How Open Source Machine Learning Software Shapes AI

Max Langenkamp*
Massachusetts Institute of Technology
Cambridge, MA, USA
maxnz@mit.edu

Daniel N. Yue*
Harvard Business School
Cambridge, MA, USA
dyue@hbs.edu

ABSTRACT

If we want a future where AI serves a plurality of interests, then we should pay attention to the factors that drive its success. While others have studied the importance of data, hardware, and models in directing the trajectory of AI, we argue that open source software is a neglected factor shaping AI as a discipline. We start with the observation that almost all AI research and applications are built on machine learning open source software (MLOSS). This paper presents three contributions. First, it quantifies the outsized impact of MLOSS by using Github contributions data. By contrasting the costs of MLOSS and its economic benefits, we find that the average dollar of MLOSS investment corresponds to at least $100 of global economic value created, corresponding to $30B of economic value created this year. Second, we leverage interviews with AI researchers and developers to develop a causal model of the effect of open sourcing on economic value. We argue that open sourcing creates value through three primary mechanisms: standardization of MLOSS tools, increased experimentation in AI research, and creation of communities. Finally, we consider the incentives for developing MLOSS and the broader implications of these effects. We intend this paper to be useful for technologists and academics who want to analyze and critique AI, and policymakers who want to better understand and regulate AI systems.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Machine learning**; • **Social and professional topics** → **Computing / technology policy**.

KEYWORDS

open source, machine learning

ACM Reference Format:
Max Langenkamp and Daniel N. Yue. 2022. How Open Source Machine Learning Software Shapes AI. In *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society (AIES'22), August 1–3, 2022, Oxford, United Kingdom.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3514094.3534167

*Both authors contributed equally to this research.

AIES'22, August 1–3, 2022, Oxford, United Kingdom

ACM ISBN 978-1-4503-9247-1/22/08…$15.00
https://doi.org/10.1145/3514094.3534167

1 INTRODUCTION

Interest in artificial intelligence (AI) has exploded over the past decade. Now, even casual consumers interact daily with AI systems. This is often attributed to advances in data, hardware, and algorithms [9]. These factors are sometimes described as inputs to the so-called 'AI production function'. In this paper, we consider a neglected factor: open source machine learning software (MLOSS). MLOSS is ubiquitous in both research and production. However, it has received comparatively little attention in the literature. In this paper, we argue that MLOSS is a powerful point of intervention for shaping AI research and a phenomenon that merits further examination.

Our argument contains three parts:

(1) MLOSS tools play an outsized role in the creation of economic value
(2) MLOSS drives AI impact through standardization, experimentation, and community creation
(3) MLOSS reinforces the deep learning paradigm

Overall, our study deepens understanding of how MLOSS impacts the AI ecosystem. We offer three contributions, corresponding to the three parts of our argument. First, we estimate the economic impact of MLOSS tools, which, to our knowledge, is the first estimate of its kind. We argue that the large cost-benefit ratio suggests MLOSS is a useful point of intervention for policymakers. Second, using qualitative interview data, we propose that MLOSS shapes AI development through standardization, experimentation, and community creation. Finally, we argue that the same factors that generate economic value reinforce the dominant paradigm of deep learning. This leads to greater value capture by large institutions and a narrower set of AI capabilities.

The paper will proceed as follows: Section 2 sets up our argument by providing definitions and a brief history of MLOSS. Section 3 contains our economic estimate of MLOSS. Section 4 contains our causal model of how MLOSS shapes the AI ecosystem. Section 5 explores how MLOSS reinforces existing data- and hardware-intensive approaches to AI. Section 6 concludes and provides two recommendations.

2 BACKGROUND

To start our discussion, we introduce some key terms and background context that are crucial for developing our argument in the following sections. In particular, we will define open source machine learning (MLOSS) and provide a brief overview of MLOSS history.

2.1 Defining Machine Learning Open Source Software (MLOSS)

Following prior work, we refer to AI as "the use of digital technology to create systems capable of performing tasks commonly thought to require intelligence" and will follow the common practice of using the terms 'machine learning' (ML) and 'artificial intelligence' (AI) interchangeably [4, 12]. We refer to machine learning open source software as computer software released under an open source license that is designed specifically for machine learning. This includes software ranging from frameworks (e.g. PyTorch and Pyro) to 'all-in-one' packages (e.g. scikit-learn) to model development tools (e.g. TensorBoard). It does not include software such as the interactive computing tool Jupyter Notebook which, although often used by machine learning practitioners, was not specifically built to accommodate machine learning.

2.2 A Brief History of MLOSS

We review the history of MLOSS to highlight that the phenomena is new, ubiquitous, and increasingly supported by industry efforts.

The history of open source machine learning can be grouped into three phases, punctuated by two critical events: the 2012 ImageNet competition and the release of TensorFlow in 2015.

- *Phase 1: Grassroots Efforts (pre 2012).* Prior to 2012, there were few large and well maintained MLOSS projects [42]. Andrew Ng's famous *Introduction to Deep Learning* was originally taught in MATLAB, a closed source language. There were some more targeted ML frameworks such as OpenNN and Torch (which later formed the foundation for PyTorch). However, the packages were either very general or difficult to install and use, and lacked features such as GPU support [43].
- *Phase 2: The Rise of Frameworks (2012-2015).* In 2012, a deep convolutional neural network later known as AlexNet handily won the ImageNet competition, attracting significant attention within academic and certain industry communities [26]. Subsequently, a wave of frameworks emerged from academic research labs, including Chainer, Theano and Caffe. Open source software played an important role in the creation of these frameworks — for instance, the creators of Caffe directly cite the decision to open source AlexNet as inspiration for their framework [46]. Simultaneously, there were a number of efforts within industry to develop private frameworks, such as Google's DistBelief. Frameworks for alternative approaches, such as Stan, also appear and start to gain prominence.
- *Phase 3: Industrialization of AI Research (2015-present).* In 2015, Google's decision to open source TensorFlow changed the landscape in a number of ways. First, by deploying over 200 engineers on the project, TensorFlow provided a package that possessed a quality of engineering far above that of other frameworks at the time[44]. This led other companies to release competitor frameworks, such as Amazon's MXNet, Microsoft's CNTK, and (later) Facebook's PyTorch. In this phase, we also witness the increasing prominence of frameworks for alternative AI methods. Gen, a probabilistic programming package within the programming language Julia, is released and begins to be used by researchers.

Now, open source technologies are ubiquitous in modern ML applications. Consider a hypothetical document-processing company. In their stack, they may leverage Detectron2 (an open source object detection model) programmed in PyTorch (an open source framework) developed in Python (an open source language), originally trained on COCO (an open source data set) [29, 38]. This is not the case in many other technical fields, such as animation graphics or sound engineering.

MLOSS is used in the vast majority of ML applications. Most organizations implement machine learning methods through cloud providers like AWS Sagemaker or GCP's AI platform. Within those platforms, the predominant way of implementing models is to build them via existing libraries such as Google's TensorFlow or Facebook's PyTorch [7]. Open source software is even more central to AI research. Paperswithcode, a community resource for practitioners to follow AI research, shows that the vast majority of publicly available research code is written using open source frameworks [39]. This matches data from interviews with AI researchers in both academia and industry, where every single practitioner acknowledged the core role of open source tools to their research process.

