Modeling Machine Learning[*]

Andrew Caplin[†], Daniel Martin[‡], and Philip Marx[§]

October 15, 2022

Abstract

What do machines learn, and why? To answer these questions we import models of human cognition into machine learning. We propose two ways of modeling machine learners based on this join: feasibility-based and cost-based machine learning. We evaluate and estimate our models using a deep learning convolutional neural network that predicts pneumonia from chest X-rays. We find these predictions are consistent with our model of cost-based machine learning, and we recover the algorithm's implied costs of learning.

Key words: Algorithms, machine learning, information frictions, information economics, rational inattention
JEL codes: D83, D91

[*]We thank Emir Kamenica, Stephen Morris, Nick Netzer, Marek Pycia, Doron Ravid, Jakub Steiner, and audiences at the University of Zurich, the European Summer Symposium in Economic Theory in Gerzensee, and the Sloan-NOMIS Summer School on Cognitive Foundation of Economic Behavior. Caplin thanks the NOMIS and Sloan Foundations for their support for the broader research program on the cognitive foundations of economic behavior.

[†]Department of Economics, New York University.

[‡]Kellogg School of Management, Northwestern University and Department of Economics, University of California, Santa Barbara.

[§]Department of Economics, Louisiana State University.

1 Introduction

Machine learning is increasingly central to the modern economy. Virtually all industries, jobs, and consumer experiences have been impacted in some way by the rapid rise in automation brought about by this technology. Economically important applications of machine learning include what ads to serve, what content to show, what coupons to provide, facial recognition, translation, voice assistants, credit scoring and loan decisions, medical decisions, product recommendations, driving routes, spam filters, fraud detection, and so on.

Yet in the push to improve the accuracy of machine learning predictions, state-of-the-art machine learning algorithms have reached a level of complexity in which they are effectively black boxes. Our understanding of the most recent machine learning algorithms is obscured by their stochastic choice among local solutions to non-convex problems, nested training protocols, and model architectures involving a large number of parameters. As a result, even if an analyst knows all of the code behind such an algorithm, it is nearly impossible to fully grasp its inner workings.

Because a complete *as is* understanding of modern machine learning algorithms is increasing out of reach, a natural question is whether there exists a parsimonious *as if* representation of opaque machine learners that can reasonably approximate their behavior. This question is in the spirit of a long literature in economics, psychology, and neuroscience that aims to generate parsimonious representations that reasonably approximate the behavior generated by the black box of human cognition.

In fact, our approach to modeling machine learners exactly mirrors how human learners are often modeled in economics, psychology, and neuroscience: as individual decision makers who engage in signal gathering, belief formation, and choice. Specifically, we assume that when facing a test set of observations, a machine learner engages in a two stage decision process: first, it optimally chooses a signal structure, and second, it chooses predictions that minimize expected losses given the posterior beliefs generated by signal realizations.

We propose two machine learning models that build on this information-theoretic foundation. Both models are identical in terms of second stage choice, but differ in the constraints that drive the choice of signal structure in the first stage. In one model, which we call *feasibility-based machine learning*, the algorithm chooses among a feasible set of ways to learn about observations in the test set. In our other model, which we call *cost-based machine learning*, the algorithm adjusts its learning in response to additional, unobservable costs of learning. These are not the monetary costs incurred in running the algorithm, instead they reflect the intrinsic difficulty the algorithm has in learning the true outcome given its training procedures and the available training data.

We evaluate and estimate our models using CheXNeXt, an influential deep learning convolutional neural network for predicting thoracic diseases from chest X-ray images (Rajpurkar, Irvin, Ball, et al. 2018). We vary the algorithm's loss function by varying its class weights, which dictate the relative value a loss function places on correct and incorrect predictions across different class labels. Rajpurkar, Irvin, Ball, et al. (2018) follow standard practice in using class weights to increase the value that the algorithm places on mistakes made for observations in the underrepresented class, which is important given that only 1.3% of chest X-rays are in fact labeled with pneumonia.

One impact of varying class weights is that it varies the machine's incentives for making different predictions. We first show that this algorithm distorts its predictions in line with these incentives precisely as predicted in our models. In addition, as we vary the algorithm's class weights we also vary the machine's incentives for learning. Feasibility-based machine learning requires that what the algorithm learns under one set of class weights is preferred to what it learns under alternative class weights, since the alternative ways of learning are also feasible and thus could have been chosen. Cost-based learning makes a related prediction, but net of additional costs of learning. We find that CheXNeXt predictions are consistent with cost-based learning, but not feasibility-based learning. This arises because of a common preference for what is learned across class weights, which violates feasibility-based learning but can be rationalized with recourse to additional structural costs of learning inherent to the algorithm. In addition, we recover sharp bounds on the structural cost parameters and construct "representative" learning costs for the algorithm. These bounds and representative costs suggest that by placing relatively lower weight on pneumonia instances the algorithm incurs higher learning costs, which is sensible given that doing so places relatively higher weight on non-pneumonia instances, which are far more common.

Our approach provides a natural join between three growing literatures: information economics, bounded rationality, and machine learning. The information-theoretic foundation of our models mirrors the core objects of information design (Kamenica and Gentzkow 2011; Kamenica 2019; Bergemann and Morris 2017),[1] and we leverage a wide range of tools recently introduced for modeling boundedly rational decision making. Our feasibility-based machine learning model is an application of the capacity constrained learning model characterized by Caplin, Martin, and Marx (2022), which generalizes the fixed capacity version of rational inattention theory proposed by Sims (2003) and the noisy cognition model proposed by Woodford (2014). Our cost-based machine learning model is an application of the general version of rational inattention theory characterized by Caplin and Dean (2015), which itself generalizes the specialized version using Shannon entropy characterized by Matejka and McKay (2015) and Caplin, Dean, and Leahy (2017). Thus, our paper is connected to both a growing literature that considers stochastic choice to be essential for studying lim-

[1]Liang, Lu, and Mu (2022) draw a point of connection between information design and machine learning to study the tradeoffs between accuracy and fairness.

ited attention in human decision making (e.g., Manzini and Mariotti 2014; Cattaneo, Ma, Masatlioglu, and Suleymanov 2020) and a growing literature that studies the theoretical properties of costly learning (e.g., Gentzkow and Kamenica 2014; De Oliveira, Denti, Mihm, and Ozbek 2017; Hébert and Woodford 2017; Denti 2022; Lipnowski and Ravid 2022). In this paper we demonstrate that the join between information economics, bounded rationality, and machine learning approaches can, in fact, produce models that reasonably approximate the behavior of opaque machine learners.

The rest of the paper proceeds as follows. Section 2 introduces the preliminaries of our model and empirical application. Section 3 covers the foundation for our learning models. Building on this foundation, Section 4 covers the feasibility- and cost-based machine learning models and cost recovery. An empirical application that tests the foundations and learning models and performs recovery in a state-of-the-art deep convolutional neural network is treated concurrently at the end of the corresponding sections. Section 5 concludes with a discussion of related literatures.

2 Preliminaries

2.1 Setup and Notation

There is a finite set of *instances* X with generic element x, a finite set of *outcomes* Y with generic element y, and a deterministic map between instances and outcomes $f : X \to Y$ that is called the *ground truth*. In our running application, the set of instances X is 112,120 chest X-ray images, the set of possible outcomes $Y = \{0, 1\}$ is an indicator for the presence of pneumonia, and the ground truth is the actual outcome (pneumonia or not) corresponding to each image.[2]

Algorithms generate predictions about each instance, and we consider a generic set of *predictions* A to accommodate an array of possibilities. For example, many classification algorithms output a numeric *confidence score* for each instance and possible outcome. In turn, confidence scores can be translated into discrete outcome predictions using a downstream classification rule, such as predicting an outcome if its confidence score exceeds a given threshold or the confidence scores of the other possible outcomes. In our application, we consider numeric confidence scores instead of a discrete downstream outcome predictions because confidence scores are more closely tied to machine incentives and yield stronger tests and sharper identification. Our framework also accommodates situations where the analyst

[2]Specifically, Wang et al. (2017) provides the ground truth by labeling whether each chest X-ray indicates pneumonia or not. As is standard when evaluating algorithms, we assume this ground truth is correct, but our approach could be extended to include uncertainty about the ground truth.

only has data on predicted outcomes or wishes to model the scoring algorithm jointly with a downstream classification rule.

