method of claim 1, comprising if the probability is not less than a threshold and a wait value is greater than a wait threshold, continuing training.

220. Baker, Lorenz, and Shridhar render obvious claim 2. As discussed in

reference to elements 1[c]-1[d], Baker and Lorenz teach using a probability of improvement threshold as a stopping criterion to determine whether to continue or terminate training based on a probability that the cost of error calculation will improve for Baker's student machine learning model. Ex. 1005, ¶¶[0031], [0037], [0095]; Ex. 1008, 4. This stopping criterion is provided so that unnecessary training is terminated, for example, where the probability of improvement value indicates that it is unlikely that further training will yield an improvement in performance of the model (e.g., based on cost of error computations). Ex. 1008, 4.

221. Baker explains that the learning coach is trained “to optimize some combination of the cost of errors and the cost of performing the computation.” Ex. 1005, ¶[0007].

222. A person of ordinary skill in the art would recognize that using a probability of improvement-based stopping criteria, as discussed in reference to 1[c]-1[d], would be relevant for optimizing the cost of errors. A person of ordinary skill in the art would have further recognized that multiple stopping criteria, including the well-known stopping criteria described in Shridhar, to optimize “the cost of errors *and* the cost of performing the computation,” as discussed in Baker.

223. For example, Shridhar discloses using early stopping based on multiple criteria: a validation accuracy volatility and a wait threshold “when validation accuracy remains unchanged for 5 epochs.” Ex. 1006, 22. Shridhar's stopping

criteria compares the current number of epochs for which validation accuracy remains unchanged (the “wait value”) against a “wait threshold” of 5 epochs. Using a “wait value” and threshold, as discussed in Shridhar, would ensure that training does not continue when training has stalled or slowed to such an extent that training does not yield an improvement in a performance metric (e.g., cost of errors) over multiple training iterations. A person of ordinary skill in the art would have been motivated to use a “wait value” and “wait threshold” monitoring the number of epochs for which the validation accuracy remains unchanged as disclosed in Shridhar with the probability of improvement threshold taught by Baker and Lorenz, to optimize the combination of the cost of errors and the cost of performing the computation of training. Specifically, if the probability of improvement remains above the threshold, but is not changing, then training continues. Using this combined early stopping criteria, training would continue while the probability of improvement is “not less than a threshold” and while the wait value (the number of epochs for which the validation accuracy remains unchanged) is greater than a wait threshold. A person of ordinary skill in the art would have combined the teachings of Baker, Lorenz, and Shridhar for at least those reasons discussed in the foregoing *Motivation to Combine Baker, Lorenz, and Shridhar* section.

224. The ’968 patent describes the use of stopping criteria to stop training in the same way, as discussed in FIGS. 4A-4B. In FIG. 4A, Operations 440-480 discuss

using a probability of improvement, and a wait value based on a historic minimum training loss to determine whether to stop or continue training. Ex. 1001, 13:36-14:67. In FIG. 4B, Operations 540-560, discuss using a probability of improvement over a best loss seen for other neural networks (e.g., trained with different hyperparameters) to determine whether a current NN is likely to achieve an improvement in the overall best loss for all NNs. *Id.*, 15:54-16:3. Like the combined teachings of Baker, Lorenz, and Shridhar, the stopping criteria described in the '968 patent determine when a performance metric (e.g., cost of error) is no longer improving relative to a best performance metric.

Claim 3: The method of claim 1, comprising increasing the wait value if the current loss of the NN is not less than the minimum loss in the loss history for the NN.

225. Baker, Lorenz, and Shridar render obvious claim 3.

226. As discussed in reference to claim 2, the '968 patent describes incrementing a wait value if the current loss of the NN is not less than the minimum loss in the loss history for the NN, so that the wait value reflects the number of consecutive training iterations (e.g., epochs) for which no improvement has been shown. Ex. 1001, 14:23-53, 18:63-19:3, 20:15-29.

