MITIGATION TECHNIQUES TO OVERCOME DATA HARM IN MODEL BUILDING FOR ML

Ayse Arslan

Oxford Alumni of Northern California, Santa Clara, USA

ABSTRACT

Given the impact of Machine Learning (ML) on individuals and the society, understanding how harm might be occur throughout the ML life cycle becomes critical more than ever. By offering a framework to determine distinct potential sources of downstream harm in ML pipeline, the paper demonstrates the importance of choices throughout distinct phases of data collection, development, and deployment that extend far beyond just model training. Relevant mitigation techniques are also suggested for being used instead of merely relying on generic notions of what counts as fairness.

KEYWORDS

Fairness in machine learning, societal implications of machine learning, algorithmic bias, AI ethics, allocative harm, representational harm.

1. INTRODUCTION

Algorithms do not "decide" or "guide" or "theorize"—the human-beings developing those algorithms do. Yet, developers and software engineers are often unable to anticipate the consequences that arise when their code and embedded assumptions interact with a complex (and unequal) world, and how that interaction will reinforce (or misguide) our interpretations of human behaviour. Algorithms, in other words, do not only help us parse data; they also generate data that will then be analysed as resulting from human behaviour.

This paper provides a framework for understanding different sources of harm throughout the ML life cycle in order to offer techniques for mitigations based on an understanding of the data generation and development processes rather than relying on generic assumptions of what being fair means.

2. EXISTING WORK

An ML algorithm aims to find patterns in a (usually massive) dataset, and to apply that knowledge to make a prediction about new data points (e.g: photos, job applicant profiles, medical records etc.) (Cusumano et al., 2019; Parker, van Alstyne, & Choudary, 2016). As a result, problems can arise during the data collection, model development, and deployment processes that can lead to different harmful downstream consequences.

This paper refers to the concept of "harm" or "negative consequences" caused by ML systems. ML (Machine Learning) can be defined as the overall process inferring in a statistical way from existing data in order to generalize to new, unseen data.

David C. Wyld et al. (Eds): NLPTA, EDU, DSA, IoTE, VLSI, DPPR, ACITY, AIAA, CNDC - 2021
pp. 195-207, 2021. CS & IT - CSCP 2021 DOI: 10.5121/csit.2021.111916

Deep reinforcement learning—where machines learn by testing the consequences of their actions—combines deep neural networks with reinforcement learning, which together can be trained to achieve goals over many steps. Most machine learning algorithms are good at perceptive tasks such as recognizing a voice or a face. Yet, deep reinforcement learning can learn tactical sequences of actions, things like winning a board game or delivering a package. In the real world, human-beings are able to very quickly parse complex scenes where simultaneously many aspects of common sense related to physics, psychology, language and more are at play.

A high-level overview of a ML-based model might look as follows:

Data Collection

Before any analysis or learning happens, data must first be collected. Compiling a dataset involves identifying a target population (of people or things), as well as defining and measuring features and labels from it. Often, ML practitioners use existing datasets rather than going through the data collection process.

Data Preparation

Depending on the data modality and task, different types of preprocessing may be applied to the dataset before using it.

As Figure 1 displays, the data generation process begins with data collection. This process involves defining a target population and sampling from it, as well as identifying and measuring features and labels. This dataset is split into training and test sets. Data is also collected (perhaps by a different process) into benchmark datasets.

Model Development

Models are then built using the training data (not including the held-out validation data).

As seen in Figure 1, a model is defined, and optimized on the training data. Test and benchmark data is used to evaluate it, and the final model is then integrated into a real-world context. This process is naturally cyclic, and decisions influenced by models affect the state of the world that exists the next time data is collected or decisions are applied. The red color indicate where in this pipeline different sources of downstream harm might arise.

