BOSTON UNIVERSITY

COLLEGE OF ENGINEERING

Dissertation

LEARNING TO PREDICT UNDER A BUDGET

by

FENG NAN

B.S., National University of Singapore, 2008
M.S., Massachusetts Institute of Technology, 2009

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2018

Approved by

First Reader

Venkatesh Saligrama, PhD
Professor of Electrical and Computer Engineering
Professor of Systems Engineering
Professor of Computer Science

Second Reader

David A. Castañón, PhD
Professor of Electrical and Computer Engineering
Professor of Systems Engineering

Third Reader

Lorenzo Orecchia, PhD
Assistant Professor of Computer Science

Fourth Reader

Alexander Olshevsky, PhD
Assistant Professor of Electrical and Computer Engineering
Assistant Professor of Systems Engineering

Acknowledgments

I have been fortunate to receive much support for the work of this thesis.

First, I am happy to give thanks to God and the Lord Jesus Christ for His love. He opened the door for me to pursue doctorate study and has carried me through.

I would also like to thank my advisor Professor Venkatesh Saligrama for his guidance and encouragement. The breath and depth of his intellect will continue to be my aspiration. In addition, I have learned what doing good research is about from seeing him at work. It is about being optimistic and enjoying the process even when the goal seems far; it is also about never giving up an idea without knowing why it does not work.

I also had the privilege of working with Professor Yannis Paschalidis, who kindly mentored me for the first year at BU and introduced me to topics in computational biology. I would also like to acknowledge Professor David Castañón and Professor Lorenzo Orecchia for their availability to share their insights with me on optimization. I also thank Professor Alexander Olshevsky for agreeing to be on my thesis committee.

I must also thank my colleagues and friends in BU for their numerous help and encouragement. They made my doctorate study a lot more enjoyable.

Last but not least, I would like to thank my family, especially my wife Yanping. She gave up much to support me through my study and gave birth to our two beautiful girls. Her sacrifice and support are indispensable to my success. I also thank our parents for their selfless love and unconditioned support.

LEARNING TO PREDICT UNDER A BUDGET

FENG NAN

Boston University, College of Engineering, 2018

Major Professor: Venkatesh Saligrama, PhD
Professor of Electrical and Computer Engineering
Professor of Systems Engineering
Professor of Computer Science

ABSTRACT

Prediction-time budgets in machine learning applications can arise due to monetary or computational costs associated with acquiring information; they also arise due to latency and power consumption costs in evaluating increasingly more complex models. The goal in such budgeted prediction problems is to learn decision systems that maintain high prediction accuracy while meeting average cost constraints during prediction-time.

In this thesis, I will present several learning methods to better trade-off cost and error during prediction. The conceptual contribution of this thesis is to develop a new paradigm of bottom-up approaches instead of the traditional top-down approaches. A top-down approach attempts to build out the model by selectively adding the most cost-effective features to improve accuracy. It leads to fundamental combinatorial issues in multi-stage search over all feature subsets. In contrast, a bottom-up approach first learns a highly accurate model and then prunes or adaptively approximates it to trade-off cost and error. We show that the bottom-up approach has several benefits.

To develop this theme, we first propose two top-down methods and then two bottom-up methods. The first top-down method uses margin information from training data in the partial feature neighborhood of a test point to either select the next best feature in a greedy fashion or to stop and make prediction. The second top-down method is a variant of random forest (RF) algorithm. We grow decision trees with low acquisition cost and high strength based on greedy minimax cost-weighted impurity splits. Theoretically, we establish near-optimal acquisition cost guarantees for our algorithm.

The first bottom-up method we propose is based on pruning RFs to optimize expected feature cost and accuracy. Given a RF as input, we pose pruning as a novel 0-1 integer program and show that it can be solved exactly via LP relaxation. We further develop a fast primal-dual algorithm that scales to large datasets. The second bottom-up method is adaptive approximation, which significantly generalizes the RF pruning to accommodate more models and other types of costs besides feature acquisition cost. We first train a high-accuracy, high-cost model. We then jointly learn a low-cost gating function together with a low-cost prediction model to adaptively approximate the high-cost model. The gating function identifies the regions of the input space where the low-cost model suffices for making highly accurate predictions.

