UNITED STATES PATENT AND TRADEMARK OFFICE

_____

BEFORE THE PATENT TRIAL AND APPEAL BOARD

_____

Perplexity AI, Inc.

Petitioner

v.

Comet ML, Inc.

Patent Owner

_____

Case: IPR2025-01063

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

_____

PETITION FOR *INTER PARTES* REVIEW
UNDER 35 U.S.C. §312 AND 37 C.F.R. §42.104

Mail Stop PATENT BOARD
Patent Trial and Appeal Board
US Patent and Trademark Office
PO Box 1450
Alexandria, Virginia 22313-1450

TABLE OF CONTENTS

TABLE OF AUTHORITIES

Page(s)

Cases

Statutes

Regulations

LIST OF EXHIBITS

Exhibit	Short Name	Description
Ex. 1001	'968 Patent	U.S. Patent No. 11,650,968 (the "'968 Patent")
Ex. 1002	Prosecution History	Prosecution History for U.S. Patent No. 11,650,968
Ex. 1003	Denning Declaration	Declaration of Scott Denning
Ex. 1004	CV	*Curriculum Vitae* of Scott Denning
Ex. 1005	Baker	International Publication No. WO 2018/175098 ("Baker")
Ex. 1006	Shridhar	*A Comprehensive guide to Bayesian Convolutional Neural Network with Variational Inference* ("Shridhar")
Ex. 1007	Jenatton	*Bayesian Optimization with Tree-structured Dependencies* ("Jenatton")
Ex. 1008	Lorenz	*Stopping criteria for boosting automatic experimental design using real-time fMRI with Bayesian optimization* ("Lorenz")
Ex. 1009	Domhan	*Speeding up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves* ("Domhan")
Ex. 1010	Baker '798 Pub.	U.S. Pub. No. 2019/0095798 ("Baker '798 Pub.")
Ex. 1011	Hall-Ellis Declaration	Declaration of Dr. Sylvia Hall-Ellis

Perplexity AI, Inc. ("Petitioner") petitions for IPR under 35 U.S.C. §§311–319 and 37 C.F.R. §42 of Claims 1-12 ("the Challenged Claims") of U.S. Patent No. 11,650,968 ("the '968 Patent"). Petitioner requests cancellation of the Challenged Claims.

I. STANDING UNDER 37 C.F.R. §42.104(a)

Petitioner certifies the '968 Patent is available for IPR, and Petitioner is not barred or estopped from requesting IPR.

II. PAYMENT OF FEES

Petitioner authorizes Account No. 16-0605 to be charged.

III. OVERVIEW OF CHALLENGES AND RELIEF REQUESTED

Pursuant to 37 C.F.R. §42.22(a)(1) and 42.104(b)(1)–(2), Petitioner requests cancellation of the Challenged Claims pursuant to the grounds below. Additional support is provided in the Declaration of Mr. Scott Denning (Ex. 1003). Ex. 1003, ¶¶1-250.

- **Ground 1**: Baker and Lorenz render Claims 1 and 6-8 obvious.
- **Ground 2**: Baker, Lorenz, and Shridhar render Claims 2-4 and 9-11 obvious.
- **Ground 3**: Baker, Lorenz, and Jenatton render Claims 5 and 12 obvious.

IV. BACKGROUND

A. Background

Neural Networks (NNs) are examples of machine learning (ML) systems that assign "weights" (also known as "trainable parameters") to neurons arranged in connected layers of the NN. Ex. 1001, 1:21-47. Once trained, the NN can predict (and generate) output data based on a received input query, so long as the NN was trained with a relevant corpus of training data that allows a NN to self-teach an algorithm describing the relationship between an input request and output data. Ex. 1001, 2:10-62. For example, a NN can be fed training data including labeled examples of images of a cat so the NN can identify cats in other images. *Id*.

Training a NN involves iteratively feeding training data to the NN in a process involving "backpropagation." *Id.* First, input data is presented to the NN, which produces an output (during a forward pass), 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. *Id.* The goal of the training process is to minimize the loss during backpropagation. Ex. 1001, 1:39-47. Generally, the loss decreases as more training data is presented to the NN until a point at which the NN begins to be "overfit" to the training data, when the self-learned algorithm becomes less capable of recognizing patterns outside of the training data. Ex. 1003, ¶51.

There are several known ways of calculating a loss, aka a cost of errors, which defines "a deviation from an expected or correct output value for a given input, during learning or training." Ex. 1001, 2:46-48. Example methods for calculating a loss function include "a negative log-likelihood or residual sum of squares." *Id*., 2:46-57.

Rather than calculating a loss function after iterating through every piece of training data, NN learning processes are typically organized into iterative phases, known as batches or "epochs." Ex. 1001, 2:58-3:3; *see also* Ex. 1006, 17, 27, 34-36; Ex. 1010, ¶[0014]. During an epoch, a subset of the corpus of training data (the subset of training data also known as a batch of forward/backward passes called "minibatches") is presented to the NN, during which multiple outputs are generated and backpropagated through the NN. *See* Ex. 1010, ¶[0013]; Ex. 1003, ¶¶48-54. The weights can be modified after each epoch (i.e., after each batch of training data is run through the NN), and training may include multiple phases, i.e., multiple epochs, that continue until some stopping criteria is reached. *Id.* That stopping criteria can be determined "by periodically testing a NN using a holdout data set not included in training data," so training can stop "when improvement stagnates." Ex. 1001, 3:4-18.