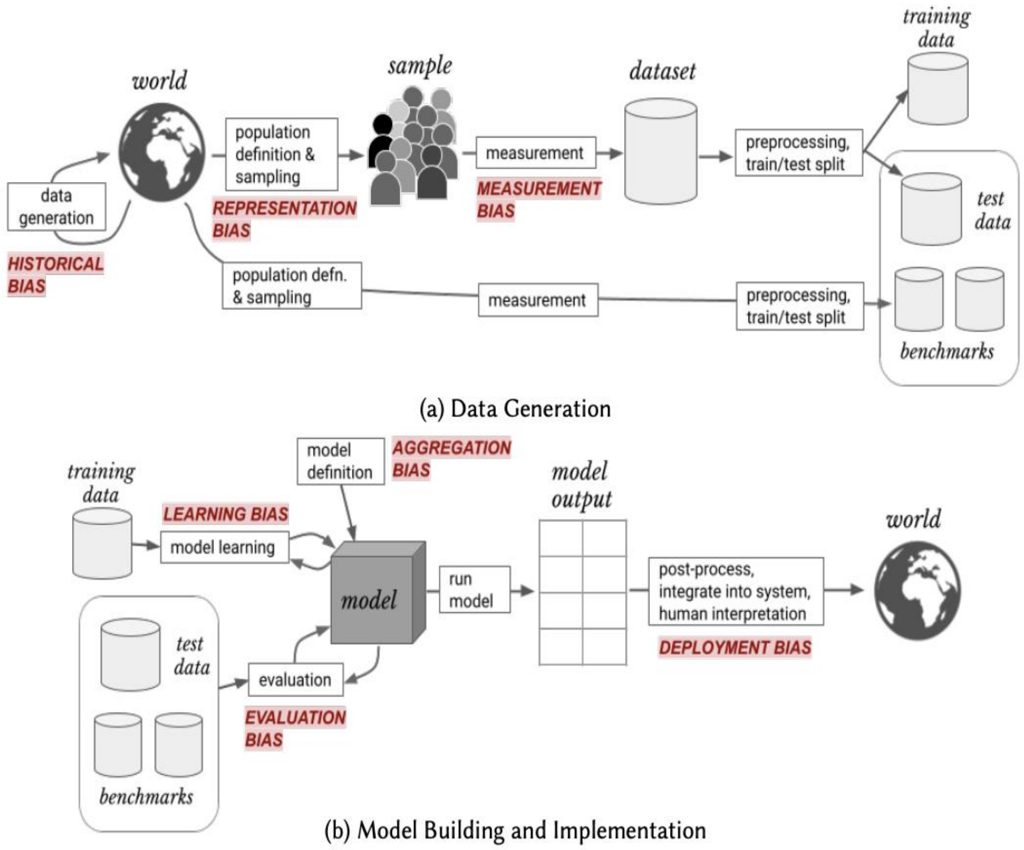

Fig. 1. Overview of ML data generation and model development

When it comes to building models for images, a training dataset can be developed for a particular detection model by manually labelling images. In case of a low precision and recall, for example due to the variety of natural and artificial features, following methods can be utilized to improve performance:

- The use of mix-up as a regularization method, where random training images are blended together by taking a weighted average. Though mix-up is originally proposed for image classification, it can be used for semantic segmentation. Regularization is important in general for segmentation task, as even with 100k training images, the training data might not capture the full variation of terrain, atmospheric and lighting conditions that the model is presented with at test time, and hence, there is a tendency to overfit.

- Another method is the use of unsupervised self-training in which the output of the best detection model from the previous stage is used as a 'teacher' to then train a 'student' model that makes similar predictions from augmented images. In practice, this could reduce false positives and sharpen the detection output. In order to overcome the issue of "blobby" detections, one can use distance weighting to adapt the loss function for making correct predictions near boundaries. During training, distance weighting places greater emphasis at the edges by adding weight to the loss — particularly where there are instances that nearly touch.

When visually inspecting the detections for low-scoring images, various causes can be noted such as problematic label errors. In order to shed light onto which methods contribute most to the final performance, mean average precision (mAP) can be measured. Distance weighting, mixup and the use of ImageNet pre-training are most common factors for the performance of the supervised learning baseline.

Model Evaluation

After the final model is chosen, the performance of the model on the test data is reported. The test data is not used before this step, to ensure that the model's performance is a true representation of how it performs on unseen data. Aside from the test data, other available datasets — also called benchmark datasets — may be used to demonstrate model robustness or to enable comparison to other existing methods.

Model Post-processing

Once a model has been trained, there are various post-processing steps that may needed. For example, if the output of a model performing binary classification is a probability, but the desired output to display to users is a categorical answer, there remains a choice of what threshold(s) to use to round the probability to a hard classification.