227. Baker teaches the same concept, to estimate the optimum value of a hyperparameter based on “observations that learning coach 41 has made during the current or previous steps of the training of student learning system 11.” Ex. 1005,

¶[0032]. The learning coach is disclosed as having an objective to “optimize some combination of the cost of errors and the cost of performing the computation.” *Id.*, ¶[0007]. In order for the learning coach to consider the “cost of errors” when optimizing for the mentioned objective, a person of ordinary skill in the art would have computed the “cost of errors” (also known as “loss”) as a part of the “observations” made by the learning coach during the current and previous steps of training the student learning system. By providing the loss to the learning coach, the system can optimize whether the current or previous loss is optimal and can use this information to determine optimal hyperparameters for the student machine learning system. A person of ordinary skill in the art would have understood that “optimizing” a combination of the cost of errors and the cost of performing the computation, the learning coach would identify a minimum loss value during the history of the current and/or previous steps of training the student learning system, because the minimum loss value represents the best performing set of hyperparameters for the student learning system.

228. A person of ordinary skill in the art would have recognized that optimizing the combination of the cost of errors and the cost of performing the computation would include observing the cost of performing further training and determining whether the cost of performing further training is warranted, for example, based on a comparison between the current cost of errors and the cost of

errors (loss) determined in previous training steps. Ex. 1005, ¶[0032]-[0033]. Baker recognizes that identifying “optimum” hyperparameter values requires a comparison between current training step results and prior training step results, stating that “the values of these [hyperparameters] are set by trial-and-error, a very tedious process since each trial requires a test run of the full training procedure on a large amount of training data and then testing on a separate validation set. The process can be automated in the sense that an entire n -dimensional grid can be explored automatically, but that is even more computationally expensive.” *Id.* The “tedious” aspect of this process is running multiple training steps and determining which of those training steps yielded the optimal (minimum) cost of errors (loss), even if a very early training step proved to generate the optimal result.

229. Baker explains that each of these training steps can proceed “until a stopping criterion is reached.” Ex. 1005, ¶[0031]. Moreover, as discussed in reference to claim 2, Shridhar teaches using a “wait value” and a “wait threshold” as a part of an early stopping criteria, stating “when validation accuracy remains unchanged for 5 epochs.” Ex. 1006, 22. By extension, Shridhar’s “wait value” (i.e., the number of epochs when the validation accuracy remains unchanged) increases for each consecutive epoch for which the validation accuracy remains unchanged and would reset to zero once the validation accuracy changes. *Id.*

230. In light of Baker’s disclosure that the learning coach uses observations

“made during the current or previous steps of the training of student learning system 11,” a person of ordinary skill in the art would have recognized that Shridar’s disclosure to stop training “when validation accuracy remains unchanged for 5 epochs” would be applicable to provide an early stopping criteria to stop training when the cost of errors (loss) remains greater than a minimum cost of errors (loss) determined in a previous step of training of student learning system, for at least a threshold number of epochs. To implement this early stopping criteria, Baker’s learning coach would have tracked the number of epochs (“wait value”) for which the cost of errors is not less than a minimum, previously determined cost of errors, and compare that number against a wait threshold. For each epoch that the cost of errors in the current training step remains not less than a minimum, previously determined cost of errors, the “wait value” would have incrementally increased by 1, to denote the current number of epochs that the current cost of errors is not less than the previously established minimum. This tracked “wait value” would stop incrementally increasing once the cost of errors in the current training step falls below the previously established minimum.

Claim 4: The method of claim 1, comprising setting the wait value to zero if the current loss of the NN is less than or equal to the minimum loss in the loss history for the NN.