We demonstrate empirical performance of these methods and compare them to the state-of-the-arts. Finally, we study adaptive approximation in the on-line setting to obtain regret guarantees and discuss future work.

Contents

List of Tables

List of Figures

List of Abbreviations

DAG	Directed Acyclic Graph
GBRT	Gradient Boosted Regression Tree
RF	Random Forest
$\Re^K$	the K-dimensional Euclidean space

Chapter 1

Introduction

1.1 Resource-constrained Machine Learning: Motivation

Machine learning plays an increasingly important role in many scientific and engineering problems. It includes problems such as classification, regression, ranking, clustering and so on. Much of machine learning research has focused on improving accuracy. But more recently, as the scale and complexity of machine learning applications grow, costs in both training and test time have gained importance. To limit scope, we consider exclusively supervised learning in this thesis. Training time thus typically involves the cost of collecting labeled data and the computational cost of processing the collected data to learn the model. In many applications such as health care, labeled data is scarce and expensive. The area of active learning (Settles, 2009) is devoted to efficiently using fewer labeled examples to train models. Once a model is trained, it is used for prediction of new examples. Prediction-time costs can arise due to monetary costs associated with acquiring information or computation time (or delay) involved in extracting features and running the algorithm; they can also arise in mobile computing due to limited memory, battery and communication.

In many machine learning applications training can be carried out off-line, separate from the production system. On the other hand, prediction typically occurs in production and is subject to more stringent budget constraints. Therefore, this thesis focuses primarily on reducing costs incurred during prediction or test time. Only toward the end (Chapter 6) we will discuss an on-line learning scenario where we bring together training and test time costs. Consider the following applications as motivation for prediction time budget constraints.

- **Automated medical diagnosis:** This is a classification task. During training, an algorithm is given medical records of diagnosed patients as input features and the diagnosis as labels. The goal is to learn a model to automatically diagnose new patients based on the outcome of their medical test results. Some of these medical tests are simple and inexpensive such as blood pressures, vitals. Others are more expensive and could potentially be harmful to the human body such as X-ray, MRI. The prediction time cost consists of the monetary cost of each medical test as well as its associated risk. When a new patient is presented to the system, it is thus undesirable to require him or her to undertake all possible medical tests and then make a prediction. Instead, we aim to learn a system that recommends only the necessary medical tests to reduce cost while maintaining high diagnosis accuracy.

- **Document ranking:** (Chapelle et al., 2011) This is a ranking task. During training, an algorithm is given a set of queries as well as a set of documents associated with each query ranked according to the relevance to the query. The goal is to learn a model so that given a new query and a set of documents, it can rank the documents according to the relevance to the query. To achieve this, features of each query-document pair must be extracted. Some features are cheap to extract, such as key word search; other features are computationally more expensive, such as textual similarity and proximity measures. Each of these features require CPU time to extract, yet the ranking has to be done in milliseconds to be displayed to the user. This precludes extraction of computationally expensive features for all query-document pairs. We aim to learn a system that extracts the expensive features only if it is necessary to reduce cost while maintaining high ranking accuracy.

- **Deep neural networks (DNNs):** DNNs have been successfully applied in many application including visual object recognition, speech recognition and machine translation. They achieve the state of the art accuracy yet require considerable computational budget during prediction due to their increasing complexity. For example, the Resnet152 (He et al., 2016) architecture with 152 layers has 4.4% accuracy gain in top-5 performance over GoogLeNet (Szegedy et al.,) on the large-scale ImageNet dataset (Russakovsky et al., 2015) but is 14X slower at test-time(Bolukbasi et al., 2017). We aim to learn systems that can reduce the computational cost while maintaining high accuracy.