Before discussing related work, we'd like to add a caveat. While this paper refers to *machine learning* open source software, much of the focus is on *deep learning* open source software. This is the case for a couple of reasons. First, deep learning tools — especially frameworks such as PyTorch and TensorFlow — are the most popular MLOSS tools ever created. Accordingly, they have played an outsized role in shaping AI research. Second, although the primary examples provided are from deep learning, we have striven to insure that the effects of MLOSS discussed are not unique to deep learning.

3 MLOSS TOOLS PLAY AN OUTSIZED ROLE IN THE CREATION OF ECONOMIC VALUE

Inspired by the observation that open source software is ubiquitous in AI applications, we argue that MLOSS tools play an outsized role in the creation of global economic value. Regardless of whether one cares about economic value for its own sake, this shows that MLOSS has an outsized role in shaping AI's impact on society both now and in the future – if not only because economic incentives drive development. In turn, this also suggests that policies around MLOSS may be particularly effective interventions for those for those seeking to improve the system.

We start by estimating the cost of MLOSS tools based on Github activity. We then construct a cost-benefit estimate for AI using the research of consulting organizations. We argue that the benefit-to-cost ratio of MLOSS tools is at least 100-to-1. In other words, for every dollar invested in MLOSS tools, we expect at least $100 is created within the AI ecosystem. By this logic, our conservative estimate for the global value created by MLOSS is $30B in 2022, a number we expect to grow. To the best of our knowledge, this represents the first estimate of the value of machine learning OSS.

3.1 The Cost of MLOSS Development

We first seek to estimate the cost of annual development of the core MLOSS repositories. Whereas this exercise would be impossible for most closed-source technologies, MLOSS development cost can be estimated from contribution data found on the public repositories on Github.

3.1.1 *Data Collection.*

Defining a Comprehensive List of MLOSS Tools. We found 139 actively maintained MLOSS tools by drawing largely from the list of tools compiled by Chip Huyen, a leading MLOSS researcher and developer [21]. A small majority of the tools came from large technology companies such as Facebook (PyTorch), and growing startups such as HuggingFace (Transformers). It draws from sources ranging from the Linux Foundation's AI Tools, FirstMark's Data and AI Landscape, and suggestions from the AI community via Twitter. We augment this list with a second list from the Journal of Machine Learning Research [1]. To validate the dataset, we asked a number of practitioners in the community and examined other surveys. The practitioners we asked suggested two more tools that were missing, and the Kaggle survey of developers reflected our intuition that the vast majority of developers use the same small number of tools [25]. For this reason, although we don't consider our list authoritative, we believe that the excluded tools will not seriously affect our estimates.

Sampling Contribution Data from MLOSS Repositories. We scrape contribution data from each repository's Github contributions page, which pre-aggregates contributions data at the Contributor-Period level, where the 'period of interest' is user-defined (here the period was chosen to be two weeks). We then randomly select 5 two-week periods from the history of each MLOSS repository. For each of those periods, we record the number of commits and lines modified (added or deleted) for each contributing user in that period.

The result is 3,932 observations of Contributor-Period level data. As an illustration, during the October 16-30th 2017 period, Soumith Chintala (PyTorch co-founder) contributed 10 commits to the PyTorch repository corresponding to 10 commits and 338 lines modified. We also aggregate to the Repo-Period level, a total of 695 observations. As an example, one row records that, during the March 16-30th 2018 period, the Scikit-Learn repository had a total of 4 contributors adding 9 commits of 2134 lines modified.

Summarizing the Data. In Figure 1, we present histograms of the number of commits and lines-modified at both the Contributor-Period (top) and Repo-Period (bottom) level. The y-axis shows the number of commits (or lines modified), and the x-axis shows the log-scaled count of observations corresponding to those values. For visualization purposes, we collapse value observations above the 99% quantile to the 99% quantile (aka winsorize at the 99% level). These resulting distributions seem reasonable. The median Contributor-Period corresponds to 2 commits of 100 lines of code modified. The median active repository receives a commit about 15 times by 3 users in a given two-week period, corresponding to 600 lines of code. However, we also observe the data is right-skewed. Therefore, while these distributions show that our results are not outlier-driven, there is some notable degree of inequality in contributions across MLOSS repositories.

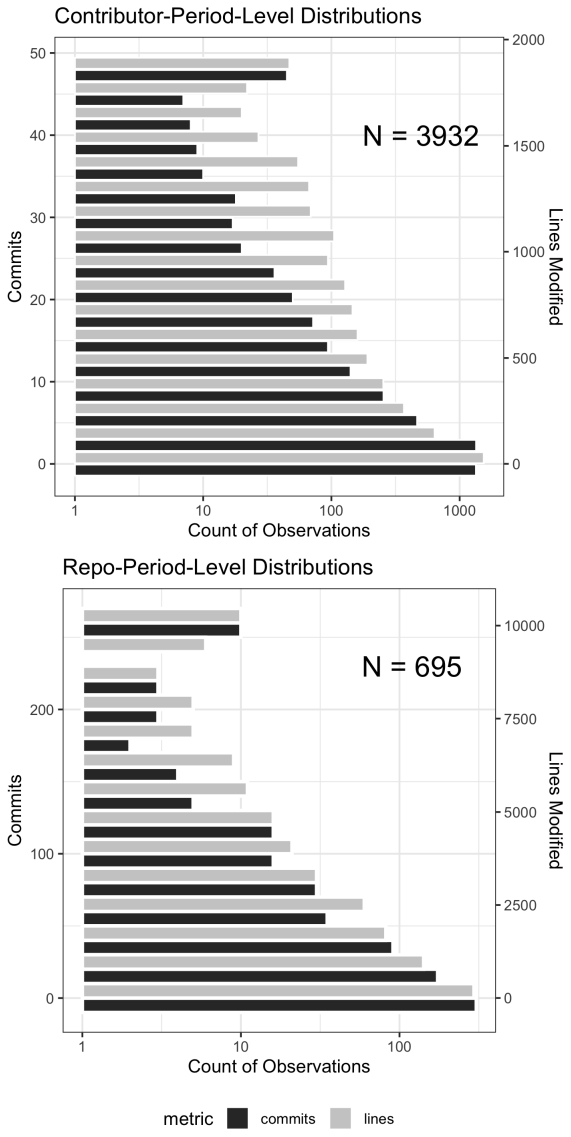

Figure 1: Histogram Plots for Commits and Lines-Modified, at Contributor-Period (Top) and Repository-Period (Bottom) levels.

3.1.2 *Cost Estimation.*

We now exploit our dataset to estimate the cost of these MLOSS repositories. Let a "unit" be either a contributor, a commit, or a line modified. We first estimate cost-per-unit, and then estimate the total number of units-per-year over our list of MLOSS repositories. By multiplying these estimates, we can estimate the cost-per-year of MLOSS development.