The learning process itself can be manipulated, for example, by adjusting the batch size, the number of trainable parameters (weights assigned to neurons in the

neural network), and the number of training samples. Ex. 1001, 1:39-67. These user-set parameters, known as "hyperparameters," can also describe attributes of the architecture of the NN, including 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. *Id.* The selection of a set of hyperparameters for training a NN influences the minimum loss achievable by the NN. Ex. 1001, 2:5-21; Ex. 1003, ¶¶55-58. Training a NN is an iterative process, in which different hyperparameter combinations are tested to attempt to find the lowest-possible loss achievable by the NN, because there is no easy way to predict which hyperparameter combination will yield a trained NN having the "best" or lowest loss function. *Id*.

B. The '968 Patent

The '968 Patent uses a second ML model to 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." Ex. 1001, 26:23-28. The'968 Patent thus attempts 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; Ex. 1003, ¶¶59-61.

The '968 Patent uses a known stopping criteria—comparing a calculated "probability of improvement" value against a threshold to determine when to stop

training. "After 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. The '968 Patent explains that multiple models are used for calculating the probability of improvement 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.* With careful planning by the user, the threshold can be set so that the training process can be stopped "at the point at or as soon as the minimum predicted validation loss is achieved." Ex. 1001, 5:62-67. Ex. 1003, ¶¶62-64.

The secondary model that assists in training a NN is a "model trained using training data from other NNs." Ex. 1001, 5:39-51. The training data reflects how certain hyperparameters impacted the training loss of other models. *Id*. As the NN is

trained, the secondary model may be used to periodically calculate a "probability of improvement over a best loss for this NN." Ex. 1001, 13:36-14:22. 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, training is stopped. Ex. 1001, 14:63-67. Ex. 1003, ¶¶65-76.

Certain embodiments use a secondary early stopping criterion, for example, incrementing a wait value if the actual training loss (as calculated) is not less than a historic minimum training loss. Ex. 1001, 14:23-61. As this wait value increases, the described method includes a comparison between the wait value and a wait threshold (referred to as a "patience" value). If the wait value is greater than or equal to the wait threshold, training is stopped. *Id*., 14:63-15:21. Ex. 1003, ¶¶77-89.

C. Prosecution History

U.S. Patent Application No. 16/707,265 was filed December 9, 2019. Ex. 1001, Cover. Ex. 1002, 105-168. Patent Owner responded to the sole Office Action by pointing out that the cited primary reference, Prechelt, discussed a "generalization loss, … which measures increase in error" and that this generalization loss "does not teach or suggest the claimed probability of improvement." Ex. 1002, 47-49. The Patent Owner further argued that a secondary cited reference, 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.* Ex. 1003, ¶¶90-95.

D. Priority Date

Petitioner assumes, without determining, that the earliest priority date for the '968 Patent is May 24, 2019. Ex. 1001; Ex. 1002, 89. Petitioner reserves the right to challenge the priority date.

E. Claim Construction

The terms should receive their respective plain and ordinary meaning as understood by a person of ordinary skill in the art under *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005). Ex. 1003, ¶98.

V. PRIOR ART

A. Baker (Ex. 1005)

PCT Publication No. WO/20188/175098 was filed March 5, 2018 and published on September 27, 2018. Ex. 1005, *Cover*; Ex. 1003, ¶¶99-122. Baker is prior art under at least AIA 35 U.S.C. § 102(a)(1). *See* 35 U.S.C. §102(a)(1).

B. Shridhar (Ex. 1006)

"A Comprehensive Guide to Bayesian Convolutional Neural Network with Variational Inference" was published January 8, 2019. Ex. 1011, ¶¶35-37; Ex. 1003, ¶¶123-130. As explained in detail by Dr. Hall-Ellis (Ex. 1011), papers such as Shridhar, that were published at Cornell University (*arXiv*, a well-known open access research-sharing platform) are given an identifier when the submission is publicly announced. Ex. 1011, ¶¶20-34. The published version of Shridhar available from *arXiv* indicates a January 8, 2019 publication date. Ex. 1011, ¶36. Shridhar qualifies as prior art under at least 35 U.S.C. § 102(a)(1) based on its publication date. *See* 35 U.S.C. §102(a)(1).

C. Jenatton (Ex. 1007)

"Bayesian Optimization with Tree-structured Dependencies" was published August 6, 2017. Ex. 1011, ¶39; Ex. 1003, ¶¶131-135. As explained in detail by Dr. Hall-Ellis (Ex. 1011), conference papers like Jenatton are made publicly available via the *ACM Digital Library*. Ex. 1011, ¶¶20-34. Jenatton is available in the *ACM Digital Library* with a publication date of August 6, 2017. Ex. 1011, ¶¶38-40. Jenatton qualifies as prior art under 35 U.S.C. § 102(a)(1) and (a)(2) based on its publication date.

D. Lorenz (Ex. 1008)

"Stopping Criteria for Boosting Automatic Experimental Design Using Real-Time fMRI with Bayesian Optimization" was published November 24, 2015. Ex. 1011, ¶42; Ex. 1003, ¶¶136-143. As explained in detail by Dr. Hall-Ellis (Ex. 1011), conference papers like Lorenz that were published via *arXiv*, a well-known open access research-sharing platform, are given an identifier when the submission is publicly announced. Ex. 1011, ¶¶20-34. The published version of Lorenz from *arXiv* indicates a publication date of November 24, 2015. Ex. 1011, ¶¶41-44. Lorenz qualifies as prior art under 35 U.S.C. § 102(a)(1) and (a)(2) based on its publication date.

VI. LEVEL OF ORDINARY SKILL IN THE ART

A person of ordinary skill in the art ("POSITA") pertinent to the '968 Patent would have had a Bachelor of Science degree in Computer Science with two years of related work experience building and training artificial neural networks (ANNs), including selecting and testing the accuracy ANNs of neural networks trained with various hyperparameter combinations. Ex. 1003, ¶46. 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. *Id.* More education can supplement practical experience and *vice versa*. *Id*.

