The Pragmatic Programmers

Your Elixir Source

Machine Learning in Elixir

Learning to Learn with Nx and Axon

Sean Moriarity

Edited by Tammy Coron

This extract shows the online version of this title, and may contain features (such as hyperlinks and colors) that are not available in the print version.

For more information, or to purchase a paperback or ebook copy, please visit https://www.pragprog.com.

Preface

Machine learning, specifically deep learning, is constantly pushing the boundaries of what we thought was possible—and it's doing it in every industry.

Seemingly every day, a company or research group releases a new model that pushes the state of the art even further ahead. In recent years, ChatGPT and Stable Diffusion, among others, have taken the world by storm, bringing artificial intelligence to the forefront and redefining what types of applications are possible.

For most of their existence, Elixir and the BEAM weren't viable options for machine learning tooling. But the Nx ecosystem has changed that. Within this ecosystem, you can now write machine learning and numerical routines directly in Elixir and achieve performance that is the same as or better than an equivalent program in Python or Julia. Nx offers new capabilities to Elixir programmers. It also provides an off-ramp for existing machine learning engineers and researchers looking to explore ecosystems and build applications without Python.

My goal is for this book to serve as the definitive Nx and machine learning resource for programmers who want to explore machine learning in Elixir. While a lot of machine learning knowledge is available online, none of the existing resources are written with Elixir in mind. This book is designed to teach machine learning the Elixir way—functional, pragmatic, and fun.

You'll start with the basics of machine learning and Nx, and you'll build up to deploying powerful pre-trained models in an application with Phoenix. Throughout this book, you'll use Livebook—Elixir's interactive code notebook—to work with data and train models in an interactive and reproducible way. By the end of the book, you'll have experience working with Nx, Axon, Scholar, Bumblebee, Explorer, VegaLite, and many more libraries in the growing Elixir machine learning ecosystem.

We've finally made it to the always-promised but never-delivered golden age of artificial intelligence-enabled products. So, if you want to build powerful,

intelligent applications augmented with machine learning models, Elixir is one of the best languages to use.

Why Elixir for Machine Learning?

Elixir is a dynamic, functional, general-purpose programming language that targets the BEAM virtual machine. It's proven itself as a capable language for building and maintaining scalable applications. But Elixir wasn't designed with machine learning in mind. Prior to the Nx[1] project, machine learning in Elixir was, at the very least, ill-advised. The BEAM virtual machine wasn't designed to perform the type of heavy lifting that machine learning requires. However, Nx and the ecosystem around it were built to enable Elixir programmers to perform the computationally expensive work required for machine learning applications.

Elixir and Nx are new players in the machine learning field. While the Nx ecosystem has no aim to overtake the established Python machine learning ecosystem, understanding machine learning in Elixir has numerous benefits.

First, Elixir is a *functional* programming language. While this might seem like a drawback for the heavy computation required in machine learning workloads, it's actually a benefit. In a functional world, everything is immutable. This means that when performing computations on large amounts of data, you need to return a new copy of the data after every individual operation. Fortunately, due to the rise of deep learning, computations are staged out to hardware accelerators such as GPUs before any of the actual operations are executed. These computations are staged out with just-in-time (JIT) compilers, which typically implement a functional interface for building up computation graphs. As you'll see in later chapters, you can write in Elixir's functional style and not worry about large intermediate copies.

Second, Elixir is designed for concurrency. When training machine learning algorithms, the performance bottleneck is typically in loading and preparing data for a training step on a hardware accelerator. This type of concurrent data processing is natural in Elixir because Elixir was designed with concurrency in mind. You need only to read *Concurrent Data Processing in Elixir [Gos21]* to see how good a fit Elixir is for those types of applications.

Third, Elixir and its web framework, Phoenix, are exceptional at building and scaling real-time, fault-tolerant applications. More and more machine learning systems are being deployed in scenarios where latency and uptime matter.

1. https://github.com/elixir-nx/nx

For example, a credit card fraud detection system should ideally work in real time, immediately alerting customers when a fraudulent transaction occurs. Additionally, with large amounts of fraudulent transactions taking place every day, those systems can't go down. Elixir and Phoenix have proven capable of building systems that are low-latency and resilient to failure. But until now, integrating machine learning in an Elixir application required leaving the safety of the BEAM to an external service.

Finally, Elixir is a lot of fun and an absolute joy to use. As you'll see throughout this book, most machine learning algorithms are naturally expressed with functional constructs that work particularly well with the beauty and thoughtfulness of the Elixir programming language.

Who This Book Is For

This book is for programmers interested in learning machine learning in Elixir and programmers with prior Elixir or machine learning experience.

Elixir programmers can pick up and read this book with no prerequisite knowledge of machine learning, mathematics, or numerical computing, as this book will teach you everything you need to know from the basics onward.

Programmers with machine learning experience but no Elixir experience will be able to follow along, but some syntactic details and functional concepts might seem unfamiliar. If you fit into this category, I recommend keeping the Elixir documentation handy to reference some language details that this book omits.

If you have no machine learning or Elixir experience, then learning both Elixir and machine learning at the same time will be a difficult task. So I recommend getting familiar with Elixir first and then returning to this book to learn machine learning in Elixir.

What's in This Book

In Chapter 1, Make Machines That Learn, on page ?, you'll answer the question, "What is machine learning?" You'll learn the fundamental concepts of machine learning and set the course for the rest of your machine learning journey. You'll create, train, and use your first machine learning model. By the end of the chapter, you'll be able to define machine learning, discuss the types of problems machine learning is good for, and solve a problem with machine learning in Elixir.