Cost-Per-Unit. We estimate this using a reference organization with known developer salaries such as PyTorch (Facebook) or TensorFlow (Google), taken from levels.fyi, a salary information website [16, 17]. We assume an average wage corresponding to the senior engineer level (e.g. L4 or E4), about $300,000 a year. This is an upper-bound on average salary (as Google and Facebook likely pay more than other MLOSS organizations), meaning we calculate

Table 1: Estimated Annual Cost of MLOSS Tools

Unit	Ref. Repo	Cost/Unit	Units/Year	Cost/Year
Commits	pytorch	2.43K	123.90K	300.48M
Contributors	pytorch	300.00K	779.00	233.70M
Lines	tensorflow	34.09	5.49M	187.09M
Lines	pytorch	34.08	5.49M	187.06M
Commits	tensorflow	986.04	123.90K	122.17M

a conservative over-estimate of the cost of MLOSS development. When the unit of interest is contributors, this salary is the cost-per-contributor (e.g. cost-per-unit). For commits and lines modified, we compute the average units per contributor-period (across our 3932 contributor-periods) and then scale this number from the two-week period to the yearly level. We then multiply by our assumed salaries and take the multiplicative inverse to compute a cost-per-unit estimate, according to the following equation:

$$\frac{\$\ \text{Cost}}{\text{Unit}} = \left[\underset{\text{Contrib}\cdot\text{Period}}{\text{AVG}} \left(\frac{\#\ \text{Units}}{\text{Contrib}\cdot\text{Period}} \right) \times \frac{26\ \text{Periods}}{\text{Year}} \times \frac{\text{Contrib}\cdot\text{Year}}{\$\ \text{Salary}} \right]^{-1}$$

Units-Per-Year. We sum across the entire set of MLOSS repositories to estimate the number of units per year. To compute contributors-per-year, we assume that the number of yearly-active contributors is the same as the average number of active contributors in the periods that we observe. This reflects the logic that not all engineers who ever contribute to an open-source project are working on it full-time. Therefore, even if our observed periods do not capture all contributors at an organization, we argue that the average number of active contributors is representative of the organizations investment in MLOSS in general. To compute lines committed or modified per year, we compute the average units contributed across all users for each of our 5x periods per repo. We then sum across all 139 repos, and scale that value to the yearly level.

$$\frac{\text{Units}}{\text{Year}} = \sum_{\text{Repo}} \left[\underset{\text{Period}}{\text{AVG}} \left(\sum_{\text{User}} \frac{\#\ \text{Units}}{\text{User}\cdot\text{Period}} \right) \right] \times \frac{\text{Period}}{2\ \text{Weeks}} \times \frac{52\cdot\text{Weeks}}{\text{Year}}$$

Sensitivity Considerations. We estimate total system cost using our three different units (Contributors, Commits, and Lines Modified) to ensure that our estimates are not driven by a weak assumption about how contribution practices from large projects extrapolate to the rest of our project. Extrapolating via Contributors assumes that different contributors are doing similar work across repositories. By contrast, commits and lines modified present alternative ways of weighting and extrapolating costs to other repos, where we assume that different commits or lines modified are comparable (even if Contributors are not comparable).

We present our cost estimates for each unit in Table 1. We find that the cost of MLOSS ranges between $100-$300MM per year.

3.1.3 Limitations to Cost Estimation. There are several important limitations to our methodology that affect our ability to interpret it as a comprehensive cost estimate of the entire MLOSS system.

Comprehensiveness of MLOSS Tool Data. While we made our list as comprehensive as possible, it's possible that we are missing significant MLOSS projects that contribute to economic value creation and costs in ways that we are missing. Furthermore, given our limited sampling, our estimates may also be driven by sampling error. Despite these concerns, we still believe our estimates capture the correct first-order approximation of costs. First, as argued above, there is significant inequality in MLOSS usage, and our expert-compiled list covers the most important tools with respect to usage. Second, the estimates of our methodology can be easily extended via further sampling to mitigate concerns around sampling noise.

Research Code. A final source of economic value missing in our estimate is the code produced by the research community, which drives the production of AI research. Research code is an important aspect of ML Tools because it enables the creation of new valuable methodologies. However, we are not including research code in our cost estimates, electing to limit our cost estimates to MLOSS. We do this for two reasons. First, in practice, applications of AI models tend to be implemented using the MLOSS included in our list. Therefore, our cost estimates do capture some degree of the costs needed to deploy cutting-edge models. Second, the broader literature on valuation of open source does not consider research costs in their estimates, and excluding research code makes our estimates more comparable to other open-source value estimates [14]. For example, in the OECD estimate of European OSS, they do not consider the research that the OSS tools are built on.

3.2 The Benefits of AI and the role of MLOSS in Economic Value Creation

We now contrast the estimated costs of MLOSS with the benefits of AI to global economic value creation in order to argue that MLOSS plays an outsized role.

3.2.1 Economic Benefits of AI. Rather than developing our own estimate, we briefly summarize AI economic benefit estimates from McKinsey and PWC. Each of these organizations estimates that AI will add roughly $3-5 trillion USD to the global economy in 2022 [32, 36]. To arrive at these estimates, these reports break down AI applications into specific use-cases across a broad range of industries, estimate the value of AI for each use-case separately, and then aggregate across use-cases. The use-cases cover both the reduction of the costs of existing processes and also product and service innovation. While estimating the value of AI for each use-case relies on strong assumptions, the overall size of the value estimate is driven by the sheer number of use-cases that AI applies to, rather than any individual estimate. However, due to the strength of assumptions going into these reports, these estimates should be taken as providing only a rough order of magnitude estimate of the economic contribution of AI. One reassuring factor is that both reports provide estimates that are (independently) within an order of magnitude of each other. We therefore feel confident in concluding that AI provides at least $300 billion in USD to the global economy in 2022 (a reduction from the original estimates by a factor of 10).

3.2.2 Attributing Economic Benefit of AI to MLOSS. We now seek to connect the overall economic benefit of AI with the specific value attributable to one part of AI: MLOSS Tools. While it's hard to

estimate precisely to break down the value of AI into components, one heuristic that has been proposed is the 70-20-10 rule, which argues that AI value creation comes from investments in Processes, Data/Technologies, and Algorithms, in those proportions [3]. We identify MLOSS with the algorithms portion of investment, based on the observation (described in Section 2) that MLOSS tools are ubiquitous and are the primary means through which models are distributed in practice. This implies that MLOSS tools are responsible for $30B/year value created, corresponding to a benefit-cost ratio of over 100-to-1.

3.2.3 Interpreting the Economic Costs and Benefits of MLOSS. This estimated annual value creation ($30B/year) and benefit-cost ratio (100:1) are very large – few other classes of technology can match this level of productivity. By comparison, the OECD estimates that open source software in Europe generates value at a 4:1 ratio [14]. Furthermore, because MLOSS is a public good, as the usage of AI scales globally, MLOSS will provide increasing returns to scale – which will only serve to increase the benefit-cost ratio.

Nevertheless, we think these estimates are reasonable and are consistent with other known estimates found in the literature. Even after our conservative estimates, which we expect to be an upper bound on cost and a lower bound on benefits, we arrived at a 100-to-1 benefit-to-cost ratio. This is consistent with Greenstein and Nagle's estimate of the economic contributions of Apache as between $2B and $12B in 2012 [19].

Although we do not know the exact value attributable to MLOSS, we are confident that it plays a outsized role in global economic value creation. Given that MLOSS tooling is ubiquitous and plays a major role in value creation, we suggest that it is a fruitful arena for fostering AI development. In order to do so, however, we need an understanding of the mechanisms through which MLOSS shapes AI. We explore this in our next section.