VII. THE CHALLENGED CLAIMS ARE UNPATENTABLE

A. Ground 1: Claims 1 and 6-8 are rendered obvious over Baker and Lorenz

Baker and Lorenz render obvious Claims 1 and 6-8. Ex. 1003, ¶¶144-204.

1. Overview of Ground 1

a. Baker

Baker describes a "coach" machine learning system to help build and train a "student" machine learning system. Ex. 1005, ¶[0004]; Ex. 1003, ¶100. 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*. Other than using the learning coach that is trained on data from the current student machine learning model and other student machine learning models, Baker's training process includes typical backpropagation techniques and continues until a stopping criterion is reached. *Id.*

The coach machine learning system modifies the student machine learning system, such as modifying the hyperparameters and/or architecture of the student machine learning system (e.g., making "additions" to the number of layers in a DNN embodying the student machine learning system) to improve the capability of the student machine learning system while guaranteeing that there will be no degradation in performance. Ex. 1005, ¶¶[0007], [0044]; Ex. 1003, ¶101. 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*.

The learning coach's objectives may differ from the student machine learning model – the learning coach's objectives may be twofold, to "optimize some combination of the cost of errors and the cost of performing the computation." Ex. 1005, ¶¶[0007], [0044]; Ex. 1003, ¶102.

Baker explains that the student and learning coach are models trained for different purposes. Ex. 1003, ¶¶103-105. 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, ¶¶[0007], [0044]; Ex. 1003, ¶¶103-105. 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]; Ex. 1003, ¶105. The learning coach, on the other hand, learns from and processes a different set of data from the student machine learning system. Ex. 1005, ¶[0026]; Ex. 1003, ¶106. The learning coach's training data includes "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]), and can include observations recorded during training of multiple student machine learning systems (including "other" student learning systems) (Ex. 1005, ¶[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." Ex. 1005, ¶¶[0007], [0040]-[0041], [0044]; *see also* Ex. 1003, ¶¶109-122.

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." Ex. 1005, ¶¶[0007], [0030], [0044]; Ex. 1003, ¶106-108.

Baker explains that the learning coach machine learning model learns to optimize hyperparameters. Ex. 1003, ¶110. When Baker's learning coach optimizes hyperparameters as continuous values, the learning coach can be a neural network, 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*. Ex. 1003, ¶109.

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], [0039]; Ex. 1003, ¶110. The learning coach minimizes this iterative and resource-intensive learning process by selecting hyperparameters for the student machine learning system based on observations in the training data that the learning coach can use to suggest optimal hyperparameter values. *Id*.

The process for training the student machine learning model, after selection of the hyperparameters, is "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." 1005, ¶[0030]; Ex. 1003, ¶¶110-111. This process continues until a stopping criterion is reached. 1005, ¶[0031].

As training of a student machine learning system proceeds, connection weight parameters for the student machine learning system are updated until a stopping criterion is reached, and 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]; Ex. 1003; ¶¶111-121. 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*.

Baker states that the "student learning system 11 and learning coach 41 can be implemented with computer hardware or software." Ex. 1005, ¶[0046]; Ex. 1003, ¶122. 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. Lorenz

Lorenz describes well-known stopping criteria, specifically as an example of Bayesian optimization used for experiment design. Specifically, Lorenz focuses on stopping criteria that were most often used during machine learning training processes, and uses these Bayesian optimization stopping criteria to design experiments to evoke a desired target brain pattern. Ex. 1008, *Abstract*, 4; Ex. 1003,

¶¶136-143. 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)." Ex. 1008, 4-5; Ex. 1003, ¶¶137-141. 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*.

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

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

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. *Id.*

Lorenz defines the PI function as:

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

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

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. Ex. 1008, 4-5; Ex. 1003, ¶¶137-141. 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*.

c. Motivation to Combine

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]. Ex. 1003, ¶¶146-147.

A POSITA would have used a known stopping criteria, as disclosed in Lorenz, to stop the training process disclosed in Baker. Ex. 1003, ¶148. 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. *Id.*

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.,* Ex.1008, 7-8 (see titles of references 6, 8, 11, as examples)). A POSITA looking for literature relating to experimental stopping criteria that are usable for training machine learning models would look at publications discussing the general principles of experimental stopping criteria, including stopping criteria known to be used in Bayesian Optimization methodologies as discussed in Lorenz. Ex. 1003, ¶148. For example, a POSITA looking for machine-learning training optimization approaches would look to Lorenz at least because Lorenz extensively discusses that the same processes have been used extensively in ML training algorithms. *Id.*

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. Ex. 1003, ¶¶149-152.

A POSITA 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 POSITA. Ex. 1003, ¶153. 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; Ex. 1003, ¶153.

A POSITA 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]; Ex. 1003, ¶¶154-155. 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 hyperparameter values proves that those values are optimal by iteratively testing other hyperparameter values and calculating a training loss for the student machine learning model. Ex. 1003, ¶154.

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 used a known early stopping criteria, as disclosed in Lorenz, to stop non-optimal training runs. Ex. 1008, 4-5; Ex. 1003, ¶154. 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, as discussed in Lorenz, 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. Ex. 1003, ¶154. This 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 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. Ex. 1003, ¶154.

2. *Claim 1*

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

Baker renders obvious element 1[pre], to the extent it is limiting. Ex. 1003, ¶¶155-158.

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*; Ex. 1003, ¶155. 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].

The learning process of the student machine learning system is controlled by hyperparameters. Ex. 1005, ¶[0030]; Ex. 1003, ¶156. 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].

Baker explains that hyperparameters control this learning process. Ex. 1005, ¶¶[0032]-[0033]; Ex. 1003, ¶157. The "learning coach" machine learning model

described in Baker is trained to optimize selection of 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].