An important input to the training of an algorithm is a *loss function* $L : A \times Y \to \mathbb{R}$, which indicates the value of a particular prediction given the outcome. It is standard practice to use cross-entropy $L(a, y) = -y \log a - (1 - y) \log(1 - a)$ over confidence scores and outcomes when training deep learning neural networks to predict the class to which an instance belongs. In addition, it is also standard practice to re-weight the loss function to make losses higher or lower for a particular outcome; such class weighting is often employed when one outcome is less common (Thai-Nghe, Gantner, and Schmidt-Thieme 2010) or with the hope of achieving some external objective (Zadrozny, Langford, and Abe 2003).

For a given algorithm, using loss function L in training generates a *trained model* $g^L : X \to A$.[3] The performance of the trained model is assessed on how well its predictions align with actual outcomes, as specified by the ground truth.[4] Because the characteristics of instances can vary widely for a particular outcome, performance is typically evaluated on how well it performs on aggregate for each outcome. Formally, aggregate level performance for each outcome is summarized by *performance data* $P^L : A \times Y \to [0, 1]$, which is the joint distribution of predictions and outcomes for the trained model in a test data set,

$$P^L(a, y) = \frac{|\{x \in X | g^L(x) = a \;\&\; f(x) = y\}|}{|X|}.$$

Because X is finite, $supp(P^L_A)$, is also finite. As is standard practice in the machine learning literature, we study algorithmic predictions over a test sample that is independently drawn from the same population as the data on which the algorithm is trained.

2.2 Application: Data and Algorithm

We demonstrate the potential empirical suitability of our machine learning models by implementing CheXNeXt, an influential deep convolutional neural network for predicting thoracic diseases from chest X-ray images (Rajpurkar, Irvin, Ball, et al. 2018). Our training models are generated using the ChestX-ray14 data set, which consists of 112,120 frontal chest X-rays which were synthetically labeled with the presence of fourteen thoracic diseases (Wang et al. 2017). The main modifications we make to the CheXNeXt training procedure are that we isolate the task of pneumonia detection as in the earlier implementation of Rajpurkar, Irvin, Zhu, et al. (2017) and we train the algorithm across various β-weighted cross-entropy loss

[3]For stochastic algorithms that can generate different predictions for the same input, we take g^L to be an average prediction for each instance across trained models, which is called an *ensemble* model.

[4]Our approach can be applied to any set X, but since algorithmic performance is typically evaluated on a hold-out or test set of instances, a natural interpretation of X is that it is the test set.

functions:

$$L^{\beta}(a,y) = -\beta y \log(a) - (1-\beta)(1-y)\log(1-a). \tag{1}$$

Specifically, we vary the loss function by considering $\beta = 0.7, 0.9, 0.99$.[5] In addition, we employ ensemble (model-averaging) methods to isolate the substantive effects of what the machine learns from random noise inherent to the stochastic training procedure. Using nested cross-validation methods, this yields an ensemble model prediction at each β for each of the 112,120 X-ray images in the original data. Further technical details of our training procedure are relegated to Appendix A. We return to our application in Subsection 3.3 after introducing our fundamental representation of machines as Bayesian expected loss minimizers.

3 Machines as Bayesian Expected Loss Minimizers

In this section we present the information-theoretic foundation of the learning models we consider, the testable implications of this foundation, and positive evidence of this foundations in our empirical application.

In this foundation we follow the Blackwell (1953) model of experimentation, signal processing, and choice. For both feasibility-based and cost-based machine learning, we model an algorithm as an optimizing agent that i) starts with a prior $\mu \in \Delta(Y)$ over outcomes, ii) gets signal realizations that provide information about the outcome, iii) forms posterior beliefs $\gamma \in \Delta(Y)$ via Bayesian updating, and iv) chooses predictions based on these posteriors to minimize expected losses. As in Kamenica and Gentzkow (2011) we define $\mathcal{Q}$ as those distributions of posteriors with finite support that satisfy Bayes' rule,

$$\mathcal{Q} \equiv \{Q \in \Delta(\Delta(Y)) | \sum_{\gamma \in supp(Q)} \gamma Q(\gamma) = \mu\}.$$

Posteriors are translated into probabilistic predictions through a prediction function $q : supp(Q) \to \Delta(A)$. For a given loss function L and distribution of posteriors $Q \in \mathcal{Q}$, the set of optimal prediction functions is defined as,

$$\hat{q}(L,Q) \quad \equiv \quad \underset{q}{\operatorname{argmin}} \sum_{\gamma \in supp(Q)} Q(\gamma) \sum_{a \in A} q(a|\gamma) \sum_{y \in Y} \gamma(y) L(a,y).$$

Note that any pair (Q,q) produces a joint distribution of predictions and outcomes given by $P_{(Q,q)} : A \times Y \to [0,1]$ where,

$$P_{(Q,q)}(a,y) \equiv \sum_{\gamma \in supp(Q)} Q(\gamma) q(a|\gamma)\gamma(y).$$

With these elements in place we can define the foundation of our subsequent learning models.

[5]For reference, the class weight used in the analysis of Rajpurkar, Irvin, Zhu, et al. (2017) is approximately 0.99 because the probability of positive pneumonia cases in the data set is 0.0127.

Definition 1. *For a given loss function L, P^L has a* ***signal-based representation*** *(SBR) if there exists a prior $\mu \in \Delta(Y)$, a Bayes consistent distribution of posteriors $Q \in \mathcal{Q}$, and a prediction function $q : supp(Q) \to \Delta(A)$ such that:*

1. *The prior is correct: $\mu(y) = \sum_{a \in supp(P_A^L)} P^L(a, y)$.*
2. *Predictions are optimal at all possible posteriors: $q \in \hat{q}(L, Q)$.*
3. *Predictions are generated by the model: $P^L(a, y) = P_{(Q,q)}(a, y)$.*

If P^L has an SBR, then it is *as if* the algorithm makes predictions to minimize the loss function given the Bayesian posterior beliefs induced by its signal structure.

3.1 Testable Condition: Loss Calibration

Applying the theoretical results of Caplin and Martin (2015) and Bergemann and Morris (2016), it is straightforward to show that the SBR foundation is characterized by a simple condition, called *loss calibration*, which requires that switching wholesale from any prediction a to any alternative prediction a' would never strictly reduce losses.[6]

Definition 2. *Performance data P^L is* ***loss-calibrated*** *to loss function L if a wholesale switch of predictions does not reduce losses according to L:*

$$a \in \underset{a' \in \mathbb{R}^n}{\operatorname{argmin}} \sum_{y \in Y} P^L(a, y) L(a', y) \text{ for all } a \in supp(P_A^L). \tag{2}$$

Any algorithm that fails this condition makes predictions that are not suitable for the loss function and that are thus inconsistent with an SBR and Bayesian expected loss minimization.[7]

Under weighted cross-entropy loss with binary outcomes (1), it is straightforward to show that loss calibration takes a unique closed form as a function of posterior probabilities. Let $a^\beta(\gamma)$ be the optimal prediction when the class weight is β and the posterior probability that the outcome is $y = 1$ is given by γ.