4 MLOSS DRIVES AI IMPACT THROUGH STANDARDIZATION, EXPERIMENTATION, AND COMMUNITY CREATION

In this section, we develop a causal model of how MLOSS creates value and shapes the AI research ecosystem.

4.1 Methodology

To develop our model, we conduct a series of interviews of active AI researchers and practitioners. We combined the interviews with a review of MLOSS online archival material. We iteratively analyzed our evidence until we converged upon a consistent model of open source effects that effectively organized our findings.

4.1.1 Interviewees. We selected AI researchers and developers as subjects largely through convenience and theoretical sampling [5]. During our selection of interview subjects, we focused on ensuring that our interviews covered experiences from both industry and academia over a variety of different ML projects. Overall, we conducted 23 interviews: 8 formal interviews for an average of 50 minutes each, and 15 informal interviews with other AI researchers. The background and experiences of our formal interview subjects is listed in Table 2.

Table 2: Overview of Formal Interview Subjects' Background. In selecting our interview subjects, we focused on finding people with a variety of difference experiences.

ID	Institutional Experience	Role	AI Fields
R1	Big Tech	PyTorch Dev	Frameworks Compilers
R2	Big Tech	TensorFlow Dev	Frameworks
R3	University Startup (Cybersecurity)	PhD Researcher	Audio
R4	University Big Tech	PhD Researcher	Robotics RL
R5	University Big Tech Startup (BioTech)	PhD Researcher	RL
R6	University Big Tech Startup (Translation)	PhD Researcher	NLP
R7	Venture Capital Startup (AV)	Investor AI Engineer	Robotics AV
R8	Startup (BioTech)	AI Engineer	Biophysics

4.1.2 Interview Structure. We followed a general interview structure where we asked about the following general questions:

- How did you first get introduced to machine learning?
- What have been your most recent machine learning projects?
- What institutional contexts did you work in, and what tools did you use?
- What are the main technologies that you depend on for your work? How do they fit into your workflow?
- Have you ever used research code from another researcher's project? Why? What was the process for using it like?
- What was the last time you used a new model or technique? What was your process for getting up to speed on it?
- Have you ever shared your own code / tools? Why? What was the process of preparing it like?

When we could, we ask interviewees to expand on points of interest. All interviews had notes written within 24-hours of the interview. We promised confidentiality and received permission to digitally record the formal interviews, allowing us to transcribe them. In total, this produced 150 pages of interview transcripts and notes.

4.1.3 Archival Materials. To form a historical perspective, we examined a variety of sources. These included materials ranging from the PyTorch five-year review to discussion on the EleutherAI community Discord to materials from Stanford's CS230 [2, 11, 37].

4.1.4 Data Analysis. We iterated between information collection and analysis to generate a theory grounded in data [18]. We use two lenses of analysis:

(1) *Thematic analysis*, where form the categories from interview and archival data;
(2) *Theoretical analysis*, where we explain the relationship between the open sourcing, our effect, and economic value.

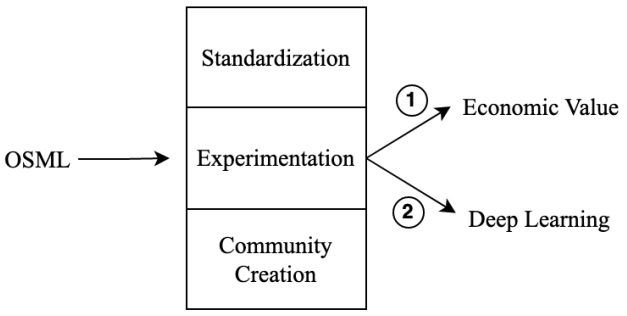

Figure 2: (1) We find that MLOSS produces economic value and shapes the AI ecosystem through three primary effects: standardization, experimentation, and community creation. (2) In Section 5, we will explore how these same factors reinforce the dominant paradigm of deep learning.

Out of the many effects we encountered over our interviews, we identified three core effects of open-sourcing projects: standardizing interfaces, enabling experimentation, and creating communities.

4.2 Findings

We organize our finding into three distinct effects: Standardization, Experimentation, and Community Creation (shown in Figure 2).

4.2.1 Effect 1: Standardizing interfaces. One major effect we observed from our interviews and our own experience with open source tools was standardization — the widespread adoption of a single technology or technique among both users and tool-makers.

Thematic Analysis. We observed standardization at three major levels of analysis.

Programming language and frameworks. Frameworks like PyTorch and TensorFlow provide the core primitives that are used by researchers and developers to construct and train machine learning models. As described in section 2, after a period of divergence in framework development, MLOSS frameworks have significantly consolidated over the past five years.

Our interviewees' experiences substantiate this general trend in framework consolidation; several interviewees started out by working in either older frameworks (R3 worked with Theano) or directly via array-based methods (R5), not a single one of our subjects regularly works with a framework outside of PyTorch, TensorFlow, or JAX today. R3 pointed out that, despite its early prevalence in the ML community, nobody uses MATLAB any more. All researchers that we interviewed emphasized the benefits of using the same framework in terms of their ability to replicate and build on the community's code, as well as sharing their ideas with other researchers.

Model Types. We noted that several of our interviewees converged on working with large neural networks. Traditionally, these models would be hard for resource-constrained researchers to leverage, but open source efforts have made many models widely available. R6: "HuggingFace, for example,... made so many things a lot easier and continues to for a lot of people in NLP who work on large models. I don't really know what I would do without HuggingFace.... there's a sense that I'm missing a big chunk of the field if I'm not working on big models at all."

In order to come to consensus on large models, the research community also needs open source datasets to establish benchmarks. For example, the ImageNet challenge was enabled by the public ImageNet data set. This was essential for establishing the importance of deep convolutional neural networks in 2012. In providing benchmarks, public datasets have also facilitated the movement away from older models such as Markov chains.

User Experience. We also noticed convergence in user experience.

A particularly prominent example that emerged from our interviews was the convergence of frameworks on eager execution over graph-based execution. Eager lets developers print values while running the model. In contrast, graph based requires users to insert placeholder variables in a computational graph.

Several interviewees noted that TensorFlow's default graph execution was counterintuitive and made it harder to learn as a beginner, which later led them to swap to PyTorch. R6 notes: "TensorFlow had such a weird model — you can't print your graph because there's no values, it's just the abstract graph. So I remember I struggled for a long time in the early days. What I've observed is TensorFlow trying to add more of that back into their framework to imitate PyTorch. So, now, you can do things like eager execution."

At this point, there are few differences between the top frameworks. R5 notes "In terms of the specific frameworks themselves... my personal opinion is that there's not a huge difference between all of them – Jax, PyTorch, TensorFlow, etc."

Beyond user interfaces, openness leads to greater integration between tools. R1 notes that the development of PyTorch XLA, which enables PyTorch (Facebook-based software) compatibility with Tensor-Processing-Units (Google-based hardware), was led by the Google research team [40].

Theoretical Analysis. Our explanation for the relationship between open-source and standardization is that open source is a powerful enabler of standardization. As Lerner and Tirole (2005) explain, forming consensus with a private tool is difficult [28]. Open sourcing a technology means that it is free to use, and complementary products, like documentation and tools built on top of it, naturally emerge (a cross-side network effect). This attracts new users, who are now able to share and collaborate with other users on the platform (a direct network effect). By contrast, a closed-source technology presents a friction that makes it difficult for all users to adopt and agree on initially, which prevents the accruing of these network effects.