The '968 Patent's discussion of "training a neural network" matches Baker's discussion of the training process for the student machine learning model. Ex. 1003, ¶158. 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 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." Ex. 1001, 1:39-67, 5:19-51, 13:10-35, 15:41-53.

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

Baker renders obvious element 1[a]. Ex. 1003, ¶¶159-166.

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, where each "batch may be a set of multiple training data presented to a NN which results in a number of outputs." Ex. 1001, 2:59-65. 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. This training process recited in 1[a] is typical for a NN. Ex. 1003, ¶160.

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," where examples of loss functions include "a negative log-likelihood or residual sum of squares." Ex. 1001, 2:46-57; Ex. 1003, ¶161.

Like the '968 Patent, Baker describes the typical training process for a NN (i.e., Baker's student machine learning model). Ex. 1003, ¶162. Baker explains that the student machine learning model undergoes "training" over a series of minibatches; each of which include training examples. Ex. 1005, ¶¶[0008], [0039]; ex. 1003, ¶162.[1] Completion of training through a series of Baker's minibatches

[1] The '968 Patent is ambiguous as to how many "losses" are calculated during an epoch. 1[a] states that "a loss is computed" during an epoch, yet 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

(which are analogous to batches of training data) is the same as completing a "training epoch" as recited in 1[a]; Ex. 1003, ¶163. 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 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].

Baker also describes computing a "log-likelihood error cost function" during training on each minibatch. Ex. 1005, ¶¶[0030]-[0031]; Ex. 1003, ¶164. This is the same as the example "negative log-likelihood" loss function utilized by the '968

---

set" so that multiple losses are computed per epoch. Ex. 1001, 2:58-66. A POSITA 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. Ex. 1003, n1. Thus, Baker discloses this training process. *Id.*

Patent to calculate a loss. Ex. 1001, 2:46-57. A POSITA would recognize that the terms "loss" and a "cost of errors" are synonymous when discussing a NN training process. *See e.g.*, Ex. 1006, 14 (discussing logarithmic formulas for likelihood functions); Ex. 1007, 3; Ex. 1003, ¶164.

A POSITA 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. Ex. 1003, ¶165.

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

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]. Ex. 1003, ¶¶167-169.

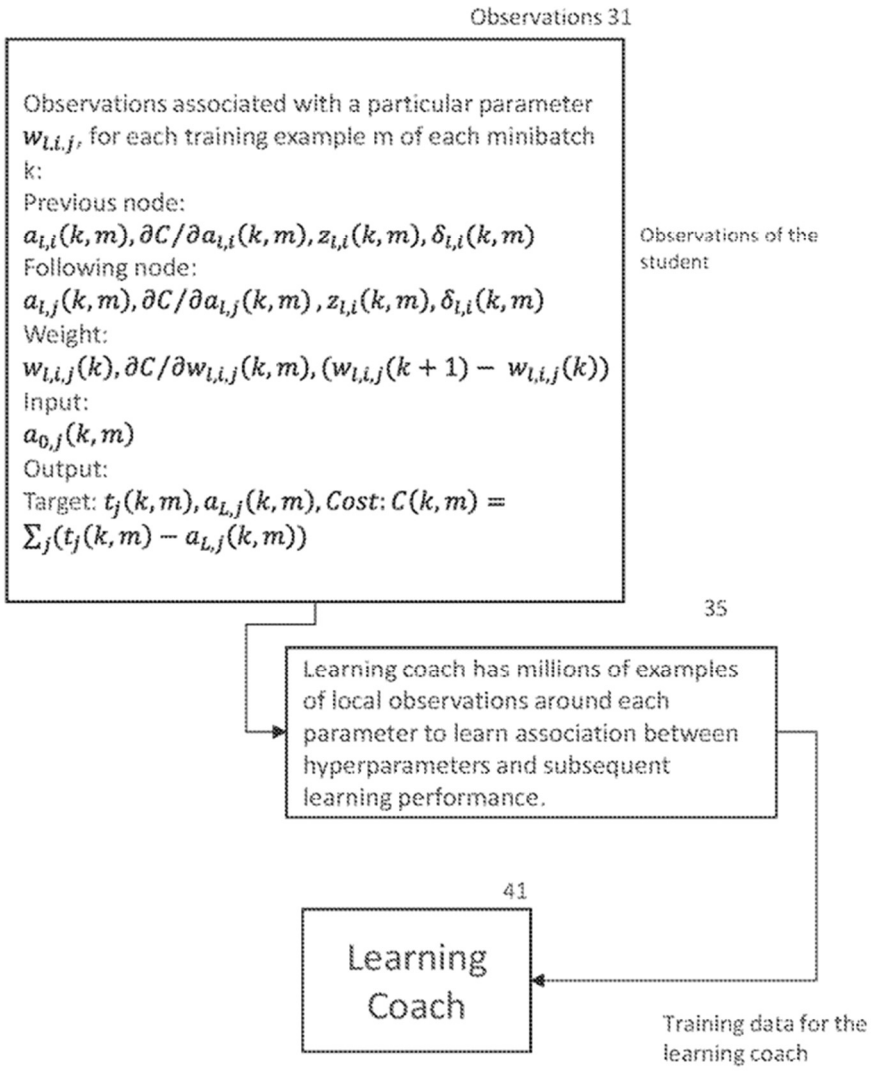
The '968 Patent describes "parameters" as including "weights of links or functions that link layers of a NN, which may be adjusted during learning or training" and "[h]yperparameters may be parameters that cannot be directly

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 include "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." *Id*.; Ex. 1003, ¶168.

Baker teaches using a set of model parameters (hyperparameters) 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 machine 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 Figure 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*. Ex. 1003, ¶169.

Ex. 1005, FIG. 3

Fig. 3



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

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]. Ex. 1003, ¶¶170-173.