Observation 1. *Consider weighted cross-entropy loss (1). For any weight $\beta \in (0, 1)$ and all posterior probabilities $\gamma \in supp(Q)$ that the outcome is $y = 1$, the unique loss-calibrated confidence score is given by:*

$$a^\beta(\gamma) = \frac{\beta\gamma}{1 - \beta - \gamma + 2\beta\gamma}. \tag{3}$$

[6] To the best of our knowledge, no such condition has been proposed in the machine learning literature.

[7] Nevertheless, this is easy to rectify. Whenever this loss function is input into the algorithm, a single line of code at the end of the computer program making a wholesale switch to predicting a whenever it would have predicted a' would make this algorithm loss-calibrated for this loss function.

In the case of unweighted cross-entropy loss ($\beta = 0.5$), the loss-calibrated scoring function (3) collapses to the optimal prediction being the posterior probability itself, $\alpha^{0.5}(\gamma) = \gamma$. Thus, as is well-known, unweighted cross-entropy incentivizes truthful revelation of beliefs.

3.2 Calibration in Machine Learning: Theory and Practice

In general, any *proper* loss function incentivizes truthful reporting under an SBR. Thus, an observable implication of SBR is that if the loss function is proper, an algorithm should be (unconditionally) *calibrated* to ground truth probabilities. That is, each confidence scores should equal the true probability of the outcome given that score. This form of calibration is an important and much-studied property in machine learning because calibrated predictions correctly reflect their uncertainty.

While there is no inherent tension between the aims of accuracy and calibration in our *as if* model of the machine, the relationship is more nuanced in the *as is* practice of machine learning. This is due to machine learning's extrapolative nature: a machine learns a model from a training sample of data in order to predict new instances outside the training sample. Furthermore, the training sample is often small relative to the potential complexity of the model.[8] The machine must therefore balance the competing aims of capturing the essential features of the training data (not underfitting) while remaining generalizable to new instances (not overfitting). Theoretically, this problem of optimal out-of-sample learning from limited data can be cast as an optimal tradeoff between bias and variance — respectively, how far predictions depart systematically from truth, and how much predictions depend on the idiosyncrasies of the training data.[9]

An optimal resolution of the bias-variance tradeoff often involves introducing bias into the predictive model in order to reduce variance.[10] Introducing bias may also introduce miscalibration, for example in the case of lasso or ridge regressions that penalize coefficient size and thereby generate *underconfident* models whose predictions are biased toward the overall mean, i.e. shrinkage bias.[11] At the other extreme, miscalibration may also arise

[8]For example, in our application, a data set of 112,120 images is used to train and evaluate an ensemble of neural networks, each of which has 6,968,206 learnable parameters (Rajpurkar, Irvin, Ball, et al. 2018).

[9]The tradeoff is clearest for squared (reducible) error, which can be precisely decomposed into a sum of squared bias and variance; e.g., see Hastie, Tibshirani, and Friedman (2009). For an extension to other loss functions, see also, e.g., Domingos (2000).

[10]This is a central distinction of machine learning relative to the regression-based methods traditionally employed in economics to address questions of causal inference (Kleinberg, Ludwig, Mullainathan, and Obermeyer 2015).

[11]Formally, miscalibration implies bias but bias does not imply miscalibration. A simple example of bias without miscalibration is extreme shrinkage bias: a constant prediction of the average training outcome, regardless of features.

from high variance, for example in the case of *overconfident* models whose out-of-sample confidence scores are biased toward the extremes due to overfitting of the training data. Since the machine aims to balance the competing objectives of reducing bias and variance, calibration and its relationship to accuracy are ultimately empirical questions.

Deep learning convolutional neural networks have been shown to suffer from miscalibration, specifically overconfidence, with the severity of miscalibration increasing in model size (Guo, Pleiss, Sun, and Weinberger 2017). Along with the good calibration properties of earlier and simpler neural networks (Niculescu-Mizil and Caruana 2005), this seems to suggest that miscalibration in practice is an inevitable cost of improvements in accuracy stemming from increasing model complexity.[12] However, calibration of deep neural nets is improved with regularization procedures such as weight decay (Guo, Pleiss, Sun, and Weinberger 2017) as well as model ensembling (Lakshminarayanan, Pritzel, and Blundell 2017). This is consistent with the intuition that both of these procedures reduce overconfidence. Model ensembling is also well-known to improve model accuracy (e.g. Dietterich 2000), which suggests there need not be a tension between accuracy and calibration per se.

The most recent evidence further supports that there is no inherent tradeoff between accuracy and calibration. In particular, Minderer et al. (2021) conduct a comprehensive comparison of 180 image classification models and find that the most accurate current models, such as non-convolutional MLP-Mixers (Tolstikhin et al. 2021) and Vision Transformers (Dosovitskiy et al. 2021), are not only well-calibrated compared to earlier models, but also that their calibration is more robust to distributions that differ from training. This body of evidence suggests that future improvements in model accuracy might only benefit calibration. Thus, our SBR representation is likely to remain a useful foundation for modeling machines even as (and perhaps because) they become increasingly complex.

3.3 Application: Loss Calibration

In the deep learning algorithm we consider — which regularizes through early stopping and aggregates over an ensemble of trained neural nets — we find that confidence scores are loss-calibrated as in (3) across various weights in weighted cross entropy loss (1). Recall that this includes unconditional calibration for unweighted cross-entropy loss ($\beta = 0.5$), which is a proper loss function.

Graphical evidence of calibration and loss calibration is provided in the left and right panels of Figure 1, respectively. In each plot, the horizontal axis represents the confidence score,

[12] The relationship between accuracy and model complexity is itself more complex than suggested by the classical bias-variance tradeoff. A recent finding is that, as a function of model complexity, accuracy may eventually increase again for sufficiently complex models that perfectly interpolate the training data (Belkin, Hsu, Ma, and Mandal 2019).

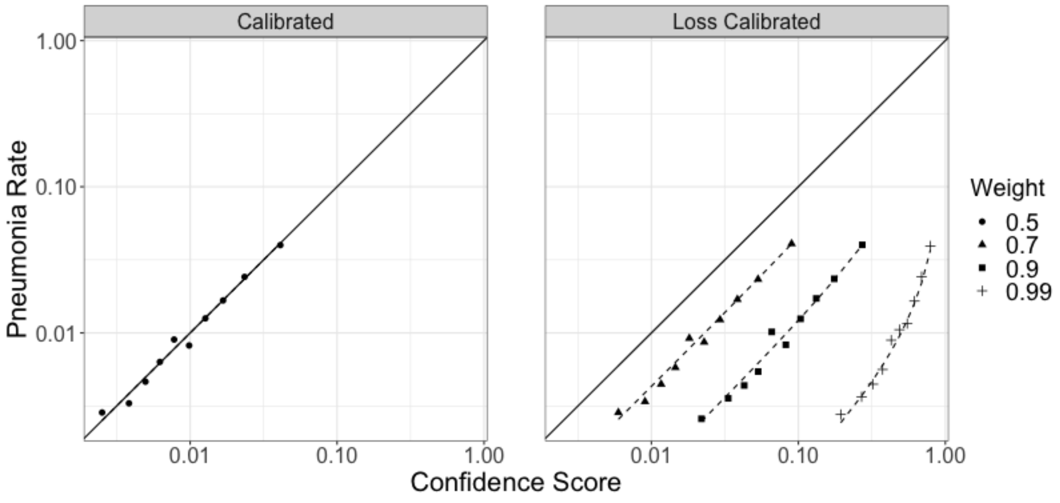

Figure 1: Theoretical relationship between confidence score and pneumonia rate for a loss-calibrated algorithm with calibration target (solid lines), loss calibration targets varying class weights (dashed lines), and empirical decile-binned calibration curves (shapes) for the pneumonia-detection algorithm presented in Subsection 2.2. All objects are displayed on a log-10 scale to improve readability. This figure provides visual evidence that the algorithm is calibrated for $\beta = 0.5$ (left panel) and loss-calibrated generally (left and right panels). This ensures an SBR representation and simplifies computation of the objects introduced in Section 4.