The core economic benefit of standardization is the creation of natural interoperability — where distinct technological systems are able to exchange services and interact in a useful way. Interoperability improves economic outcomes by lowering the costs necessary to train and transfer skills across domains, as well as adding value through the ability for technologies to work with each other.

4.2.2 Effect 2: Enabling experimentation. Open source projects shaped our interviewees' project preferences, helped them work faster, and gave them new ways of thinking about problems.

Thematic Analysis. We organize our concept of experimentation into three sub-concepts: project choice, development speed-up, and novel conceptualization.

Project Choice. We found that the availability of open-source code had a dramatic effect on the projects that our interviewees worked on.

The availability of code was a primary motivation for some research projects. R4 said "If PyTorch or an equivalent didn't exist, and I was looking at these papers that were, 'hey here's this neural network that can do this task and we don't know how it works, but we're seeing great performance,' I might just let that be. Whereas because those tools are there and I can just grab the person's implementation of this, I can very quickly start to probe it to see if the network is actually learning things and test my explainability methods with it. It's just so much easier for me to work on those types of problems, so they become more attractive." R5 stated "My master's thesis was playing around with adversarial examples… looking at neural-tangent-kernel-based models... If 'neural-tangents' (a key library) disappeared, that would be very annoying. I literally don't know how to do neural tangent stuff without 'neural-tangents', and I would literally have to go back into the papers… which would be pretty bad."

When code was not available or did not work, it changed the scope of the project or whether someone would pursue it. R4 said "I was trying to build a system where the first step was to implement these old interactive RL Frameworks. I was having some problems with that, so it definitely was going really slowly and was off putting to me. So I ended up diverting the project. Instead of implementing those things myself, I ended up doing more of a literature review."

Beyond research, open-source enables startups that work on AI problems with limited resources to exist. R8 argues that if their startup "had to reimplement a deep-learning framework from scratch, that would not be feasible, because we would need to hire people who really understand compilers and CUDA and things like that. If it was closed source, we would pay for it, maybe… but you definitely need to have ML at a certain stage of maturity to allow [our startup] to do what we do."

Faster Research and Development. Most obviously, open-source projects speed up development of new AI applications.

Most subjects felt that, although they conceptually understood open source frameworks such as TensorFlow and PyTorch, these tools vitally lowered the friction and increased the speed of specifying model architectures and training neural networks. R4: "For me, the turnaround could be as short as eight weeks. [Because of existing code] it's very, very quick to get a prototype, and then you're running your experiments. The cycle is super rapid because of the availability of these tools."

MLOSS infrastructure also enables startups to iterate quickly. R7 notes that "these open source tools let you rapidly prototype and iterate, which is important in the early stages of a company, when they're figuring out what their product is"

Novel Conceptualization. The most subtle form of increased experimentation comes from the way that open-source projects change how AI developers conceptualize problems.

One way this shows up is in how developers conceive of what new models and applications can be built. R1 recalls a fellow researcher's comment: "there's no way I would have thought of these ideas if it wasn't for using PyTorch". R3 also notes, "In PyTorch you can have 'if' statements and all sorts of weird things that are not normally part of neural networks, and it back-propagates through them easily. So you have more freedom to experiment with novel ideas and structures because of that." Because of advances in Torch in particular, new kinds of network architectures such as Tree-LSTMs are possible [47].

Theoretical Analysis. Open-sourcing a project enables experimentation because it significantly lowers both the economic and knowledge barriers between project creators and consumers. Especially in the case of MLOSS, given available code, the barriers to reproducing a paper are very low. For example, R3 notes that "if you read a biology paper, there's no way you're going to, in an afternoon, reproduce the results... But in machine learning, that's pretty doable." By lowering the barriers to entry, open-source encourages researchers to enter fields based on the quality of their ideas rather than their prior knowledge-base or institutional circumstance. This model is similar to the one presented in Murray et al 2015, which finds that openness enables researchers to join new fields quickly and opportunistically work on relevant problems in the context of biology research.

Enabling experimentation creates economic value because it leads to the discovery of a variety of machine learning models that enable AI to solve a broad range of problems. This enables AI to solve a diverse breadth of use-cases across a variety of problem domains. Furthermore, it allows for the most effective techniques of different AI subfields to be transferred over rapidly to new subfields – for example the recent transfer of Transformer architectures from NLP problems over to computer-vision problems.

4.2.3 *Effect 3: Community Creation.*

Perhaps the most under-discussed mechanism that we observe is the effect of open-source on community creation. By community, we mean a space for both technology contributors and users to interact. Common digital spaces today are Github, Reddit, and Discord.

Thematic Analysis. We observe that open-sourcing a project leads creators to be more in-touch with users, encourages users to contribute tools themselves, and inspires the creation of related educational materials that make it easier for others to get involved.

Increased Feedback. Open sourcing projects enables greater feedback on the project, which improves its design. Soumith Chintala, a co-creator of PyTorch, emphasized the role of the openness of the community in helping to direct the prioritization of PyTorch and making it a great user experience. "[Soumith] read the entire volume of information that [his] community produced – github issues, forum posts, slack messages, twitter posts, reddit and hackernews comments. It was an incredibly useful signal..." [6].

This effect extends beyond the focal project – R2 noted that, because of the open-nature of the feedback, PyTorch had an advantage as "second mover". PyTorch learned from the mistakes of the previous TensorFlow framework.

For researchers, open-sourcing code enables their ideas to be more closely validated. R4 notes "If I make mistakes, I want somebody else to publish a paper saying, 'hey, you got this wrong...' I want this pursuit of truth and openness is the best way to get there."

Users Become Contributors. Open source machine learning software encourages and makes possible broad involvement. R4 comments "If [research] required you to build your own system... we would see many fewer people participating in this field." R5 observes a cultural aspect associated with open source projects not shared by closed source projects: "Open source incentivizes people to play around with the frameworks. I don't see people say, 'Here's some cool thing I did in Matlab, come check it out.' But people will say 'PyTorch is a cool framework, and here's something I made while messing around in PyTorch.' And they'll share it in the blog post..."

Furthermore, MLOSS encourages unlikely participants to contribute to projects. Consider EleutherAI community, an open-source community that grows and coordinates primarily through their Discord Server. One undergrad who contributes to the project writes "One day during the pandemic summer of 2020, I found myself in this strange dream-like place, a community of international Machine Learning flaneurs who somehow became convinced that they could actually make history. At first, I thought it would just be a fun place to discuss new AI developments. But I soon discovered that yeah, these people are serious about their ambitions, and more thrillingly they actually would like to have me on board! As it turns out, the fact that Machine Learning engineers despise JavaScript (while still needing it) become [sic] my entry ticket to some of the coolest projects I ever worked on."

Improved Educational Materials and Settings. We observe that open-sourcing tools inspire the community to develop associated educational material to extend the reach of the user base. All of our interviewees entered the field through openly available education materials on AI – ranging from Nielsen's online book on Neural Networks to Andrew Ng's CS231 course at Stanford. R3 notes that "I studied [the Nielsen textbook] on my own time and got very interested because I actually realized that this whole thing is not as complicated as I thought it would be. I could actually run the example and eventually started building some of my own things."