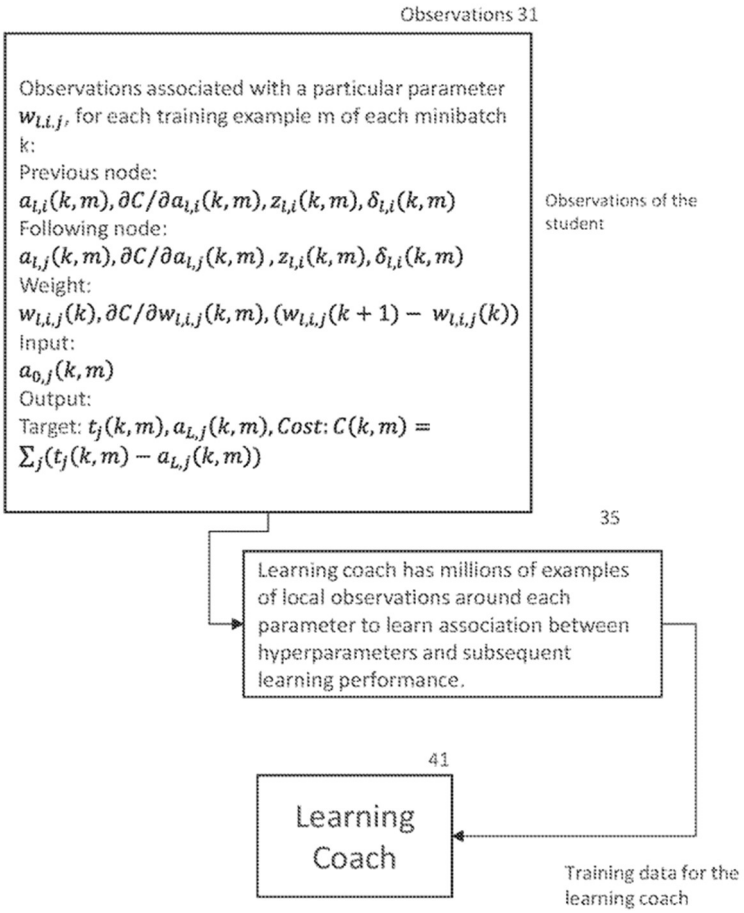
Baker's "cost of errors" is the same as the "loss" discussed in the '968 Patent. Ex. 1003, ¶171.

Baker discloses that its 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]; Ex. 1003, ¶172. 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 POSITA would have recognized that the training data for the learning coach 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. Ex. 1003, ¶172. 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.

This enables the learning coach to attempt to identify hyperparameter values that optimize the cost of errors. Ex. 1003, ¶172. Figure 3 of Baker, reproduced below, demonstrates that a "cost" is calculated for a target output included in the training data. *Id.* 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*.

Ex. 1005, FIG. 3

Fig. 3



The'968 Patent similarly discusses using training losses. Training losses are computed during training of a NN. Ex. 1001, 2:46-66, 13:10-23, 15:41-53; Ex. 1003, ¶173. Data reflecting historical training losses from the current NN and/or other NNs may also be included in training data for a model used to facilitate training of the NN. Ex. 1001, 5:7-18, 5:39-51, 6:7-61, 12:40-13:9, 15:32-40.

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

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]. Ex. 1003, ¶¶174-178.

Like the "model" discussed in the '968 Patent, Baker's learning coach assists in training a NN (e.g., Baker's student machine learning model). Ex. 1001, 5:19-20, 26:23-29; Ex. 1005, ¶[0007], [0025]-[0026]; Ex. 1003, ¶175.

The learning coach is a "model that has been trained using training losses of a plurality of NNs other than the NN" as recited in 1[b.3]. Baker's learning coach is a machine learning system. Ex. 1005, ¶¶[0025]-[0026], 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 identifies relationships between certain hyperparameter values and the resulting cost of errors (considering other contextual data included in the training data). Ex. 1003, ¶176.

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]. 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]; Ex. 1003, ¶177.

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, including data reflecting the performance of student machine learning models (e.g., a measure of the cost of errors) trained with various hyperparameter combinations. *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. Ex. 1003, ¶178.

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

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].

To the extent it is argued that Baker does not explicitly disclose 1[c], Baker and Lorenz render obvious 1[c]. Ex. 1003, ¶¶179-188.

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. Ex. 1003, ¶¶180-182.

Baker explains that training a NN with a selected set of hyperparameters continues "until a stopping criterion is reached." Ex. 1005, ¶[0031]. A POSITA 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]), thus 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. Ex. 1003, ¶183. A POSITA 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]; Ex. 1003, ¶186. Thus, Baker renders obvious 1[c]. Ex. 1003, ¶183.

To the extent it is argued that Baker does not teach or render obvious 1[c], a POSITA would have turned to Lorenz. Ex. 1003, ¶184.

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; Ex. 1003, ¶185. Lorenz describes the two well-known stopping criteria (and a combination of those two well-known stopping criteria) as using "expected improvement (EI)" and "probability of improvement (PI)" acquisition functions

based on 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*. These acquisition functions use a performance metric for comparison, i.e., what constitutes an "expected improvement" or "probability of improvement." Ex. 1003, ¶185.

Specifically, Lorenz describes a method of combining these two well-known 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*. 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. Ex. 1003, ¶185.

Baker discloses 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]; Ex. 1003, ¶186. 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]; Ex. 1003, ¶186. A POSITA would understand 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. *Id*. 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].

A POSITA would have been motivated to combine Baker and Lorenz for at least the reasons set forth in the section herein titled *Motivation to Combine.* Ex.