and the vertical axis the corresponding pneumonia rate in the data (both on a log scale). The shapes in the figure provide the empirical decile-binned calibration curves (DeGroot and Fienberg 1983; Niculescu-Mizil and Caruana 2005). The solid lines represent the situation where confidence scores are calibrated. Thus, we observe the algorithm appears effectively calibrated for the unweighted loss function $\beta = 0.5$, and miscalibrated otherwise. The dashed lines in the right plot show the theoretical relationship between scores and pneumonia rates for the relative positive class weights $\beta = 0.7, 0.9, 0.99$ if an algorithm is loss-calibrated (note that the loss-calibrated and calibrated lines coincide on the left when $\beta = 0.5$). As β increases, the algorithm is increasingly incentivized to provide a score that is higher than the machine's actual "belief" about the probability of pneumonia, which causes the lines to bow out. The alignment of theoretical predictions and empirical estimates strongly suggests that the algorithm is generally very close to being loss-calibrated, and very close to being calibrated when the loss function is unweighted.

Finally, note that our finding of calibration with an unweighted loss function is consistent with previously documented calibration for deep learning convolutional neural networks that use regularization (i.e., weight decay in Guo, Pleiss, Sun, and Weinberger 2017) and deep ensembling (Lakshminarayanan, Pritzel, and Blundell 2017). With a viable SBR in hand, we now turn to our models of what the machine learns.

4 Models of Machine Learning

SBR leaves open the question of how a machine learning algorithm arrives at its signal structure – that is, what the machine decides to learn based on the incentives provided by the loss function. We now propose and characterize two nested alternatives: choosing among a set of feasible signal structures or choosing among signal structures of different costs.

4.1 Feasibility-Based Machine Learning

Our first model class assumes the algorithm chooses among a set of feasible signal structures to best match the incentives provided by the loss function.

To translate this into the SBR framework of Section 3, we define a feasible set of experiments $\mathcal{Q}^* \subset \mathcal{Q}$. This feasible set depends only on the algorithm's capability and is not specific to the loss function provided. We define the algorithm's strategy space Λ to include both Q and q:

$$\Lambda = \{(Q, q) | Q \in \mathcal{Q}, q : supp(Q) \to \Delta(A)\}.$$

For a given loss function L and feasible set $\mathcal{Q}^*$, the set of optimal strategies $\tilde{\Lambda}(L, \mathcal{Q}^*)$ is,

$$\tilde{\Lambda}(L, \mathcal{Q}^*) \equiv \operatorname*{arg\,inf}_{(Q,q)\in\Lambda, Q\in\mathcal{Q}^*} \sum_{\gamma\in supp(Q)} Q(\gamma) \sum_{a\in A} q(a|\gamma) \sum_{y\in Y} \gamma(y) L(a, y).$$

With this we can define all performance data sets that are consistent with optimality for a given feasible set $\mathcal{Q}^*$ as,

$$\tilde{P}(L, \mathcal{Q}^*) \equiv \{P_{(Q,q)} | (Q, q) \in \tilde{\Lambda}(L, \mathcal{Q}^*)\}.$$

Feasibility-based machine learning requires that there exist a feasible set $\mathcal{Q}^*$ such that the performance data produced by an algorithm are optimal given that feasible set for all $L \in \mathcal{L}$.

Definition 3. *An algorithm is consistent with* ***feasibility-based machine learning*** *if there exists a feasible set $\mathcal{Q}^* \subset \mathcal{Q}$ such that $P^L \in \tilde{P}(L, \mathcal{Q}^*)$ for all $L \in \mathcal{L}$.*

4.2 Cost-Based Machine Learning

Our second model class assumes the algorithm chooses among signal structures of different costs. Once again, these are not the monetary costs incurred in running the algorithm, instead they reflect the intrinsic difficulty the algorithm has in learning the true outcome given its training procedures and the available training data. One way to interpret these costs is as the resource costs of different mathematical operations.

To formalize this, we define a learning cost function $K : \mathcal{Q} \to \mathbb{R}$ and denote the set of all possible learning cost functions as $\mathcal{K}$. An algorithm's learning cost function depends only on its capabilities and is not specific to the loss function provided.

Given loss function $L \in \mathcal{L}$ and learning cost function $K \in \mathcal{K}$, the *resource-adjusted loss* $\hat{L}$ of strategy (Q, q) is,

$$\hat{L}((Q,q)|L,K) \equiv \sum_{\gamma \in supp(Q)} Q(\gamma) \sum_{a \in A} q(a|\gamma) \sum_{y \in Y} \gamma(y) L(a,y) + K(Q).$$

The corresponding set of optimal strategies $\hat{\Lambda}(L, K)$ is then defined as,

$$\hat{\Lambda}(L,K) \quad \equiv \quad \underset{(Q,q)\in\Lambda}{\arg\inf} \ \hat{L}((Q,q)|L,K).$$

This optimization problem formalizes the way in which the algorithm trades off losses with learning costs. Given any $L \in \mathcal{L}$ the set of all performance data sets that are consistent with optimality for a given learning cost function $K \in \mathcal{K}$ are,

$$\hat{P}(L,K) \equiv \{P_{(Q,q)} | (Q,q) \in \hat{\Lambda}(L,K)\}.$$

Definition 4. *An algorithm is consistent with* ***cost-based machine learning*** *if there exists a learning cost function $K \in \mathcal{K}$ such that $P^L \in \hat{P}(L,K)$ for all $L \in \mathcal{L}$.*

The second learning model generalizes the first model because a feasible set of posterior distributions $\mathcal{Q}^*$ is equivalently specified as a learning cost function K^* for which the cost is zero for every feasible posterior distribution $Q \in \mathcal{Q}^*$ and infinite otherwise.

4.3 Testing the Models

In order to simplify the characterizations of these models, we restrict consideration to performance data sets with an SBR representation (or its empirically verifiable counterpart, loss calibration). Because indexing will be useful in what follows, we will take as given a finite set of M loss functions, indexed by $1 \leq m \leq M$. For notational simplicity, we denote the performance data set from training the algorithm with the m-th loss function as $P^m = P^{L^m}$.

Our characterization of feasibility-based learning is taken directly from Caplin, Martin, and Marx (2022), who characterize capacity constrained learning for human decision makers. In the context of machine learning, the key idea is to ensure that losses cannot be lowered by counterfactually switching to the predictions from training with a different loss function. All such comparisons are visible in the *cross-loss matrix* G with generic element G^{mn} in row m and column n that specifies the minimized expected losses when the loss function is L^m and the performance data is P^n:

$$G^{mn} \equiv \sum_{a \in supp(P_A^n)} \min_{a' \in A} \sum_{y \in Y} L^m(a', y) P^n(a, y).$$

The operation on the right hand side of the equation takes any prediction $a \in supp(P_A^n)$, picks some alternative prediction $a' \in A$ to replace it wholesale, computes the corresponding expected losses for L^m, and then minimizes.

A feasibility-based machine learning representation requires that no such switch of performance data can lower losses. To formalize we define the $M \times M$ *direct loss difference matrix* D_0 by,

$$D_0^{mn} \equiv G^{mn} - G^{mm}. \tag{4}$$

An algorithm with an SBR is *strongly loss-adapted* if for all $1 \leq m, n \leq M$,

$$D_0^{mn} \geq 0, \text{ or equivalently } G^{mn} \geq G^{mm}.$$

The results in Caplin, Martin, and Marx (2022) can be used to show that, together with loss-calibration, an algorithm being strongly loss-adapted is necessary and sufficient for feasibility-based machine learning. To apply their results one maps utility maximization to loss minimization and action sets to loss functions.