Model Deployment

There are many steps that arise in deploying a model to a real-world setting. For example, the model may need to be changed based on requirements for explainability or apparent consistency of results, or there may need to be built-in mechanisms to integrate real-time feedback. Importantly, there is no guarantee that the population a model sees as input after it is deployed (here, we will refer to this as the use population) looks the same as the population in the development sample.

The algorithms used to parse and analyze those data become commercial black boxes. Barocas et al. [4] provide a useful framework for thinking about how these consequences actually manifest, splitting them into allocative harms (when opportunities or resources are withheld from certain people or groups) and representational harms (when certain people or groups are stigmatized or stereotyped). For example, algorithms that determine whether someone is offered a loan or a job [12, 36] risk inflicting allocative harm. We, human-beings are fallible in making unbiased decisions ourselves and algorithms can actually help us detect human-generated (and socially reinforced) discrimination (Kleinberg et al., 2020; Mullainathan, 2019).

There's a large body of work on testing common sense and reasoning in AI systems. Many of them are focus on natural language understanding, including the famous Turing Test and Winograd schemas. In contrast, the AGENT project focuses on the kinds of reasoning capabilities humans learn before being able to speak. The idea behind the AGENT (Action, Goal, Efficiency, coNstraint, uTility) test by DeepMind Team is to assess how well AI systems can mimic this basic skill, what they can develop psychological reasoning capabilities, and how well the representations they learn generalize to novel situations.

According to the DeepMind Team, the AGENT test takes place in two phases:

- First, the AI is presented with one or two sequences that depict the agent's behavior. These examples should familiarize the AI with the virtual agent's preferences.

- After the familiarization phase, the AI is shown a test sequence and it must determine whether the agent is acting in an expected or surprising manner.

The designers of the tests have included human inductive biases, which means the agents and environment are governed by rules that would be rational to humans (e.g., the cost of jumping or climbing an obstacle grows with its height). This decision helps make the challenges more realistic and easier to evaluate.

The Deepmind researchers tested the AGENT challenge on two baseline AI models. The first one, Bayesian Inverse Planning and Core Knowledge (BIPaCK), is a generative model that integrates physics simulation and planning.

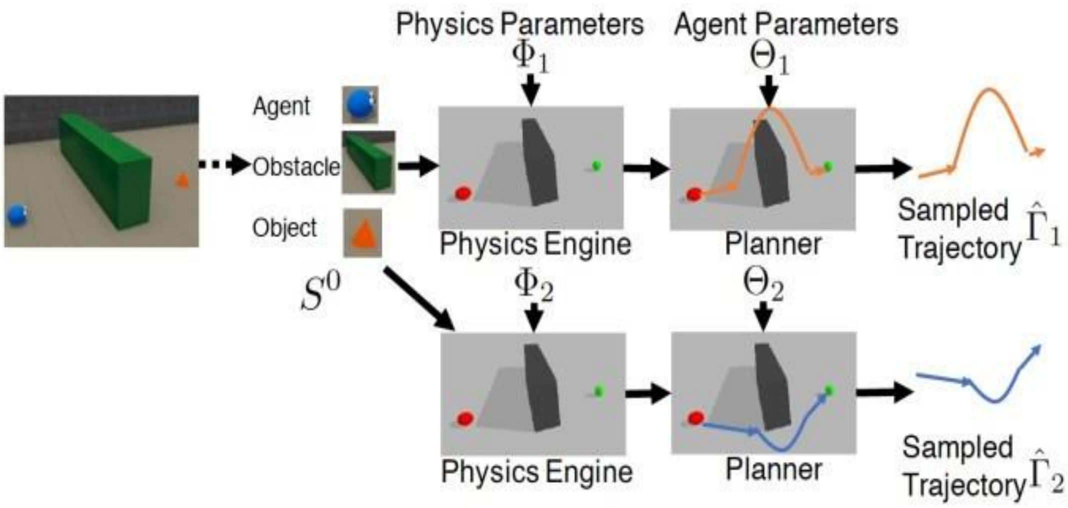

Fig. 2. Overview of BIPaCK Model

As seen Figure 2., the BIPaCK model uses planner and physics engines to predict the trajectory of the agent. The model uses the full ground-truth information provided by the dataset and feeds it into its physics and planning engine to predict the trajectory of the agent. However, in the real world, AI systems don’t have access to precisely annotated ground truth information and must perform the complicated task of detecting objects against different backgrounds and lighting conditions.