231. As discussed in reference to claim 3, Baker, Lorenz, and Shridhar render obvious increasing a wait value if the current loss of the NN is not less than

the minimum loss in the loss history for the NN. *Supra*. As mentioned above, Shridhar teaches using a “wait value” and a “wait threshold” as a part of an early stopping criteria, stating that training stops “when validation accuracy remains unchanged for 5 epochs.” Ex. 1006, 22. By extension, Shridhar’s “wait value” (i.e., the number of epochs when the validation accuracy remains unchanged) increases for each consecutive epoch for which the validation accuracy remains unchanged and would reset to zero once the validation accuracy changes. *Id*. This ensures that Shridhar’s “wait value” only increments while the validation accuracy remains unchanged, in the same way that the ’968 patent describes “resetting” a wait value if a performance criteria is reached: “if the current or actual training loss is less than a historic minimum training loss the wait value is not increased, and may be reset or set to zero, or to a value indicating waiting should begin at the beginning or for the maximum of the wait period.” Ex. 1001, 14:25-29.

232. A person of ordinary skill in the art would have been motivated to use an analogous extension to Shridhar’s explicit disclosure when counting the number of epochs that the cost of error (loss) is greater than a previously determined minimum, so that the wait value resets to zero once the current cost of error (loss) is less than or equal to the previously determined minimum loss in the loss history for the NN.

Claim 9: The system of claim 8, wherein the processor is configured to, if the probability is not less than a threshold and a wait value is greater than a wait threshold, determine to continue training.

233. See 2 *supra*.

Claim 10: The system of claim 8, wherein the processor is configured to increase the wait value if the current loss of the NN is not less than the minimum loss in the loss history for the NN.

234. See 3 *supra*.

Claim 11: The system of claim 8, wherein the processor is configured to set the wait value to zero if the current loss of the NN is less than or equal to the minimum loss in the loss history for the NN.

235. See 4 *supra*.

C. Ground 3: Baker, Lorenz, and Jenatton Render Obvious Claims 5 and 12 of the '968 patent.

236. Baker, Lorenz, and Jenatton render obvious claims 5 and 12 of the '968 patent.

Motivation to Combine Baker, Lorenz, and Jenatton

237. There are numerous reasons a person of ordinary skill in the art would have combined Baker, Lorenz, and Jenatton.

238. A person of ordinary skill in the art would have had a reasonable expectation of success in the combination of the references based on the explicit teachings of the references and the knowledge of the person of ordinary skill in the art, including those concepts discussed in Section VI herein.

239. Baker describes a learning coach machine learning model that assists in training a student machine learning model, such as a NN. *See e.g.*, Ex. 1005, *abstract*, ¶¶[0004], [0007], [0025]-[0029]. Baker’s learning coach controls hyperparameters and “makes structural modifications to the student learning system to optimize some combination of the cost of errors and the cost of performing the computation.” Ex. 1005, ¶[0025]. Baker’s learning coach uses the error cost function, specifically a log-likelihood error cost function calculated during a training process to determine the performance of the student machine learning model. Ex. 1005, ¶[0030]. The training process uses the error cost function to proceed iteratively through “minibatches” of training data, and updates trainable parameters for each minibatch. *Id.* The training process proceeds “until a stopping criteria is reached.” Ex. 1005, ¶[0031].

240. A person of ordinary skill in the art would have used a known structure for the learning coach machine learning model, such as the tree structure for arranging hyperparameter combinations as disclosed in Jenatton, to render Baker’s learning coach machine learning model more efficient for identifying potential hyperparameter values for the student machine learning system. Moreover, a person of ordinary skill in the art would have used a known stopping criteria for training student machine learning system with potential hyperparameter values, such as the stopping criteria disclosed in Lorenz, to stop the training process disclosed in Baker.

Using the probability of improvement acquisition function discussed in Lorenz as a part of the training process described in Baker would enable the learning coach to stop a learning process once it is determined to be unlikely that the training process will yield optimal student machine learning model performance. *Id.* Baker explains that a “stopping criteria” is used to stop a training process, and Lorenz provides a detailed explanation of known stopping criteria, including probability of improvement.