The corresponding characterization of cost-based learning is based on paths of switches. Define $H(m, n)$ as all sequences of indices $\vec{h} = (h(1), h(2), \ldots, h(J(\vec{h}))$ in which the first $J(\vec{h}) - 1$ entries are distinct. The *indirect loss-difference matrix* D computes minimizing summed loss differences on such paths

$$D^{mn} \equiv \min_{\{\vec{h} \in H(m,n)\}} \sum_{j=1}^{J(\vec{h})-1} D_0^{h(j)h(j+1)}. \tag{5}$$

Formally, an algorithm with an SBR is *loss-adapted* if for all $1 \leq m \leq M$,

$$D^{mm} \geq 0.$$

This condition requires that no cycle of switches could lower losses. Applying the results in Caplin and Dean (2015), an algorithm with an SBR has a cost-based explanation if and only if it is loss-adapted.

Caplin, Martin, and Marx (2022) show that if an algorithm is loss-adapted, then the loss-difference matrix can be computed in polynomial time by applying the Floyd-Warshall algorithm to the complete weighted directed graph with weight D_0^{mn} on the directed edge from node $1 \leq m \leq M$ to node $1 \leq n \leq M$. Loss adaptedness is easily verified by the Floyd-Warshall procedure: an algorithm is loss-adapted if and only if the candidate matrix thus computed has a zero diagonal.

4.4 Qualifying Costs of Learning

Building on Caplin and Dean (2015), Caplin, Martin, and Marx (2022) show how to identify all *qualifying* costs $\{K^m\}_{m=1}^M$ that rationalize the observed performance data according to cost-based learning, in that costs are minimized by choosing performance data set P^m at cost K^m when the loss function is L^m

$$D_0^{mm} + K^m \leq D_0^{mn} + K^n, \tag{6}$$

for all $1 \leq m, n \leq M$. For present purposes, a key observation is that, normalizing to $K^M = 0$, qualifying learning costs $\{K^m\}_{m=1}^M$ define a convex polyhedron in $\mathbb{R}^{M-1}$ with the sign-inverted M-th row $(-D^{M1}, \cdots, -D^{M(M-1)})$ and M-th column $(D^{1M}, \cdots, D^{(M-1)M})$ providing a subset of the extreme points. Furthermore, the average cost $\bar{K}$ across normalizations is potentially appealing as a representative learning cost because it is "central," qualifying, and easy to compute.[13]

4.5 Application

The main products of our empirical application are estimates of the cross-loss and indirect loss-difference matrices G and D introduced in the previous Subsection 4.3. To facilitate their computation, we rely on the strong evidence for an SBR representation provided in Subsection 3.3. This strong evidence of loss calibration allows us to compute cross-losses in the G matrix, as given below, by directly recalibrating confidence scores to recover optimal confidence scores across weights.[14]

[13]Denti (2022) provides a linear program that recovers the minimum learning cost function in which not learning is free. For human decision-makers, it is natural to assume that inattention is free, but this assumption is less natural for algorithms. Also, the minimal monotone learning cost might be quite different from the rest of the qualifying learning costs, and so might not be very representative.

[14]In turn, this analytical mapping circumvents the need to bin data to recover posterior beliefs, avoiding the finite sample issues associated therewith. Note, however, that the evidence of loss calibration still involved binning the data into deciles.

$$G = 0.01 \begin{array}{c} \\ \left(\begin{array}{ccc} P^{0.7} & P^{0.9} & P^{0.99} \\ 3.750 & 3.752 & 3.762 \\ 3.349 & 3.352 & 3.362 \\ 1.365 & 1.366 & 1.370 \end{array}\right) \end{array} \begin{array}{c} \\ L^{0.7} \\ L^{0.9} \\ L^{0.99} \end{array}$$

Recall that a necessary and sufficient condition for feasibility-based machine learning is that the algorithm is strongly loss-adapted:

$$\mathcal{H}_0 : \; D_0^{mn} \equiv G^{mn} - G^{mm} \geq 0 \quad \text{for all } 1 \leq m, n \leq M.$$

In order to statistically test this multivariate one-sided hypothesis, we first estimate a 9×9 covariance matrix for the cross-loss matrix G via 10,000 bootstrap samples from the data set of ensemble predictions. We then compute p-values for the constituent univariate one-sided Wald tests and apply a Bonferroni correction. Even using this conservative approach to bounding the family-wise error rate, we reject the null hypothesis at standard levels of significance with $p = 0.0014$. Further inspection of G reveals a systematic reason for why we reject the null hypothesis: loss functions have a common preference for the performance data from training with lower β. Thus, while the predictions for the loss function with weight $\beta = 0.7$ are consistent with the algorithm being strongly loss-adapted, the predictions for the loss functions with weight $\beta = 0.9, 0.99$ are not.

Our second question is whether the algorithm is loss-adapted, and thereby consistent with cost-based learning:

$$\mathcal{H}_0 : \; D^{mm} \geq 0 \quad \text{for all } 1 \leq m \leq M.$$

The estimated D matrix, given below, satisfies this null hypothesis.

$$D = 0.1^5 \begin{array}{c} \\ \left(\begin{array}{ccc} P^{0.7} & P^{0.9} & P^{0.99} \\ 0 & 2.548 & 12.467 \\ -2.529 & 0 & 9.938 \\ -6.993 & -4.463 & 0 \end{array}\right) \end{array} \begin{array}{c} \\ L^{0.7} \\ L^{0.9} \\ L^{0.99} \end{array}$$

We therefore fail to reject that the algorithm is consistent with cost-based learning at any level of significance.[15] Intuitively, a necessary condition for this is that, even though all loss functions are minimized by switching to lower-β performance data, the gains from switching are lower for higher-β loss functions.

[15]Pointwise consistency with loss-adaptedness is satisfied in 53% of our bootstrap samples.

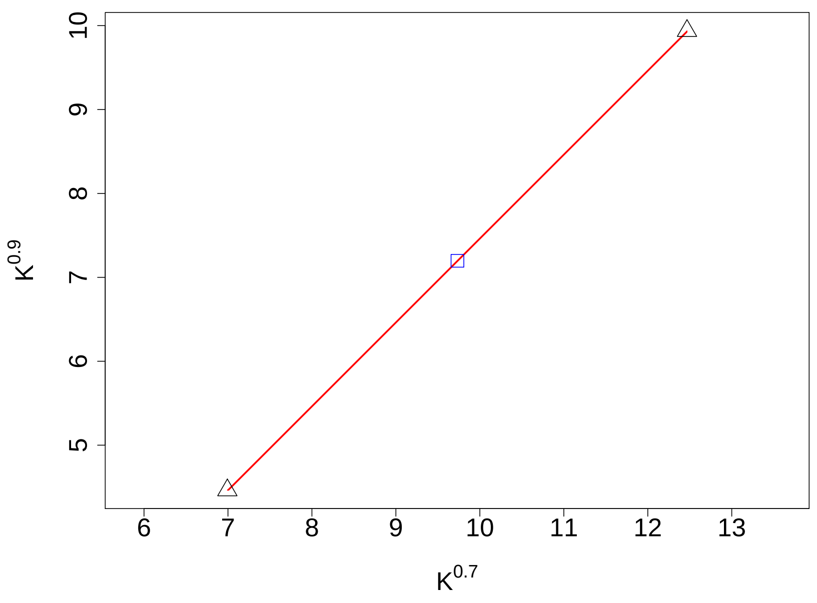

Figure 2: Recovery of cost polyhedron (red) when $K^{0.99}$ is normalized to zero, with two extreme points (triangles) and the representative cost (square) marked. The relationship between $K^{0.7}$ and $K^{0.9}$ is nearly point identified by the data. Furthermore, any rationalizing cost function has the feature that $K^{0.99} \leq K^{0.9} \leq K^{0.7}$. For legibility, the axis values are re-scaled by a factor of 10^5.