- **Mobile computing, Internet of Things (IoT):** Smart devices include phones, watches, cameras and sensors (known as edge devices) have been widely used to gather and process information for tasks such as activity recognition and surveillance. Such devices typically have limited battery, memory and computational power. Machine learning models that run on such devices are constrained by these physical limitations. For real-time applications, there is also a communication cost in terms of latency whenever the edge devices communicate with the server(cloud). We aim to develop machine learning systems that are suitable to be deployed on such edge devices and use small budget to achieve high accuracy.

1.2 Problem Definition

In this section, we introduce some basic notations and present the general problem of learning with prediction-time costs similar to the formulation in (Trapeznikov and Saligrama, 2013; Wang et al., 2014b). We focus on the supervised setting where we assume fully annotated datasets are available for training. We seek to learn decision systems that maintain high-accuracy while meeting average resource constraints during prediction-time.

Suppose an example-label pair (x, y) is drawn from distribution $\mathcal{P}$. The goal is to learn a prediction function f from a family of functions $\mathcal{F}$ that minimizes expected prediction error subject to a budget constraint:

$$\min_{f \in \mathcal{F}} \mathbb{E}_{(x,y)\sim\mathcal{P}} \left[err\left(y, f(x)\right)\right], \text{ s.t. } \mathbb{E}_{x\sim\mathcal{P}_x}\left[C\left(f, x\right)\right] \leq B, \tag{1.1}$$

where $err\left(y, \hat{y}\right)$ is the error function; $C(f, x)$ is the cost of evaluating the function f on example x and B is a user specified budget constraint.

In practice, we are not given the distribution but instead are given a set of training data $(x^{(1)}, y^{(1)}), \dots, (x^{(n)}, y^{(n)})$ drawn i.i.d. from distribution $\mathcal{P}$. We can then minimize an empirical approximation of the expected error function:

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L\left(y^{(i)}, f(x^{(i)})\right), \text{ s.t. } \frac{1}{n}\sum_{i=1}^{n} C\left(f, x^{(i)}\right) \leq B, \tag{1.2}$$

where $L\left(y, \hat{y}\right)$ is a loss function. Note our budget constraint is on prediction costs averaged over the examples. This allows the flexibility to spend the budget in an example-dependent manner.

The definition of $C(f, x)$ is application specific as seen in the motivation examples in Section 1.1. We shall focus on the *feature acquisition cost* in this thesis while addressing other types of costs such as *computational* and *communication/latency* costs as well.

1.2.1 Feature Acquisition Cost

Features (or covariates in statistics) are the numerical attributes associated with an input example. They provide information about the examples as a basis for prediction. There is often a cost associated with acquiring or extracting these feature values. Suppose $x \in \Re^K$ is the feature vector with an acquisition cost $c_\alpha \geq 0$ assigned to each of the features $\alpha = 1, \ldots, K$. [1]

For a given example x, we assume that once it pays the cost to acquire a feature, its value can be efficiently cached; and subsequent use of the feature value does not incur additional cost. Thus, the cost of utilizing a particular prediction function, denoted by $C(f, x)$, is computed as the sum of the acquisition cost of *unique* features required by f for x.

1.2.2 Computational Cost

$C(f, x)$ can also measure the amount of computation required to compute $f(x)$. In a decision tree f, for example, it is proportional to the number of internal nodes x traverses. In a neural network, it is proportional to the number of layers and the number of connections between the layers.

1.2.3 Communication/Latency Cost

In mobile applications, prediction $f(x)$ may involve communication between the edge device and the server (cloud). $C(f, x)$ can capture such costs in terms of communication/latency cost.

1.3 Challenges

The problem of learning to prediction under a budget might appear well-studied as formulated in Eq.(1.2), which consists of an empirical loss minimization subject to a constraint. Indeed, the sparse learning or feature selection problem is an instance of learning to predict under a budget. Consider each feature element carries a unit acquisition cost and $\mathcal{F}$ is the space of linear regressors. Each $f \in \mathcal{F}$ can be parame-

[1]Note that our algorithms can be adapted to handle group-structured features where several elements in x may be associated with one feature acquisition cost. In other words, several elements in the x vector can be obtained together by paying the acquisition cost for one feature. We avoid it in the exposition for clarity purpose.