241. The combination of Baker, Lorenz, and Jenatton is obvious because the combination relies on known solutions. Jenatton discloses using a tree structure for organizing hyperparameter values into leaves to assist in selection of optimal hyperparameters for training a particular NN, and Baker discloses a learning coach machine learning model for identifying optimal hyperparameter values. Using Jenatton’s data structure with Baker’s learning coach (and a probability of improvement early stopping criterion as disclosed in Lorenz) would improve other known systems to save on resources by streamlining the acquisition function search of potential hyperparameter values. Baker describes the learning coach machine learning system can be, as examples a regression or classification-based model (*see e.g.*, Ex. 1005, ¶[0032]), or can be any other appropriate machine learning structure. A person of ordinary skill in the art would have known that a tree-based machine learning model may be most-appropriate for providing certain outputs. Therefore, a

person of ordinary skill in the art would have used a tree-based machine learning structure for the learning coach, as disclosed in Jenatton, to optimize the operation of the learning coach machine learning model when determining prospective optimal hyperparameter values, using the tree-structure having data relevant to one or more NNs in a leaf to streamline the process of canvassing the search space for potentially optimal hyperparameter values. Ex. 1007, 2, 4

242. A person of ordinary skill in the art would have had a reasonable expectation of success in combining Baker, Lorenz, and Jenatton based on the explicit teachings of the references and the knowledge of the person of ordinary skill in the art. Baker discloses that the learning coach is a machine learning model (Ex, 1005, ¶[0026]) that can estimate the optimum value of a hyperparameter using observations that learning coach has made during the current or previous steps of the training of student learning system, or using training data generated from training other student machine learning models. Ex. 1005, ¶[0041]. A person of ordinary skill in the art would have recognized that using a tree data structure as disclosed in Jenatton to identify hyperparameters relevant to the student learning system within leaves of the tree data structure, would streamline the search for optimal hyperparameters, as compared with an iterative canvassing of an entire search space. Jenatton states that such data (e.g., hyperparameter values) can be populated within leaf nodes of a tree data structure. Ex. 1007, 2.

243. Once hyperparameter values are selected, Baker explains that the learning process continues “until a stopping criterion is reached.” Ex. 1005, ¶[0031]. Lorenz explains that a probability of improvement calculation is a known stopping criteria, demonstrates that the use of probability of improvement criteria are usable in machine learning training methods, and provides details about how to use the probability of improvement calculation as a stopping criteria, when the calculated probability of improvement is less than a threshold significance level. *See e.g.*, Ex. 1008, 4-5. Accordingly, a person of ordinary skill in the art had a reasonable expectation of success in combining Baker, Lorenz, and Jenatton.

Claim 5: The method of claim 1, wherein determining using a model a probability of improvement in the loss of the NN comprises obtaining from a leaf of at least one tree data structure data relevant to the NN.

244. Baker, Lorenz, and Jenatton render obvious claim 5.

245. As discussed in reference to element 1[b.3]-1[c], Baker and Lorenz render obvious using a model (e.g., the learning coach of Baker) as a part of a determination of a probability of improvement in the loss of a NN (e.g., the student machine learning model of Baker). Baker explains that the learning coach seeks to optimize “some combination of the cost of errors and the cost of performing the computation.” Ex. 1005, ¶[0007]. Baker explains that the learning coach may model the learning performance of student learning system 11, for example, as a regression, or the learning coach may directly estimate the optimum value of a hyperparameter

using observations the learning coach 41 has made during the current or previous steps of the training of student learning system 11. *Id.*, ¶[0032]. Baker also explains that the learning coach “may learn categorical hyperparameters as a classification task.” *Id.* Therefore, Baker teaches that the learning coach model can generate an output, including hyperparameter values, as a result of training the learning coach with data. A person of ordinary skill in the art would recognize that the structure of the learning coach (e.g., the type of machine learning model used) is provided such that it generates relevant output for a student machine learning model being trained.