Because the algorithm in our application is consistent with cost-based learning as in Section 4, we can also recover all qualifying costs. Normalizing $K^{0.99} = 0$, the cost polyhedron for remaining costs $K^{0.7}$ and $K^{0.9}$ is illustrated in Figure 2. The figure also plots the pair of extreme costs identified by D (triangles) and the representative cost (square). Given that the value of performance data is decreasing in the β of the weighted loss function, the costs of learning must also be decreasing in β in order to rationalize the observed choices.

Finally, we conclude with a brief observation on the power of our test for cost-based learning: *any* reordering of chosen information structures would have resulted in a pointwise rejection of pairwise loss-adaptedness, and therefore also of full loss-adaptedness. This simple reasoning means that we failed to reject the null hypothesis in spite of — rather than in the absence of — a powerful test.

5 Discussion

Machine learning is increasingly central both to the modern economy and to the field of economics itself, where it has yielded improvements in policy-relevant predictions (Kleinberg, Ludwig, Mullainathan, and Obermeyer 2015), causal inference with high-dimensional data (Belloni, Chernozhukov, and Hansen 2014, Athey 2017), and the analysis of rich new sources of data (Gentzkow, Kelly, and Taddy 2019).[16] Furthermore, machine learning has been usefully applied in microeconomic theory, for example as a complement to economic theory (Fudenberg and Liang 2019) and as a benchmark of model completeness (Fudenberg, Kleinberg, Liang, and Mullainathan 2022).

However, state-of-the art machine learning algorithms lack what Lipton (2018) refers to as *algorithmic transparency*: an understanding of why an algorithm chooses the prediction model that it does. For example, standard OLS has high algorithmic transparency because the resulting model is the unique solution to a convex optimization problem and has a closed-form expression in terms of the training data. Because modern machine learning algorithms lack such transparency, we propose two parsimonious *as if* representations of them. As with human decision-making, having a parsimonious representation that reasonably approximates machine learning behavior could enhance the theoretical and empirical analysis of modern machine learning, and more generally, would open the door to applying many tools of economics to better understand the latest approaches in machine learning.

Our analysis illustrates several advantages to applying models from information economics and bounded rationality to machine decision makers. First, the machine's loss function is a known and manipulable primitive of the decision problem, whereas a human's utility function must be inferred and is only indirectly manipulable. In direct contrast, Pattanayak and Krishnamurthy (2021) assume that each algorithm has an unobservable "utility" function that dictates the priorities it assigns to correct and incorrect predictions rather than treating the algorithm's objective as known and subject to external control as we do. A second advantage to applying models from the information economics and bounded rationality literatures to machine decision makers is that machines naturally generate state-dependent stochastic choice data (Caplin and Martin 2015), which is particularly well-suited for analyzing such models. Finally, machines may better approximate and emulate the rational paradigm. Human decisions contain strong and possibly immutable deviations from Bayesian updating and optimal choice, as documented by an extensive literature in behavioral economics. For example, a human may update beliefs in a biased manner, possibly for self-protective reasons.

While there are notable exceptions (e.g. Zhao, Ke, Wang, and Hsieh (2020) embed behavioral forces in a neural net structure, and Danan, Gajdos, and Tallon (2020) apply decision-theoretic approaches to recommendation systems), it is striking how little is known about algorithms as decision makers. By providing a parsimonious *as if* representation for opaque machine learners, we hope to open the door to applying the tools of economic analysis to better understand these important learners.

[16]See Varian (2014), Mullainathan and Spiess (2017), Athey (2018), and Athey and Imbens (2019).

References

Athey, Susan (2017). “Beyond prediction: Using big data for policy problems”. In: *Science* 355, pp. 483–485.

— (2018). “The impact of machine learning on economics”. In: *The economics of artifical intelligence: An agenda*. University of Chicago Press, pp. 507–547.

Athey, Susan and Guido W. Imbens (2019). “Machine learning methods that economists should know about”. In: *Annual Review of Economics* 11, pp. 685–725.

Bai, Yu, Song Mei, Huan Wang, and Caiming Xiong (2021). “Don’t just blame over-parameterization for over-confidence: theoretical analysis of calibration in binary classification”. In: *arXiv preprint arXiv:2102.07856*.

Belkin, Mikhail, Daniel Hsu, Siyuan Ma, and Soumik Mandal (2019). “Reconciling modern machine-learning practice and the classical bias-variance trade-off”. In: *Proceedings of the National Academy of Sciences* 116.32, pp. 15849–15854.

Belloni, Alexandre, Victor Chernozhukov, and Christian Hansen (2014). “High-dimensional methods and inference on structural and treatment effects”. In: *Journal of Economic Perspectives* 28.2, pp. 29–50.

Bergemann, Dirk and Stephen Morris (2016). “Bayes correlated equilibrium and the comparison of information structures in games”. In: *Theoretical Economics* 11.2, pp. 487–522.

— (2017). “Information design: A unified perspective”. In.

Blackwell, David (1953). “Equivalent comparisons of experiments”. In: *Annals of Mathematical Statistics*, pp. 265–272.

Caplin, Andrew and Mark Dean (2015). “Revealed preference, rational inattention, and costly information acquisition”. In: *American Economic Review* 105.7, pp. 2183–2203.

Caplin, Andrew, Mark Dean, and John Leahy (2017). *Rationally inattentive behavior: Characterizing and generalizing Shannon entropy.* Tech. rep. National Bureau of Economic Research.

Caplin, Andrew and Daniel Martin (2015). “A testable theory of imperfect perception”. In: *The Economic Journal* 125.582, pp. 184–202.

Caplin, Andrew, Daniel Martin, and Philip Marx (2022). “Revealed Bayesian Learning”. In.

Cattaneo, Matias D, Xinwei Ma, Yusufcan Masatlioglu, and Elchin Suleymanov (2020). “A random attention model”. In: *Journal of Political Economy* 128.7, pp. 2796–2836.

Danan, Eric, Thibault Gajdos, and Jean-Marc Tallon (2020). “Tailored recommendations”. In: *Social Choice and Welfare*, pp. 1–20.

De Oliveira, Henrique, Tommaso Denti, Maximilian Mihm, and Kemal Ozbek (2017). “Rationally inattentive preferences and hidden information costs”. In: *Theoretical Economics* 12.2, pp. 621–654.

DeGroot, Morris H and Stephen E Fienberg (1983). "The comparison and evaluation of forecasters". In: *Journal of the Royal Statistical Society: Series D (The Statistician)* 32.1-2, pp. 12–22.

Deng, Jia et al. (2009). "Imagenet: a large-scale hierarchial image database". In: *Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255.

Denti, Tommaso (2022). "Posterior separable cost of information". In: *American Economic Review* 112.10, pp. 3215–59.

Dietterich, Thomas G. (2000). "Ensemble methods in machine learning". In: *International workshop on multiple classifier systems*, pp. 1–15.

Domingos, Pedro (2000). "A unified bias-variance decomposition". In: *Proceedings of 17th International Conference on Machine Learning (ICML)*, pp. 231–238.

Dosovitskiy, Alexey et al. (2021). "An image is worth 16x16 words: Transformers for image recognition at scale". In: *International conference on learning representations*.

Fudenberg, Drew, Jon Kleinberg, Annie Liang, and Sendhil Mullainathan (2022). "Measuring the completeness of economic models". In: *Journal of Political Economy* 130.4, pp. 956–990.

Fudenberg, Drew and Annie Liang (2019). "Predicting and understanding initial play". In: *American Economic Review* 109.12, pp. 4112–4141.

Gentzkow, Matthew and Emir Kamenica (2014). "Costly persuasion". In: *American Economic Review* 104.5, pp. 457–62.