1003, ¶187. 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 Baker uses parameters, a calculated training loss, and a model (the learning coach) for optimizing some measure of the cost of errors. Ex. 1003, ¶187. Thus, a POSITA would have combined Baker and Lorenz so that Baker's learning coach calculates a probability of improvement (PI) in a cost of errors, to be used as a well-known stopping criterion for training a student machine learning model using predicted "best" hyperparameter values. *Id.* 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. *Id.* Thus, Baker and Lorenz render obvious 1[c]. *Id.*

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

Baker and Lorenz renders obvious 1[d]. Ex. 1003, ¶¶189-192.

Baker describes a "well-known training process" in which a set of operations is performed "until a stopping criterion is reached." Ex. 1005, ¶[0031]; Ex. 1003, ¶190. That is, 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].

A POSITA 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. Ex. 1003, ¶191. 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. *Id.*

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.

A POSITA would have combined Baker and Lorenz references for the reasons discussed in the above motivation to combine section. Therefore, Baker and Lorenz renders obvious 1[d]. Ex. 1003, ¶192.

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

Baker and Lorenz render obvious claim 6. Ex. 1003, ¶¶193-195.

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. *See* 1[c] and 1[d]*, 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:

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

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 of the measured performance metric, a POSITA 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. Ex. 1003, ¶¶194-195.

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

Baker renders obvious claim 7. Ex. 1003, ¶¶196-198.

The '968 Patent describes "hyperparameters" as a type of "parameters." Ex. 1001, 1:39-2:21; Ex. 1003, ¶197.

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]; Ex. 1003, ¶198. 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 herein, 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). These

hyperparameters are included in training data for the learning coach, as shown in Figure 3, 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, and consequently impacts the calculation of a probability of improvement, as discussed in reference to element 1[c]. *Id*. Therefore, Baker renders obvious claim 7. Ex. 1003, ¶198.

5. *Claim 8*

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

See 1[pre] *supra*. Ex. 1003, ¶199.

b. 8[a]: a memory; and

Baker teaches element 8[a]. Ex. 1003, ¶¶200-201.

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), thus Baker's system for training the student machine learning model comprises a memory. Ex. 1005, ¶[0046]; Ex. 1003, ¶201.

c. 8[b]: a processor configured to:

Baker renders obvious element 8[b]. Ex. 1003, ¶¶202-203.

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].

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

See 1[a] *supra.*

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

See 1[b.1] *supra.*

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

See 1[b.2] *supra.*

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

See 1[b.3] *supra.*

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

See 1[c] *supra.*

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

See 1[d] *supra.*

B. Ground 2: Claims 2-4 and 9-11 are rendered obvious over Baker, Lorenz, and Shridhar

Baker, Lorenz, and Shridhar render obvious Claims 2-4 and 9-11. Ex. 1003, ¶¶210-235.

1. *Overview of Ground 2*

a. Shridhar

Shridhar discusses other well-known stopping criteria when testing proposed uses of a Bayesian Convolutional Neural Network (BayesCNN) using Variational Interference. Ex. 1006, *Abstract*; Ex. 1003, ¶¶123-130. Shridhar's proposed structure addresses problems of overfitting on small datasets by adding a measure of uncertainty and regularization to predictions generated by BayesCNN during inference. Ex. 1006, 2.

The crux of the Shridhar's contribution includes 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*.

b. Motivation to combine

There are several reasons a POSITA would have combined, and had a reasonable expectation of success in combining, Baker, Lorenz, and Shridhar. Ex. 1003, ¶¶211-219. 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]. Ex. 1003, ¶213.

A POSITA would have used a known stopping criteria, such as those disclosed in Lorenz and Shridhar, to stop the training process disclosed in Baker. Ex. 1003, ¶214.

Because Baker describes the learning coach's objective is to optimize the cost of errors and the cost of performing the computation of the learning process, a POSITA 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]; Ex. 1003, ¶¶214-216. 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. In addition, 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. Ex. 1003, ¶215.

A POSITA would have combined Baker, Lorenz, and Shridhar to improve the cost of computation of Baker's training system, such as by using stopping criteria proposed by Lorenz and Shridhar. Ex. 1003, ¶216.

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. Ex. 1003, ¶217.

A POSITA 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 POSITA. Ex. 1003, ¶¶212, 218. 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]; Ex. 1003, ¶218. 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.

A POSITA 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. Ex. 1003, ¶219. Providing multiple early stopping criteria would minimize the number of suboptimal training iterations that proceed to completion. Accordingly, a POSITA had a reasonable expectation of success in combining Baker, Lorenz, and Shridhar. *Id.*

2. *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.*

Baker, Lorenz, and Shridhar render obvious claim 2.

As discussed with respect to 1[c]-1[d], Baker and Lorenz render obvious using a probability of improvement in a training loss threshold as a stopping criterion to determine whether to continue or terminate a learning process for a set of hyperparameters 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; Ex. 1003, ¶220. This stopping criterion is provided so that unnecessary training using a prospective set of hyperparameters 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; Ex. 1003, ¶220.

Baker also 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]; Ex. 1003, ¶221.

A POSITA would have recognized that using a probability of improvement-based stopping criteria (*see* 1[c]-1[d] *supra*) would be relevant for optimizing the cost of errors. Ex. 1003, ¶222. A POSITA 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. *Id*.

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 POSITA 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. Ex. 1003, ¶¶222-223. 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 POSITA would have combined the teachings of Baker, Lorenz, and Shridhar for at least those reasons discussed in the foregoing *Motivation to Combine* section. Ex. 1003, ¶223.

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. Ex. 1003, ¶224; *see also* Ex. 1001, 13:36-14:67, FIGs. 4A-4B, 15:54-16:3.

3. *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.*

Baker, Lorenz, and Shridar render obvious claim 3. Ex. 1003, ¶¶225-230.

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; thus 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; Ex. 1003, ¶226; [2] *supra*.