Open source communities incentivize the creation of high-impact educational settings. R2 comments "we [Google] escalated from (just) teaching university students in the US to going to these road shows, because we also obviously wanted to teach people in all sorts of different emerging markets. TensorFlow is an international platform and is adopted by people everywhere, so… we teach them colab, introducing them to colab, helping them connect to the TPU or GPU accelerator so that they can run a model in their browser now they don't have to worry about actually installing it."

Theoretical Analysis. The formation of open-source communities has been studied extensively in the literature, with prior explanations focusing on desires to reciprocate in response to someone else's effort, to have impact, or to gain a reputation in a way that is useful [14]. We think all of these mechanisms are likely at play in the open-source machine-learning setting. However, we note one final mechanism for community involvement – participating in these communities makes the products themselves better, in a way that benefits the user.

Community creation creates economic value because it lowers the cost of using these tools and increases the number of available applications. By encouraging community members to become well-versed in the available tools and models, open-sourcing also makes it easier for firms to find the necessary labor needed to implement machine learning models that meet their organizational needs.

4.2.4 MLOSS vs OSS. At this stage, there may be a natural question that has occurred to the reader: *how is MLOSS conceptually different from open source software?* Here, we'll offer two preliminary thoughts. First, our model of the effects of MLOSS captures some effects that are common to other OSS as well some effects that are more unique to MLOSS. For example, the act of open sourcing facilitates standardization and the creation of communities in MLOSS and OSS more generally. However, MLOSS seems to play a particularly important role in enable experimentation early-stage research and development - something that we do not observe in many other OSS technologies (e.g., operating systems). Finally, we consider this an excellent question that is worthy of deeper investigation beyond what we discuss in this paper. Second, whether there are significant conceptual differences between MLOSS and OSS more generally does not diminish the importance of studying MLOSS. MLOSS is important because ML is important. Nevertheless, we think that this question is important and deserving of further consideration beyond our brief discussion here.

4.2.5 Summary. In summary, we believe open-source creates economic value through three distinct intermediate mechanisms: standardization, experimentation, and community creation. These concepts are represented in Figure 2. However, as part of the same analysis process, we came to realize that these same mechanisms that create economic value also potentially lead to the selective acceleration of research. We expand on this concept in the following section.

5 DISCUSSION: INCENTIVES FOR MLOSS AND IMPLICATIONS FOR AI DEVELOPMENT

In this section, we apply our model to understand how incentives for MLOSS have shaped the AI research ecosystem. First, we describe how economic incentives led businesses to fund MLOSS development in a way that encouraged research attention to converge on deep learning as a paradigm in AI research. Second, we examine data tools as part of the next frontier of ML technologies. Because business incentives for data tools differ from those of deep learning frameworks, we suggest they may not be provided as open source software.

5.1 MLOSS Reinforces the Deep Learning Paradigm

5.1.1 Economic Incentives Led Businesses to Support Deep Learning Tool Development. The dominant tools for machine learning are primarily developed by large technology companies. There are three main reasons: talent acquisition, technology control, and commercial benefit. Before discussing these points, we'd like to briefly acknowledge that incentives differ among types of businesses; smaller companies generally are more motivated by talent acquisition and commercial benefit than technology control. Because of their larger influence, our focus here will be the larger companies.

First, by developing highly regarded tools, companies gain prestige in the ML researcher community. Over the course of many conversations with dozens of ML practitioners, it was obvious that open source tool providers are held in high esteem. Researchers and practitioners, when prompted, are quick to express gratitude for well-engineered tools. One developer working on a popular framework told us that their primary motivation for working in a large technology company was to work on their open source framework.

Second, sponsoring an open source tool gives the sponsor significant indirect power within the ecosystem. Large technology companies often patent at the same time that they open source their software [41]. Patenting ensures that the companies are protected from intellectual property lawsuits while open sourcing facilitates mass adoption. Although open source norms are often strong enough to prevent companies from locking in design decisions that are harmful to other members in the ecosystem, companies can change community focus by prioritizing, for instance, one type of hardware over another. At the very least, providing a critical component of infrastructure ensures that key pieces of future research will be compatible with their existing system.

Finally, large technology companies accrue direct commercial benefits from the release of open source tools. One may observe that three of four of the largest funders of open source deep learning software (Facebook, Google, Microsoft, and Amazon) sell either cloud computing, or else sell advertising as part of their core businesses. As cloud computing providers, these companies benefit from increased demand for compute from others organizations seeking to build their own ML models. For advertising, improvements in large language models increases the value they can capture. In this sense cloud computing and advertising services are tightly coupled to deep learning capabilities. Deep learning can therefore be understood as a complementary capability to the commercial services of large technology companies.

5.1.2 Non Deep Learning Paradigms have Worse Support. One outcome of concentrated support for deep learning tools is that other paradigms for AI research (such as probabilistic machine learning [27], rule based expert systems [10], and automated planning [15]) do not benefit from similar levels of technical development.

For example, the two most popular open source tools in deep learning and in automated planning are, respectively, PyTorch and FastDownward [20]. As a tool developed largely by the Facebook AI Research term, PyTorch is incredibly well supported and well documented. By contrast FastDownward requires non-trivial installation steps and basic knowledge of operating systems to handle downloading a compressed bundle of files and managing their installation manually. The project supports Linux, macOS, and Windows, but does not appear to have support for GPUs or more exotic operating systems. Furthermore, it's difficult to get immediate support if a user runs into technical issues: In most of the bug queries we tried, a straightforward search via a search engine did not yield answers, and we had to turn to their custom forum. We do not mean to disparage FastDownward, which appears to be a well maintained project with clear documentation. Our point here is that it is very difficult for a much smaller community and project to match the support for one that is so much better resourced in terms of engineers.

5.1.3 MLOSS Can Shift Attention Orthogonally to Scientific Merit. The quality of available tools affects how researchers choose the paradigm they work in. Given the friction of using a tool like FastDownward and the ease of experimenting with deep learning frameworks and the documentation produced by its communities, researchers are (all else equal) more likely to work on problems in deep learning. These may be tasks like investigating the outputs of large language models, which involve large amounts of data, rather than, for instance 2D scene navigation, where problem formulation and algorithmic construction is more important [8].

While deep learning has clearly been remarkably successful at a wide range of tasks, the literature on shortcomings of deep learning is extensive [30, 31, 33, 35]. The main weaknesses are the lack of interpretability and the massive amount of data needed to make these systems work, which may not be realistic for many important applications. By contrast, a paradigm like automated planning may provide formal guarantees that are important for high-stakes decisions such as flight autopilot software. Paradigms such as probabilistic programming allow inference using much less data [27]. Each paradigm has distinct affordances that lend themselves to different problems and different tooling. We agree with Dotan and Milli that "progress" is value-laden and not objective. As they point out, one big result of the 2012 Imagenet competition was the mass adoption of benchmarks that favored data and compute-rich environments [13]. Different tasks lend themselves to different paradigms. If we want a plurality of needs to be addressed by AI, we should be careful about deep learning capturing community attention and consider important problems that are underincentivized within deep learning yet relevant to AI.