Gentzkow, Matthew, Bryan Kelly, and Matt Taddy (2019). "Text as data". In: *Journal of Economic Literature* 57.3.

Guo, Chuan, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger (2017). "On calibration of modern neural networks". In: *International Conference on Machine Learning*. PMLR, pp. 1321–1330.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Vol. 2. Springer: New York, NY, USA.

Hébert, Benjamin and Michael Woodford (2017). *Rational inattention and sequential information sampling*. Tech. rep. National Bureau of Economic Research.

Huang, Gao, Zhuang Liu, Kilian Q. Weinberger, and Laurens van der Maaten (2016). "Densely connected convolutional networks". In: *arXiv preprint arXiv:1608.06993*.

Ioffe, Sergey and Christian Szegedy (2015). "Batch normalization: accelerating deep network training by reducing internal covariate shift". In: *International Conference on Machine Learning (ICML)*, pp. 448–456.

Kamenica, Emir (2019). "Bayesian persuasion and information design". In: *Annual Review of Economics* 11, pp. 249–272.

Kamenica, Emir and Matthew Gentzkow (2011). "Bayesian persuasion". In: *American Economic Review* 101.6, pp. 2590–2615.

Kingma, Diederik and Jimmy Ba (2014). “Adam: a method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980.*

Kleinberg, Jon, Jens Ludwig, Sendhil Mullainathan, and Ziad Obermeyer (2015). “Prediction policy problems”. In: *American Economic Review: Papers and Proceedings* 105.5, pp. 491–495.

Lakshminarayanan, Balaji, Alexander Pritzel, and Charles Blundell (2017). “Simple and scalable predictive uncertainty estimation using deep ensembles”. In: *Advances in neural information processing systems 31.*

Liang, Annie, Jay Lu, and Xiaosheng Mu (2022). “Algorithmic design: Fairness versus accuracy”. In: *Proceedings of the 23rd ACM Conference on Economics and Computation*, pp. 58–59.

Lipnowski, Elliot and Doron Ravid (2022). “Predicting Choice from Information Costs”. In: *arXiv preprint arXiv:2205.10434.*

Lipton, Zachary C. (2018). “The mythos of model interpretability”. In: *ACM Queue* 16.3, pp. 31–57.

Liu, Sheng et al. (2022). “Deep probability estimation”. In: *arXiv preprint arXiv:2111.10734.*

Manzini, Paola and Marco Mariotti (2014). “Stochastic choice and consideration sets”. In: *Econometrica* 82.3, pp. 1153–1176.

Matejka, Filip and Alisdair McKay (2015). “Rational inattention to discrete choices: A new foundation for the multinomial logit model”. In: *American Economic Review* 105.1, pp. 272–98.

Minderer, Matthias et al. (2021). “Revisiting the calibration of modern neural networks”. In: *Neural Information Processing Systems (NeurIPS).*

Mullainathan, Sendhil and Jann Spiess (2017). In: *Journal of Economic Perspectives* 31.2, pp. 87–106.

Niculescu-Mizil, Alexandru and Rich Caruana (2005). “Predicting good probabilities with supervised learning”. In: *Proceedings of the 22nd international conference on Machine learning*, pp. 625–632.

Pattanayak, Kunal and Vikram Krishnamurthy (2021). “Behavioral Economics Approach to Interpretable Deep Image Classification. Rationally Inattentive Utility Maximization Explains Deep Image Classification”. In: *arXiv preprint arXiv:2102.04594.*

Rajpurkar, Pranav, Jeremy Irvin, Robyn L Ball, et al. (2018). “Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to practicing radiologists”. In: *PLoS medicine* 15.11, e1002686.

Rajpurkar, Pranav, Jeremy Irvin, Kaylie Zhu, et al. (2017). “CheXNet: radiologist-level pneumonia detection on chest x-rays with deep learning”. In: *arXiv preprint arXiv:1711.05225.*

Sims, Christopher A (2003). “Implications of Rational Inattention”. In: *Journal of Monetary Economics* 50.3, pp. 665–690.

Thai-Nghe, Nguyen, Zeno Gantner, and Lars Schmidt-Thieme (2010). "Cost-sensitive learning methods for imbalanced data". In: *The 2010 International joint conference on neural networks (IJCNN)*. IEEE, pp. 1–8.

Tolstikhin, Ilya O. et al. (2021). "MLP-mixer: An all-MLP architecture for vision". In: *Advances in neural information processing systems 34*.

Varian, Hal R. (2014). In: *Journal of Economic Perspectives* 28.2, pp. 3–28.

Wang, Xiaosong et al. (2017). In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2097–2106.

Woodford, Michael (2014). "Stochastic choice: An optimizing neuroeconomic model". In: *American Economic Review* 104.5, pp. 495–500.

Zadrozny, Bianca, John Langford, and Naoki Abe (2003). "Cost-sensitive learning by cost-proportionate example weighting". In: *Third IEEE international conference on data mining*. IEEE, pp. 435–442.

Zhao, Chen, Shaowei Ke, Zhaoran Wang, and Sung-Lin Hsieh (2020). "Behavioral Neural Networks". In: *Available at SSRN 3633548*.

A Application: Technical Details

Here we summarize the technical details of Section 2.2. Our model training procedure essentially follows that of the CheXNeXt algorithm (Rajpurkar, Irvin, Ball, et al. 2018), in which a deep neural network was trained using the ChestX-ray14 data set of Wang et al. (2017). The ChestX-ray14 data set consists of 112,120 frontal chest X-rays that were synthetically labeled with up to fourteen thoracic diseases. Our code for model training is adapted from the publicly available CheXNeXt codebase of Rajpurkar, Irvin, Ball, et al. (2018). However, we follow the earlier CheXNet implementation of Rajpurkar, Irvin, Zhu, et al. (2017) in three ways. First, we restrict to the binary classification task of pneumonia detection, where the labels of interest are pneumonia ($y = 1$) or not ($y = 0$). In addition, we trade off a higher batch rate of 16 at the expense of a slightly smaller imaging scaling size of 224 by 224 pixels (instead of a batch size of 8 and an image rescaling of 512 by 512 pixels, respectively). As in Rajpurkar, Irvin, Ball, et al. (2018), we adopt random horizontal flipping, and normalize based on the mean and standard deviation of images in the ImageNet data set (Deng et al. 2009). For each model, we train a 121-layer dense convolutional neural network (DenseNet, Huang, Liu, Weinberger, and Maaten 2016) with weights of the network initialized to those pretrained on ImageNet, using Adam with standard parameters 0.9 and 0.999 (Kingma and Ba 2014), and using batch normalization (Ioffe and Szegedy 2015). We use an initial learning rate of 0.0001 that is decayed by a factor of 10 each time the validation loss plateaus after an epoch, and we conduct early stopping based on validation loss. Each model was trained using either an Nvidia Tesla V100 16GB GPU or an Nvidia Tesla A100

40GB GPU on the Louisiana State University or Northwestern University high performance computing clusters, respectively.

Given the inferential nature of our exercise, we deviate substantively from this prior art in two ways. First, we induce variation in the cross-entropy loss function (1) across multiple positive class weights $\beta_1 = 0.7, 0.9, 0.99$, with 0.99 approximately equal to the inverse probability class weights for pneumonia detection adopted in Rajpurkar, Irvin, Zhu, et al. (2017). In addition to varying class weights, the main difference in our implementation and the implementation of Rajpurkar, Irvin, Zhu, et al. (2017) are our data splits and our recourse to additional ensemble methods to account for randomness in the training procedure. This use of ensemble methods also likely explains why our confidence scores are loss-calibrated, despite recent evidence that deep neural networks and cross-entropy loss may inherently produce poor calibration because of overconfidence (Bai, Mei, Wang, and Xiong 2021, Liu et al. 2022). Specifically, we adopt a nested cross-validation approach where we randomly split the data set into ten approximately equal folds and then iterate through 70-20-10 train-validation-test splits (the split distribution also used in Wang et al. 2017 and a secondary application of Rajpurkar, Irvin, Zhu, et al. 2017). We train a total of 480 models, yielding an ensemble of 96 trained models for each observation in the data set where that observation was in a test fold. The final score for each observation in the data set is then obtained by averaging confidence scores across the observation's ensemble. This procedure is repeated on the same set of data splits for each weight $\beta = 0.7, 0.9, 0.99$ we consider.