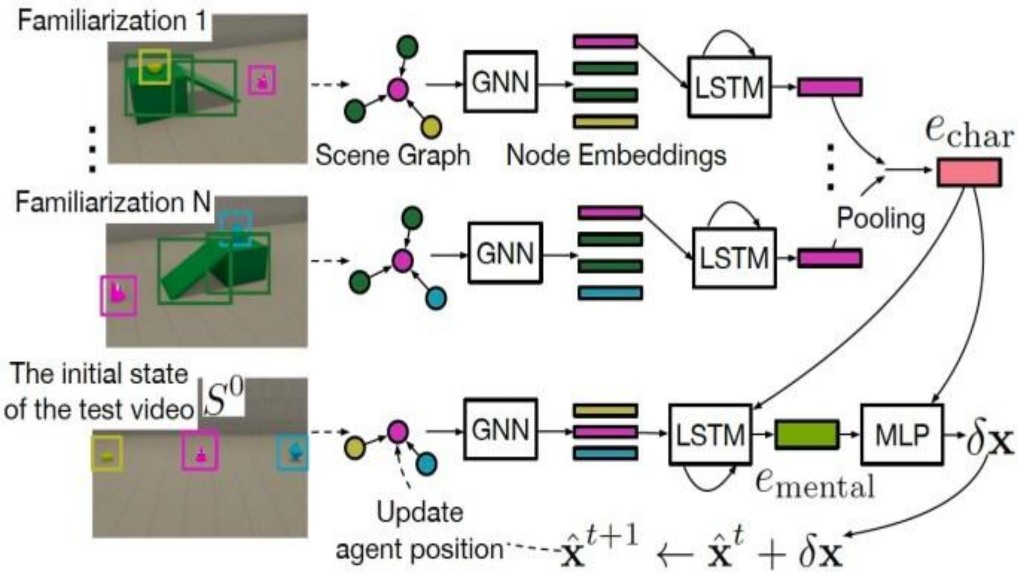

Fig. 3. Overview of ToMnet-G model

The ToMnet-G model uses graph neural networks and LSTMs to embed scene representations and predict agent behavior (Fig. 3). The contrast between the two models highlights the challenges of the simplest tasks that humans learn without any instructions.

In order for an ML model to work well, the following simple steps can be implemented:

1. Train a classifier on labeled data.
2. The bigger classifier model then infers pseudo-labels on a much larger unlabeled dataset.
3. Then, it trains a larger classifier on the combined labeled and pseudo-labeled data, while also adding noise.
4. (Optional) Going back to step 2, the smaller model may be used a new classifier.

One can view this as a form of self-training, because the model generates pseudo-labels with which it retrains itself to improve performance. One underpinning hypothesis is that the noise added during training not only helps with the learning, but also makes the model more robust. This approach is similar to knowledge distillation, which is a process of transferring knowledge from a large model to a smaller model. The goal of distillation is to improve speed in order to build a model that is fast to run in production without sacrificing much in quality compared to the bigger model.

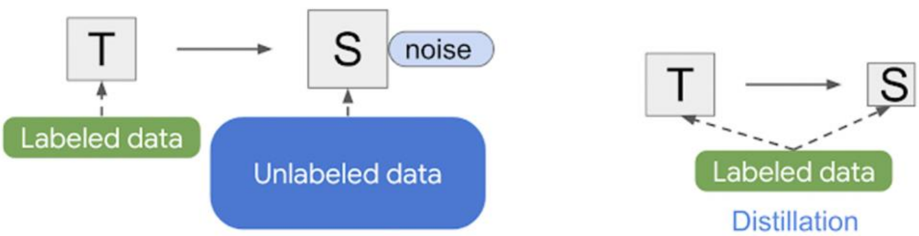

Fig. 4. Simple illustrations of the model and knowledge distillation.

Knowledge distillation does not add noise during training (e.g., data augmentation or model regularization) and typically involves a smaller inference model. In contrast, one can think of it as the process of “knowledge expansion”. One strategy for training production models is to apply training twice (Fig. 4):

- first to get a larger inference model T’ and then
- to derive a *smaller* model S.