In Chapter 2, Get Comfortable with Nx, on page ?, you'll get familiar with Nx—the numerical computing library that serves as the backbone of the Elixir machine learning ecosystem. You'll learn about the fundamental data structure in numerical computing, the tensor, and a bit about what makes Nx so important for machine learning. By the end of this chapter, you'll know how to create and manipulate tensors, encode real-world data with Nx, and accelerate routines with numerical definitions.

In Chapter 3, Harness the Power of Math, on page ?, you'll channel your inner mathematician and dive into machine learning math. You'll learn the three areas of mathematics that underpin most of the concepts in this book: linear algebra, probability, and vector calculus. By the end of this chapter, you'll understand the power of mathematics and how it relates to machine learning.

In Chapter 4, Optimize Everything, on page ?, you'll start navigating an ocean of loss curves with optimization routines. You'll learn what a loss function is and how optimization and machine learning are related and different. By the end of this chapter, you'll be able to formulate machine learning problems as optimization problems. You'll also be able to implement some loss functions and optimization algorithms in Nx.

In Chapter 5, Traditional Machine Learning, on page ?, you'll graduate from machine learning's foundations to real machine learning with Scholar. You'll learn about various traditional machine learning algorithms, such as linear and logistic regression. By the end of this chapter, you'll be able to use Scholar to create and train traditional machine learning algorithms.

In Chapter 6, Go Deep with Axon, on page ?, you'll start working with Axon to create and train neural networks in Elixir. You'll learn what deep learning is and why it's so powerful. You'll create neural networks in both Nx and Axon, and you'll get a better understanding of what a neural network really is. By the end of this chapter, you'll know how to create and train basic neural networks with Axon. You'll also be able to distinguish and discuss the trade-offs between traditional and deep learning methods.

In Chapter 7, Learn to See, on page ?, you'll implement computer vision models with Axon and Elixir. You'll learn about convolutional neural networks and how they offer an improvement over traditional neural networks when working with image data. By the end of this chapter, you'll be able to create and train a convolutional neural network in Axon. You'll also be able to implement more complex machine learning training pipelines with Elixir's standard libraries.

In Chapter 8, Stop Reinventing the Wheel, on page ?, you'll make use of pre-trained models to improve performance and save time and money. You'll learn about transfer learning and fine-tuning. By the end of this chapter, you'll be able to implement transfer learning in Axon. You'll also know how to convert models from Python to Elixir to take advantage of Python's vast machine learning ecosystem.

In Chapter 9, Understand Text, on page ?, you'll teach your programs to read with recurrent neural networks, and you'll learn how to work with text in Elixir. By the end of this chapter, you'll be able to create and train recurrent neural networks in Axon, define and use various text preprocessing strategies with Elixir and Nx, and discuss why recurrent neural networks perform better than traditional models when working with sequential data-like text.

In Chapter 10, Forecast the Future, on page ?, you'll break out your crystal ball and attempt to peer into the future with recurrent neural networks. You'll learn about time-series forecasting and some of the challenges of predicting the future with machine learning. By the end of this chapter, you'll be able to create and train a recurrent neural network to perform single-step time-series predictions.

In Chapter 11, Model Everything with Transformers, on page ?, you'll unlock the power behind models like GPT-3 and Stable Diffusion with transformers. You'll learn what transformers are and why they perform so well on almost all types of data. By the end of this chapter, you'll be able to use pre-trained transformer models with Bumblebee. You'll also know how to fine-tune a pre-trained transformer with Bumblebee and Axon.

In Chapter 12, Learn Without Supervision, on page ?, you'll go off the rails and create models that learn without supervision. You'll learn about unsupervised learning and generative modeling. By the end of this chapter, you'll be able to create and train autoencoders, variational autoencoders, and generative adversarial networks.

In Chapter 13, Put Machine Learning into Practice, on page ?, you'll use everything you learned throughout this book, and you'll put a real machine learning model into a production application. You'll learn what to consider when operationalizing machine learning models, and you'll deploy a model in a Phoenix application. By the end of this chapter, you'll know how to use `Nx.Serving` to integrate your trained models into real applications.

In Chapter 14, That's a Wrap, on page ?, you'll conclude your journey with this book. You'll get a recipe for training models in practice, and you'll get resources for staying up to date with the latest trends in machine learning.

You'll also learn a bit about emerging trends in the field, as well as some topics this book didn't cover, such as adversarial attacks. By the end of this chapter, you'll be ready to build intelligent applications with machine learning in Elixir.

How to Use This Book

The code examples in each chapter of this book are isolated and designed to be run as standalone programs. For the best experience, read the chapters in successive order. While you can skip around, the concepts in each chapter are designed to build off one another, so you might be left confused at certain points if you skip around too much.

If you have some machine learning experience and want to dive right in, you can safely skip Chapter 1, Make Machines That Learn, on page ?, Chapter 3, Harness the Power of Math, on page ?, and Chapter 4, Optimize Everything, on page ?. I still recommend reading Chapter 2, Get Comfortable with Nx, on page ?, to get a better understanding of Nx and then reading the remaining chapters in order.

I also recommend you follow along with the examples in a Livebook as most of the chapters will assume you're running the code in a Livebook installation.

Finally, don't worry about having access to CPUs or GPUs. A modern, commercial laptop should be powerful enough to run all the examples in this book.