5.2 The Future of ML Tools

Community attention is shifting away from framework development. Soumith Chintala, one of the creators of PyTorch, writes "With PyTorch and TensorFlow, you've seen the frameworks sort of converge... the next war is compilers for the frameworks… a lot of innovation is waiting to happen" [24]. Similarly, we now briefly turn our attention beyond frameworks to consider the next frontier of ML tooling: data-centric tooling.

Data-centric tools are the technologies that assist with producing, inspecting, managing, and service the data that is used to train ML models. Data work is essential to ML projects – for most data-intensive projects, a majority of the time is spent preparing the data [22]. However, despite its central importance to model-building, working with data is ad-hoc in practice. In industry, large companies also seem to build their own data pipelines, but rather than a single PhD establishing a custom pipeline for their experiment, companies like Tesla will have dozens of ML engineers working on a highly efficient patented pipeline [45] that their researchers can easily build on. Although there are several startups attempting to solve this problem, ranging from Snorkel AI to Octopai, there has yet to be standardization or a consolidation on one particular tool. Especially among researchers, there is far from a 'PyTorch for data'. This has led early deep learning researcher Andrew Ng to argue for 'data-centric AI': "[h]old the code fixed and iteratively improve the data" [34]. He argues that data has been heavily neglected (1% of research on data improvement vs 99% of research on model improvement),

and helped to start the 2021 NeurIPS Data-Centric AI workshop [22].

Will future data tooling be open source? Currently, the companies with the largest presence in the data labeling space like Amazon (Mechanical Turk) and Scale AI are proprietary platforms. However, there are notable exceptions like Databricks, a $38B enterprise data company that builds on top of the open source framework MLFlow.

We argue that whether data tools are open sourced will depend in part on the incentives to produce those tools – which in turn depends on the scale of the task. Systems that require petabytes of data, such as Tesla's self-driving car pipeline, are far less likely to be open sourced than the gigabyte-sized experiments at a university or small startup. Since academic researchers are unlikely to use large-scale data tooling, open sourcing such a tool wouldn't attract talent nor grant prestige within academia in the same way that open sourcing a framework could. Furthermore, the data pipeline is often the source of competitive differentiation among large companies that use AI, with much clearer direct commercial benefits to keeping the technology closed source. One plausible scenario is a bifurcation between large and small scale tools. The large scale tools are provided by proprietary platforms like Scale whereas some fraction of the small scale data tools are open source and freely available to researchers.

6 CONCLUSION AND RECOMMENDATIONS

We have argued that MLOSS plays a large role in creating economic value. Next, we developed a model of the effect of how open sourcing machine learning tools shapes the ecosystem. Finally, we explored incentives behind MLOSS and how MLOSS may selectively reinforce deep learning as a paradigm.

Trends in tooling point towards increasing concentration of capabilities and influence. We are heading towards a future where fewer and fewer firms shape AI. This may allow for easier regulation; governments have in the past successfully demonstrated the ability to regulate monopolies emerging from general purpose technologies like electricity. However, it may be much more concerning that these capabilities are developing far more rapidly than our wisdom of how to control our technologies. We conclude with two recommendations.

First, we recommend further study of the effects of machine learning tools. How are the incentives similar or different for MLOSS vs other OSS? Should we expect data tools and pretrained model tools to develop similarly to frameworks (i.e., will they be open source?) What are the trends in probabilistic programming tools? A better understanding of the factors shaping AI will help us govern it.

Second, we recommend support of open source data tooling. The paths for these tools are not set in stone, and funding at this stage can have a large influence on the practice of AI in the near future. The business incentives for MLOSS data-centric tooling are different from the incentives for framework development. Insofar as one believes that open source tools are important for research quality or fairness among research subjects, as Jo and Gebru argue [23], support for open source data tools could be a helpful lever to create further value. This need not be through the creation of an entirely new project, but rather support for less-developed data capabilities. Furthermore, we believe that data tooling is not inclined to misuse in a way that tools for pre-trained models may be, and are thus a safer investment. Balancing a careful understanding of AI with concerns of fairness and safety is crucial to wisely steer the trajectory of our technology in the face of the daunting challenges that behold us.

A ANALYSIS CODE AND MLOSS TOOL LIST

In the spirit of this paper, we've made our data and analysis code open source. This includes our list of MLOSS tools. Interested readers can find our work on Github at https://github.com/Yichabod/OSML_value/.

REFERENCES

[1] 2022. Machine Learning Open Source Software. Website. https://www.jmlr.org/mloss/

[2] Eleuther AI. 2022. Join the EleutherAI Discord Server! Website. Retrieved 2022-03-01 from https://discord.com/invite/zBGx3azzUn

[3] BCG. 2020. Artificial Intelligence and AI at Scale. Blogpost. https://www.bcg.com/en-us/capabilities/digital-technology-data/artificial-intelligence

[4] Miles Brundage, Shahar Avin, Jack Clark, Helen Toner, Peter Eckersley, Ben Garfinkel, Allan Dafoe, Paul Scharre, Thomas Zeitzoff, Bobby Filar, Hyrum Anderson, Heather Roff, Gregory C. Allen, Jacob Steinhardt, Carrick Flynn, Seán Ó hÉigeartaigh, Simon Beard, Haydn Belfield, Sebastian Farquhar, Clare Lyle, Rebecca Crootof, Owain Evans, Michael Page, Joanna Bryson, Roman Yampolskiy, and Dario Amodei. 2018. The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation. *arXiv:1802.07228 [cs]* (Feb. 2018). http://arxiv.org/abs/1802.07228 arXiv: 1802.07228.

[5] Kathy Charmaz. 2006. *Constructing grounded theory.* Sage Publications, London ; Thousand Oaks, Calif.

[6] Soumith Chintala. 2021. Growing open-source. Blogpost. https://soumith.ch/posts/2021/02/growing-opensource/

[7] Google Cloud. 2022. Introduction to built-in algorithms > AI Platform Training. Website. Retrieved 2022-03-01 from https://cloud.google.com/ai-platform/training/docs/algorithms

[8] International Planning Conference. 2022. Booklet & Papers. Website. Retrieved 2022-03-01 from https://ipc.hierarchical-task.net/results/booklet-and-papers

[9] Allan Dafoe. 2018. AI Governance: A Research Agenda. University of Oxford. https://www.fhi.ox.ac.uk/wp-content/uploads/GovAI-Agenda.pdf

[10] Randall Davis and Jonathan J King. 1984. The origin of rule-based systems in AI. *Rule-based expert systems : The MYCIN experiments of the Stanford heuristic programming project / Edited by Bruce G. Buchanan and Edward H. Shortliffe* (1984). Place: S.l. Publisher: s.n. OCLC: 848324685.

[11] Stanford CS Department. 2022. CS230 Deep Learning. Website. Retrieved 2022-03-01 from https://cs230.stanford.edu/

[12] Joseph Donia and Jay Shaw. 2021. Co-design and Ethical Artificial Intelligence for Health: Myths and Misconceptions. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society.* Association for Computing Machinery, New York, NY, USA, 77. https://doi.org/10.1145/3461702.3462537

[13] Ravit Dotan and Smitha Milli. 2019. Value-laden Disciplinary Shifts in Machine Learning. *arXiv:1912.01172 [cs, stat]* (Dec. 2019). http://arxiv.org/abs/1912.01172 arXiv: 1912.01172.