B The Loss-Difference Matrix

B.1 Derivation Example

To illustrate, consider the following loss functions for binary predictions and outcomes:

$$L^1 = \begin{array}{cc} y=0 & y=1 \\ \left(\begin{array}{cc} 0 & .5 \\ .5 & 0 \end{array}\right) & \end{array} \begin{array}{c} \\ a=0 \\ a=1 \end{array} \qquad L^2 = \begin{array}{cc} y=0 & y=1 \\ \left(\begin{array}{cc} 0 & .6 \\ .4 & 0 \end{array}\right) & \end{array} \begin{array}{c} \\ a=0 \\ a=1 \end{array}$$

and suppose an algorithm produces the following performance data:

$$P^1 = \begin{array}{cc} y=0 & y=1 \\ \left(\begin{array}{cc} .25 & .1 \\ .25 & .4 \end{array}\right) & \end{array} \begin{array}{c} \\ a=0 \\ a=1 \end{array} \qquad P^2 = \begin{array}{cc} y=0 & y=1 \\ \left(\begin{array}{cc} .3 & .2 \\ .2 & .3 \end{array}\right) & \end{array} \begin{array}{c} \\ a=0 \\ a=1 \end{array}$$

```
for k=1:M
    for i=1:M
        for j=1:M
            if W(i,j)>W(i,k)+W(k,j)
                W(i,j)=W(i,k)+W(k,j);
            end
        end
    end
end
```

Figure 3: Using the Floyd–Warshall algorithm to generate the D matrix.

The cross-loss matrix can be computed as:

$$D_0 = \begin{array}{cc} P^1 & P^2 \\ \end{array} \quad \begin{array}{l} \\ \end{array}$$

$$D_0 = \left(\begin{array}{cc|c} P^1 & P^2 & \\ .175 & .2 & L^1 \\ .16 & .2 & L^2 \end{array}\right)$$

Note that D_0 contains two rows and columns with binary variation in the loss function $M = 2$. Based on this cross-loss matrix, the loss-difference matrix D is

$$D = \left(\begin{array}{cc|c} P^1 & P^2 & \\ -.015 & .025 & L^1 \\ -.04 & -.015 & L^2 \end{array}\right)$$

The off-diagonal elements D^{12} and D^{21} are both defined by the direct change in losses caused by the corresponding switches of loss function. The diagonal elements D^{11} and D^{22} reflect indirect paths. Starting from L^2 the indirect path back to L^2 involves first lowering losses by $0.2 - 0.16 = 0.04$ switching P^2 to P^1, then raising them by $0.2 - 0.175 = 0.025$ switching P^1 to P^2. Therefore $D^{22} = -0.015 < 0$; analogous logic explains why $D^{11} = -0.015 < 0$.

B.2 Computing the D Matrix with Floyd-Warshall

The Floyd-Warshall algorithm, which identifies the minimal cost of weighted paths between nodes in a weighted directed graph, can be used to generate the D matrix when the algorithm is loss-adapted. This algorithm, presented in Figure 3, is trivial to apply. It takes as an input a directed graph with weight $W(i, j)$ on the vertex from node i to node j and cycles through these weights.

Perhaps the most important point is that the well-known Floyd-Warshall algorithm is polynomial: it has a complexity of $O(V^3)$. To determine whether an algorithm is not loss-adapted, it is sufficient to add a simple check at each step in the Floyd-Warshall algorithm. If $i = j$ and $W(i, j) < 0$, then the algorithm has identified a negative cycle, meaning that the algorithm is not loss-adapted. Note that if the algorithm is not loss-adapted, the candidate matrix computed by Floyd-Warshall need not coincide with the loss-difference matrix D. For example, it does not coincide with the true D matrix computed in the previous derivation (Appendix B.1); nevertheless, it generally recovers whether the algorithm is loss-adapted.

C An Illustrative Example of Cost Recovery

This appendix provides an alternative example and illustration of cost recovery. Consider the following hypothetical D matrix in the case of three loss functions, $M = 3$.

$$D = \begin{array}{c} \\ \\ \\ \end{array}\begin{array}{cccc} P^1 & P^2 & P^3 & \\ \left(\begin{array}{c} 0 \\ 3 \\ 1 \end{array}\right. \begin{array}{c} -2 \\ 0 \\ -2 \end{array} & & \left.\begin{array}{c} 1 \\ 3 \\ 0 \end{array}\right) & \begin{array}{c} L^1 \\ L^2 \\ L^3 \end{array} \end{array} \tag{7}$$

The algorithm is consistent with cost-based machine learning because the diagonal elements of D are zero. This raises the question of what we can recover about the algorithm's costs of learning.

(-1,2) and (1,3) are extreme points of the cost polyhedron normalized to $K^3 = 0$. Qualifying costs satisfy $K^m - K^n \leq D^{mn}$, so

$$\begin{aligned} K^1 - K^2 &\leq -2; \\ K^2 - K^1 &\leq 3. \end{aligned}$$

Combining yields $K^1 - K^2 \in [-2, -3]$. By analogous reasoning and the normalization $K^3 = 0$, we have $K^1 \in [-1, 1]$ and $K^2 \in [2, 3]$. The resulting cost polyhedron is illustrated in Figure 4. To illustrate the representative cost, consider again the previous example (7). The average of the last row is $-\frac{1}{3}$, and the average of the last column is $\frac{4}{3}$, so

$$\frac{\bar{D}^{M*} - \bar{D}^{*M}}{2} = -\frac{5}{6}$$

Thus,

$$\bar{K} = (\bar{K}^1, \bar{K}^2, \bar{K}^3) = (\frac{\bar{D}^{1*} - \bar{D}^{*1}}{2} + \frac{5}{6}, \frac{\bar{D}^{2*} - \bar{D}^{*2}}{2} + \frac{5}{6}, -\frac{5}{6} + \frac{5}{6}) = (0, 2.5, 0).$$

This representative cost is illustrated by the black square in Figure 4.

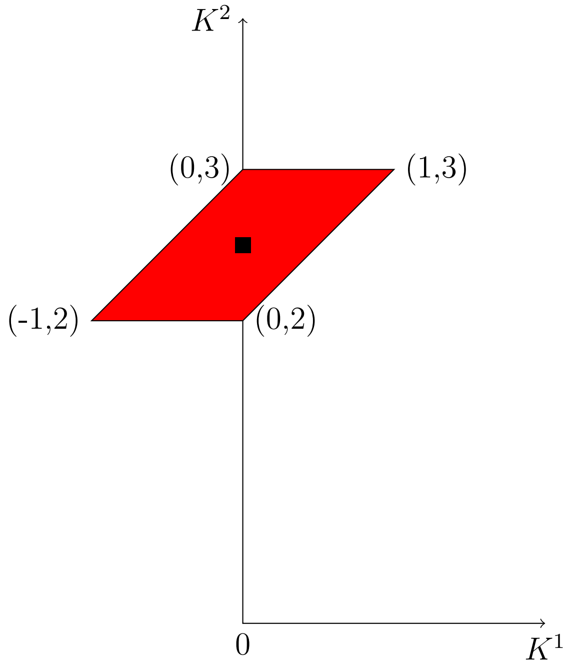

Figure 4: Example 2-dimensional learning cost polyhedron with representative cost function.