246. Jenatton teaches using a tree-structure to identify hyperparameters that are relevant to a NN to be trained. Ex. 1007, 2. Jenatton provides an example in which hyperparameter selections are encoded as data in a decision tree, “where inner nodes select between different models and hyperparameters populate leaf nodes.” *Id.* Jenatton proposes to leverage this tree structure during an acquisition function, such as to select data in the form of optimal hyperparameters for a model by “select[ing] the most promising leaf node to score, effectively restricting our attention to a portion of X . We then optimize over all possible anchor points in this portion of space. This can result in a drastic reduction in the number of surrogate functions to optimize over.” *Id.* Selecting “the most promising leaf node” as discussed in Jenatton is the claimed “obtaining from a leaf of at least one tree data structure data relevant to the NN,” because a “most promising leaf node” in a tree

data structure comprising hyperparameters for a neural network would be “promising” based on the leaf node’s relevance to the NN being trained. A person of ordinary skill in the art would have recognized that selecting a “most promising” leaf node for training a model would include determining that a particular leaf node is relevant to the NN being trained, such as by determining that the hyperparameters could be used during training of the NN.

247. Moreover, Jenatton describes that acquisition functions include, among other examples “probability of improvement (PI)” functions. Ex. 1007, 4. Although Jenatton focuses on a different acquisition function, expected improvement (EI), Jenatton states that acquisition function “plays a critical role in Bayesian Optimization as it selects anchor points by performing an exploration-exploitation trade-off.” *Id.* A person of ordinary skill in the art reviewing Jenatton would have understood that, if PI were used as the acquisition function, as discussed in Lorenz and Jenatton, leaf nodes and anchor points of a tree-structure would have been selected as being “most promising” based on a determination that the hyperparameters within the selected leaf node has a high probability of improving the performance of the NN, as compared with hyperparameters in a current training step. *See* Ex. 1006, 22; Ex.1007, 4. Although Jenatton indicates that a PI acquisition function is not as optimal as an EI acquisition function, the PI function would still be usable for selection of data relevant to a current NN being trained. Moreover,

Lorenz suggests combining PI and EI acquisition functions to further improve performance, and a person of ordinary skill in the art would recognize that Lorenz's teachings to combine these acquisition functions would have been usable with Jenatton's tree data structure. Using a PI acquisition function (alone or in combination with an EI acquisition function) to select leaf nodes having data relevant to a NN from Jenatton's tree structure would have included calculating the probability of improvement in performance (e.g., cost of error, also known as training loss) of the NN using the data in the leaf node, and determining that the calculated probability of improvement indicates that the leaf node is "most promising" for improving the performance of the NN. Therefore, a POSTIA would have recognized that Jenatton's tree structure and acquisition functions would have been an efficient structure for Baker's learning coach to identify hyperparameters relevant for training a student machine learning model.

248. Jenatton's tree structure, including leaf nodes having data relevant to a NN is the same as the '968 patent's discussion of using a tree data structure that organizes NN into leaf nodes of a tree data structure, for example, based on "hyperparameters and other NN characteristics." Ex. 1001, 6:36-61, 7:30-34, 9:12-46.

249. A person of ordinary skill in the art would have recognized that Jenatton's tree-structure would have been usable with Baker's learning coach, so

that the learning coach could efficiently identify candidate hyperparameter values from an organization tree-structure in which candidate hyperparameter values are organized in leaf nodes of the tree-structure. *See* Ex. 1005, ¶[0032], Ex. 1007, 2. By looking at “most promising” leaf nodes in a tree-structure as disclosed in Jenatton, instead of exploring “an entire n -dimensional grid” as discussed in Baker, Baker’s learning coach can efficiently identify candidate hyperparameter values for testing, and can use Lorenz’s PI acquisition function (as also discussed in Jenatton) to determine the “most promising” leaf node in the tree structure.

Claim 12: The system of claim 8, wherein determining using a model a probability of improvement in the loss of the NN comprises obtaining from a leaf of at least one tree data structure data relevant to the NN.

250. *See* 5 *supra*.

X. Conclusion

I declare that all statements made herein of my own knowledge are true, and that all statements made on information and belief are believed to be true, and that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Dated: June 5, 2025

Scott Denning
Scott Denning