[14] European Commission. Directorate General for Communications Networks, Content and Technology. 2021. *The impact of open source software and hardware on technological independence, competitiveness and innovation in the EU economy: final study report.* Publications Office, LU. https://data.europa.eu/doi/10.2759/430161

[15] Richard E. Fikes and Nils J. Nilsson. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2, 3 (Dec. 1971), 189–208. https://doi.org/10.1016/0004-3702(71)90010-5

[16] Levels FYI. 2022. Facebook E4 Software Engineer Compensation. Website. Retrieved 2022-03-01 from https://www.levels.fyi/company/Facebook/salaries/Software-Engineer/E4/

[17] Levels FYI. 2022. Google L4 Software Engineer Compensation. Website. Retrieved 2022-03-01 from https://www.levels.fyi/company/Google/salaries/Software-Engineer/L4/

[18] Barney Glaser and Anselm Strauss. 1967. *The Discovery of Grounded Theory.* Weidenfeld and Nicolson, London.

[19] Shane Greenstein and Frank Nagle. 2014. Digital dark matter and the economic contribution of Apache. *Research Policy* 43, 4 (May 2014), 623–631. https://doi.org/10.1016/j.respol.2014.01.003

[20] M. Helmert. 2006. The Fast Downward Planning System. *Journal of Artificial Intelligence Research* 26 (July 2006), 191–246. https://doi.org/10.1613/jair.1705

[21] Chip Huyen. 2020. Machine Learning Tools Landscape v2 (+84 new tools). Blogpost. https://huyenchip.com/2020/12/30/mlops-v2.html

[22] Indeed.com. 2022. Probabilistic Programming Job Query. Website. https://www.indeed.com/jobs?q=Probabilistic%20Programming&l&vjk=4ac08ba6112c10e2

[23] Eun Seo Jo and Timnit Gebru. 2020. Lessons from archives: strategies for collecting sociocultural data in machine learning. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency (FAT* '20)*. Association for Computing Machinery, New York, NY, USA, 306–316. https://doi.org/10.1145/3351095.3372829

[24] Khari Johnson. 2020. Top minds in machine learning predict where AI is going in 2020. Website. Retrieved 2022-05-20 from https://venturebeat.com/2020/01/02/top-minds-in-machine-learning-predict-where-ai-is-going-in-2020/

[25] Kaggle. 2021. State of Data Science and Machine Learning 2021. Website. Retrieved 2022-03-01 from https://www.kaggle.com/kaggle-survey-2021

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, Vol. 25. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html

[27] Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. 2017. Building machines that learn and think like people. *Behavioral and Brain Sciences* 40 (2017). https://doi.org/10.1017/S0140525X16001837 Publisher: Cambridge University Press.

[28] Josh Lerner and Jean Tirole. 2005. The Economics of Technology Sharing: Open Source and Beyond. *Journal of Economic Perspectives* 19, 2 (June 2005), 99–120. https://doi.org/10.1257/0895330054048678

[29] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. 2015. Microsoft COCO: Common Objects in Context. *arXiv:1405.0312 [cs]* (Feb. 2015). http://arxiv.org/abs/1405.0312 arXiv: 1405.0312 version: 3.

[30] Gary F. Marcus. 1998. Rethinking eliminative connectionism. *Cognitive Psychology* 37, 3 (1998), 243–282. https://doi.org/10.1006/cogp.1998.0694 Place: Netherlands Publisher: Elsevier Science.

[31] Gary F. Marcus. 2001. *The Algebraic Mind: Integrating Connectionism and Cognitive Science*. A Bradford Book, Cambridge, MA, USA.

[32] McKinsey. 2018. Sizing the potential value of AI and advanced analytics | McKinsey. Blogpost. https://www.mckinsey.com/featured-insights/artificial-intelligence/notes-from-the-ai-frontier-applications-and-value-of-deep-learning

[33] Marvin Minsky and Seymour A. Papert. 1969. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, MA, USA.

[34] Andrew Ng. 2021. MLOps: From Model-centric to Data-centric AI. Website. , 29 pages.

[35] Steven Pinker and Alan Prince. 1988. On language and connectionism: Analysis of a parallel distributed processing model of language acquisition. *Cognition* 28, 1-2 (1988), 73–193. https://doi.org/10.1016/0010-0277(88)90032-7 Place: Netherlands Publisher: Elsevier Science.

[36] PWC. 2018. PwC's Global Artificial Intelligence Study: Sizing the prize. Blogpost. https://www.pwc.com/gx/en/issues/data-and-analytics/publications/artificial-intelligence-study.html

[37] PyTorch. 2022. PyTorch Turns 5! Video. https://www.youtube.com/watch?v=r7qB7mKJOFk

[38] Facebook AI Research. 2021. Detectron2 A PyTorch-based modular object detection library. Blogpost. Retrieved 2022-03-01 from https://ai.facebook.com/blog/-detectron2-a-pytorch-based-modular-object-detection-library-/

[39] Facebook AI Research. 2022. Papers With Code Trends. Website. Retrieved 2022-03-01 from https://paperswithcode.com/trends

[40] Facebook AI Research. 2022. PyTorch - XLA. Github Repository. Retrieved 2022-03-01 from https://github.com/pytorch/xla original-date: 2018-11-05T22:42:04Z.

[41] Tom Simonite. 2018. Despite Pledging Openness, Companies Rush to Patent AI Tech. Website. Retrieved 2022-05-20 from https://www.wired.com/story/despite-pledging-openness-companies-rush-to-patent-ai-tech/

[42] Sören Sonnenburg, Mikio L. Braun, Cheng Soon Ong, Samy Bengio, Leon Bottou, Geoffrey Holmes, Yann LeCun, Klaus-Robert Müller, Fernando Pereira, Carl Edward Rasmussen, Gunnar R&#228, tsch, Bernhard Schölkopf, Alexander Smola, Pascal Vincent, Jason Weston, and Robert Williamson. 2007. The Need for Open Source Software in Machine Learning. *Journal of Machine Learning Research* 8, 81 (2007), 2443–2466. http://jmlr.org/papers/v8/sonnenburg07a.html

[43] Synced Review. 2020. A Brief History of Deep Learning Frameworks | Synced. Blogpost. https://syncedreview.com/2020/12/14/a-brief-history-of-deep-learning-frameworks/

[44] The TWIML AI Podcast with Sam Charrington. 2017. Soumith Chintala Interview - Pytorch: Fast Differentiable Dynamic Graphs in Python. Video. https://www.youtube.com/watch?v=am895oU6mmY

[45] Timofey Uvarov, Brijesh Tripathi, and Evgene Fainstain. 2022. Data Pipeline and Deep Learning System for Autonomous Driving. Patent. https://patentscope.wipo.int/search/en/detail.jsf?docId=WO2019245618

[46] Edge AI + Vision. 2016. The Caffe Deep Learning Framework: An Interview with the Core Developers. Blogpost. https://www.edge-ai-vision.com/2016/01/the-caffe-deep-learning-framework-an-interview-with-the-core-developers/

[47] Yao Zhou, Cong Liu, and Yan Pan. 2016. Modelling Sentence Pairs with Tree-structured Attentive Encoder. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. The COLING 2016 Organizing Committee, Osaka, Japan, 2912–2922. https://aclanthology.org/C16-1274