Baker teaches estimating 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]; Ex. 1003, ¶227. Baker disclose that the learning coach has an objective to "optimize some combination of the cost of errors and the cost of performing the computation." Ex. 1005,, ¶[0007]; Ex. 1003, ¶227. For the learning coach to consider the "cost of errors" when optimizing for the mentioned objective, a POSITA 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. Ex. 1003, ¶227. By providing this 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 POSITA 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. Ex. 1003, ¶227.

A POSITA 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]; Ex. 1003, ¶228. 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. *Id.*

Baker explains that each of these training steps can proceed "until a stopping criterion is reached." Ex. 1005, ¶[0031]. 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. Shridhar's "wait value" (i.e., the number of epochs when the validation accuracy remains unchanged) thus increases for each consecutive epoch for which the validation accuracy remains unchanged and would reset to zero once the validation accuracy changes. *Id*.; Ex. 1003, ¶229.

In view of Baker's explicit disclosure that the learning coach uses observations "made during the current or previous steps of the training of student learning system 11," a POSITA would have recognized that Shridar's disclosure to stop training "when validation accuracy remains unchanged for 5 epochs" would be applicable to provide 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. Ex. 1003, ¶230.

4. *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.*

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. *See* claim 3, *supra*. 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; Ex. 1003, ¶231. Shridhar's "wait value" (i.e., the number of epochs when the validation accuracy remains unchanged) thus 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 being at the beginning or for the maximum of the wait period." Ex. 1001, 14:25-29.

A POSITA 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. Ex. 1003, ¶¶231-232. Therefore, Baker, Lorenz, and Shridhar render obvious claim 4. Ex. 1003, ¶232.

5. ***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.***

See 2 *supra*.

6. ***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.***

See 3 *supra.*

7. ***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.***

See 4 *supra.*

C. Ground 3: Claims 5 and 12 are rendered obvious over Baker, Lorenz, and Jenatton

Baker, Lorenz, and Jenatton render obvious claims 5 and 12.

1. *Overview of Ground 3*

a. Jenatton

Jenatton describes a tree data structure useable by a machine learning model to assist in selecting hyperparameters during training of a NN. Ex. 1007, *Abstract*. 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 (e.g., hyperparameter sets) that must be evaluated, using the tree-structure to inform decisions about what outputs should be evaluated in depth. Ex. 1007, 1; Ex. 1003, ¶¶131-135.

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.

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*.

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, even though the acquisition function does not explore all possibilities (i.e., does not explore all leafs of the decision tree). Ex. 1007, 6-8.

b. Motivation to Combine

A POSITA would have had numerous reasons to combine, and a reasonable expectation of success in combining, Baker, Lorenz, and Jenatton. Ex. 1003, ¶¶237-243. 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]. Ex. 1003, ¶239.

A POSITA would have used a known formulation of a 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. Ex. 1003, ¶240. A POSITA 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. *Id.* 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. *Id.*

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. Ex. 1003, ¶241. In view of the teaching in Baker of using a learning coach machine learning model to assist in training a student machine learning model, a POSITA would have used a known 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; *see also* Ex. 1003, ¶¶238, 242-243.

A POSITA 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 POSITA. Ex. 1003, ¶¶238, 242-243. 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 POSITA 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; Ex. 1003, ¶¶238, 242-243.

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 POSITA had a reasonable expectation of success in combining Baker, Lorenz, and Jenatton. Ex. 1003, ¶243.

2. *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.*

Baker, Lorenz, and Jenatton render obvious claim 5. Ex. 1003, ¶¶244-249.

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]; *see* 1[b.3], 1[c] *supra*; Ex. 1003, ¶245. 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." Ex. 1005, ¶[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 POSITA would recognize that the structure of the learning coach (e.g., the type of ML model used) would have been a known ML model structure, such as the tree-structure described in Jenatton. Ex. 1003, ¶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 with 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 "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 including data relevant to the NN being trained. A POSITA would have recognized that selecting a "most promising" leaf node for training a model included 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. Ex. 1003, ¶246.

Jenatton describes that the tree-structure is usable by a model for assisting NN training, and the model uses well-known acquisition functions for selecting hyperparameter values, including, among other examples "probability of improvement (PI)" functions. Ex. 1007, 4; Ex. 1003, ¶247. Although Jenatton focuses on expected improvement (EI), Jenatton explains how the tree-structure could be used with any acquisition function, including a PI-based acquisition function, stating that the acquisition function "plays a critical role in Bayesian Optimization as it selects anchor points by performing an exploration-exploitation trade-off." *Id*. A POSITA reviewing Jenatton would have understood that, if PI were used as the acquisition function of Baker's learning coach, 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; Ex. 1003, ¶247. Jenatton explains that either the EI or PI functions would have been usable for selection of data relevant to a current NN being trained. Lorenz suggests combining PI and EI acquisition functions to further improve performance, and a POSITA would have recognized that Lorenz's

teachings to combine these acquisition functions would have been usable with Jenatton's tree data structure. Ex. 1003, ¶¶248-249. 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 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 was an efficient structure for Baker's learning coach to identify hyperparameters relevant for training a student machine learning model. Ex. 1003, ¶¶248-249.

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.

A POSITA would have recognized that Jenatton's tree-structure was 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; *see also* Ex. 1003, ¶249. 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. Ex. 1003, ¶249. Baker, Lorenz, and Jenatton render obvious claim 5.

3. *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.*

See 5 *supra*.

VIII. CONCLUSION

Petitioner requests cancellation of the Challenged Claims.

IX. MANDATORY NOTICES UNDER 37 C.F.R. §42.8

A. Real Party in Interest Under 37 C.F.R. §42.8(b)(1)

The real parties in interest are Perplexity AI, Inc.; Perplexity AI Portugal, Unipessoal LDA; Perplexity AI Ltd.; Perplexity Management Co., LLC; and Perplexity F7 Fund I LP.