terized by $w \in \Re^K$. The cost $C(f, x)$ is equal to the number of non-zero elements in w: $C(f, x) = \|w\|_0$. The budget constraint on the prediction-time feature acquisition cost thus reduces to a sparsity constraint on w. The sparse linear regression problem is

$$\min_{w \in \Re^K} \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - w^T x^{(i)}\right)^2, \text{ s.t. } \|w\|_0 \leq B. \tag{1.3}$$

Algorithms including LASSO (Tibshirani, 1996) and other subset selection methods have been well established (Miller, 2002). Yet we highlight that the goal of traditional sparse learning or feature selection is to identify a subset of the features to be used for all the examples. The assumption is that there exists a common subset of features that are useful for predicting *all* examples. But in practice, different examples may benefit from different subsets of features. Consider the medical diagnosis example, it makes sense to recommend different subsets of medical tests for different patients, depending on their individual conditions. In other words, the decision functions that we seek to learn are more general, able to *adapt* to different input examples.

The key idea in our budgeted prediction framework is to recognize that in many machine learning tasks not all input examples are created equal. There are "easy" examples that can be predicted at low cost (e.g. using a few low cost features or going through a small number of layers in a neural network). Only the "difficult" examples require more cost (e.g. using more features or going through many layers in a neural network). Since the budget constraint is on the average prediction cost over the examples, we can achieve high prediction accuracy by allocating less budget on the "easy" examples and more budget on the "difficult" ones.

In some sense, the family of decision functions $\mathcal{F}$ in Eq. (1.2) that we optimize over is a family of adaptive decision rules, or decision policies, rather than static models such as a linear predictor. We highlight several challenges this entails.

- **Distinguish "easy" V.s. "difficult" examples:** Given a training dataset, it is not clear how to partition the examples into "easy" V.s. "difficult" ones. The partition function itself is a classifier that needs to be learned. Furthermore, how the dataset is partitioned impacts the data distribution for the downstream prediction models. In other words, the partition function should be learned jointly with a "cheap" prediction model that handles the "easy" examples as well as an "expensive" prediction model that handles the "difficult" ones. This interdependency translates to products of indicator functions in the optimization objective and leads to non-convexity.

- **Combinatorial state space:** With feature acquisition costs, the adaptive decision rule can be represented by a Directed Acyclic Graph (DAG) (Wang et al., 2015). The internal nodes correspond to feature subsets and decision functions at each node choose whether to acquire a new feature or predict using the already acquired features. The edges correspond to acquiring new features, transitioning from one feature subset to another. The number of states, or feature subsets, is 2^K, where K is the number of features. Learning decision functions for each state becomes intractable when the number of features is large.

1.4 Contribution

We develop several novel algorithmic approaches to the learning under prediction budget problem, improving the state of the art performance with each method. More importantly, through these methods, we develop a new *bottom-up* paradigm for the learning under prediction budget problem. Here we give a summary of the contributions made in this thesis.

- We propose a nearest neighbor approach to feature selection that incorporates margins in classification.

- We propose to learn the adaptive decision rule as random forests. We propose a family of impurity measures and a splitting criteria so that the decision trees we grow are guaranteed to have near-optimal feature acquisition costs.

- Given any random forest, we propose to prune it to optimize expected feature cost & accuracy. We pose pruning RFs as a novel 0-1 integer program and establish total unimodularity of the constraint set to prove that the corresponding LP relaxation solves the original integer program. We further exploit connections to combinatorial optimization and develop an efficient primal-dual algorithm that scales to large problems. This *bottom-up* pruning approach circumvents the need for combinatorial search faced by the *top-down* approaches.

- We develop an adaptive approximation framework as a general *bottom-up* approach. The framework incorporates general machine learning models such as RF, boosting, SVM and neural networks. It also accounts for various types of costs such as feature acquisition, computational and communication/latency costs.

- We propose an on-line learning framework for the adaptive approximation and provide regret analysis.

1.5 Related Work

The problem of learning from full training data for prediction-time cost reduction (MacKay, 1992) has been extensively studied. We summarize related work according to the key properties in their approaches. We focus on the feature acquisition costs first.