Out of an abundance of caution, Petitioner has identified all parties affiliated with the above-named Petitioner as potential real parties in interest only for the purpose of this proceeding and only to the extent that Patent Owner contends that these separate legal entities should be named real parties in interest in this IPR. Petitioner does so to avoid the potential expenditure of resources to resolve such a challenge. No unnamed entity is funding, controlling, or otherwise has an opportunity to control or direct this Petition or Petitioner's participation in any resulting IPR. Petitioner is also not aware of any affiliate that would be barred from filing this Petition under 35 U.S.C. § 315(e).

B. Related Matters Under 37 C.F.R. §42.8(b)(2)

Petitioner is not aware of any other matters relating to the '968 Patent.

C. Designation of Counsel Under 37 C.F.R. §42.8(b)(3)

Petitioner provides the following designation of counsel. Lead Counsel	Daniel J. O'Connor Reg. No. 69,023 **ALSTON & BIRD LLP** 755 Page Mill Road Building C – Suite 200 Palo Alto, CA 94304

	Phone: 650.838.2021 Fax: 704.444.1111 Email: dan.oconnor@alston.com
Backup Counsel for Petitioner	Dana Zottola Reg. No. 65,942 **ALSTON & BIRD LLP** 55 2nd Street, Suite 2100 San Francisco, CA 94105 Phone: 650.838.2000 Fax: 704.444.1111 Email: dana.zottola@alston.com S. Benjamin Pleune Reg. No. 52,421 **ALSTON & BIRD LLP** Vantage South End 1120 S. Tryon Street, Suite 300 Charlotte, NC 28203 Phone: 704.444.1000 Fax: 704.444.1111 Email: ben.pleune@alston.com Yuri Mikulka (*pro hac vice* application to be filed) Alston & Bird LLP 350 South Grand Avenue 51st Floor Los Angeles, CA 90071 Telephone: 650.838.2044 Fax: 213.576.1100 Email: yuri.mikulka@alston.com

Pursuant to 37 C.F.R. § 42.8(b)(4), Petitioner agrees to service by mail as detailed above and electronic service by email to the following email addresses:

dan.oconnor@alston.com;

dana.zottola@alston.com;

ben.pleune@alston.com;

yuri.mikulka@alston.com; and

Perplexity-Comet-IPR@alston.com.

Pursuant to 37 C.F.R §42.10(b), a Power of Attorney executed by Petitioner accompanies this Petition.

D. Service Information

Please address all correspondence and service to the addresses listed above. Petitioners consent to electronic service at the addresses above.

Date: June 5, 2025 By: /Daniel J. O'Connor/

Daniel J. O'Connor
Reg. No. 69,023
ALSTON & BIRD LLP
755 Page Mill Road
Building C – Suite 200
Palo Alto, CA 94304
Phone: 650.838.2021
Fax: 704.444.1111
Email: dan.oconnor@alston.com

CLAIMS APPENDIX

Claim Element	Claim Limitation
1pre.	A method of training a neural network (NN), the method comprising:
1a.	over a series of NN training epochs, where in each epoch the NN undergoes training and a loss is computed:
1b.	determining, using:
1b1.	a set of model parameters;
1b2.	data describing training loss of the NN; and
1b3.	a model that has been trained using training losses of a plurality of NNs other than the NN;
1c.	a probability of improvement in the loss of the NN; and
1d.	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.
8pre.	A system of training a neural network (NN), the system comprising:
8a.	a memory; and
8b.	a processor configured to:

Claim Element	Claim Limitation
8c.	over a series of NN training epochs, where in each epoch the NN undergoes training and a loss is computed:
8d.	determine, using:
8d1.	a set of model parameters;
8d2.	data describing training loss of the NN; and
8d3.	a model that has been trained using training losses of a plurality of NNs other than the NN;
8d4.	a probability of improvement in the loss of the NN; and
8e.	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.

CERTIFICATION UNDER 37 C.F.R. §42.24

Petitioners certify that the word count in this Petition is 13,969 words, as counted by the word-processing program (Microsoft Word for Office 365) used to generate this Petition, where such word count excludes the table of contents, table of authorities, mandatory notices, certificate of service, appendix of exhibits, and this certificate of word count. This Petition is in compliance with the 14,000-word limit set in 37 C.F.R. § 42.24(a)(1)(i).

Date: June 5, 2025 By: /Daniel J. O'Connor/

Daniel J. O'Connor
Reg. No. 69,023
ALSTON & BIRD LLP
755 Page Mill Road
Building C – Suite 200
Palo Alto, CA 94304
Phone: 650.838.2021
Fax: 704.444.1111
Email: dan.oconnor@alston.com

CERTIFICATE OF SERVICE

The undersigned certifies that, in accordance with 37 C.F.R. § 42.6(e) and 37 C.F.R. § 42.105, the **Petition for *Inter Partes* Review of U.S. Patent No. 11,650,968, Petitioner Perplexity AI, Inc.'s Power of Attorney; and the associated Perplexity AI Exhibits 1001-1011** were served in their entireties on the 5th day of June 2025, the same day as the filing of the above-identified documents in the United States Patent and Trademark Office/Patent Trial and Appeal Board, upon the Patent Owner by serving the correspondence address of record with the USPTO as follows: via Overnight Courier:

Pearl Cohen Zedek Latzer Baratz LLP
7 Times Square
19th Floor
New York, NY 10036
Telephone: (415) 480-4100

Dated: June 5, 2025

Respectfully submitted,

/Daniel J. O'Connor/
Daniel J. O'Connor
Reg. No. 69,023
dan.oconnor@alston.com
ALSTON & BIRD LLP
755 Page Mill Road, Suite C-200
Palo Alto, CA 94304
Lead Counsel for Petitioner
Perplexity AI, Inc.