In some cases, the training may need data augmentation, yet, in certain applications, e.g., natural language processing, such types of input noise are not readily available. For those applications, the training model can be simplified to have no noise. In that case, the above two-stage process becomes a simpler method:

- First, the bigger model infers pseudo-labels on the unlabeled dataset from which is a new model (T’) that is of *equal-or-larger* size than the original model being trained.
- The self-training phase is then followed by knowledge distillation to produce a smaller model for production.

3. Sources of Harm in ML

This section explores each potential source of harm in-depth. Each subsection will detail where and how in the ML pipeline problems might arise, as well as a characteristic example. These categories are not mutually exclusive; however, identifying and characterizing each one as distinct makes them less confusing and easier to tackle.

3.1. Historical Bias

Historical bias arises even if data is perfectly measured and sampled, if the world as it is or was leads to a model that produces harmful outcomes. Such a system, even if it reflects the world accurately, can still inflict harm on a population. Considerations of historical bias often involve evaluating the representational harm (such as reinforcing a stereotype) to a particular group.

3.2. Representation Bias

Representation bias occurs when the development sample under-represents some part of the population, and subsequently fails to generalize well for a subset of the use population. Representation bias can arise in several ways:

(1) When defining the target population, if it does not reflect the use population. Data that is representative of Boston, for example, may not be representative if used to analyze the population of Indianapolis.
(2) When defining the target population, if contains under-represented groups. Say the target population for a particular medical dataset is defined to be adults aged 18-40. There are minority groups within this population: for example, people who are pregnant may make up only 5% of the target population.
(3) When sampling from the target population, if the sampling method is limited or uneven. For example, the target population for modeling an infectious disease might be all adults, but medical data may be available only for the sample of people who were considered serious enough to bring in for further screening. As a result, the development sample will represent a skewed subset of the target population. In statistics, this is typically referred to as sampling bias.

3.3. Measurement Bias

Measurement bias occurs when choosing, collecting, or computing features and labels to use in a prediction problem. For example, “creditworthiness” is an abstract construct that is often operationalized with a measureable proxy like a credit score. Proxies become problematic when they are poor reflections or the target construct and/or are generated differently across groups, which can happen when:

(1) The proxy is an oversimplification of a more complex construct. Consider the prediction problem of deciding whether a student will be successful (e.g., in a college admissions context). Algorithm designers may resort to a single available label such as “GPA” [28], which ignores different indicators of success present in different parts of the population.
(2) The method of measurement varies across groups. For example, consider factory workers at several different locations who are monitored to count the number of errors that occur (i.e., observed number of errors is being used as a proxy for work quality). This can also lead to a feedback loop wherein the group is subject to further monitoring because of the apparent higher rate of mistakes [5, 17].
(3) The accuracy of measurement varies across groups. For example, in medical applications, “diagnosed with condition X” is often used as a proxy for “has condition X.” However, structural discrimination can lead to systematically higher rates of misdiagnosis or underdiagnosis in certain groups [23, 32, 35].

3.4. Aggregation Bias

A particular dataset might represent people or groups with different backgrounds, cultures or norms, and a given variable can mean something quite different across them. Aggregation bias can lead to a model that is not optimal for any group, or a model that is fit to the dominant population (e.g., if there is also representation bias).

3.5. Learning Bias

Learning bias arises when modeling choices amplify performance disparities across different examples in the data [24]. For example, an important modeling choice is the objective function that an ML algorithm learns to optimize during training. Typically, these functions encode some measure of accuracy on the task (e.g., cross-entropy loss for classification problems or mean squared error for regression problems).

3.6. Evaluation Bias

Evaluation bias occurs when the benchmark data used for a particular task does not represent the use population. Evaluation bias ultimately arises because of a desire to quantitatively compare models against each other. Such generalizations are often not statistically valid [38], and can lead to overfitting to a particular benchmark.

3.7. Deployment Bias

Deployment bias arises when there is a mismatch between the problem a model is intended to solve and the way in which it is actually used. This often occurs when a system is built and evaluated as if it were fully autonomous, while in reality, it operates in a complicated socio-technical system moderated by institutional structures and human decision-makers (Selbst et al. [39] refers to this as the “framing trap”).

4. A FRAMEWORK FOR DATA GENERATION AND ML PIPELINE

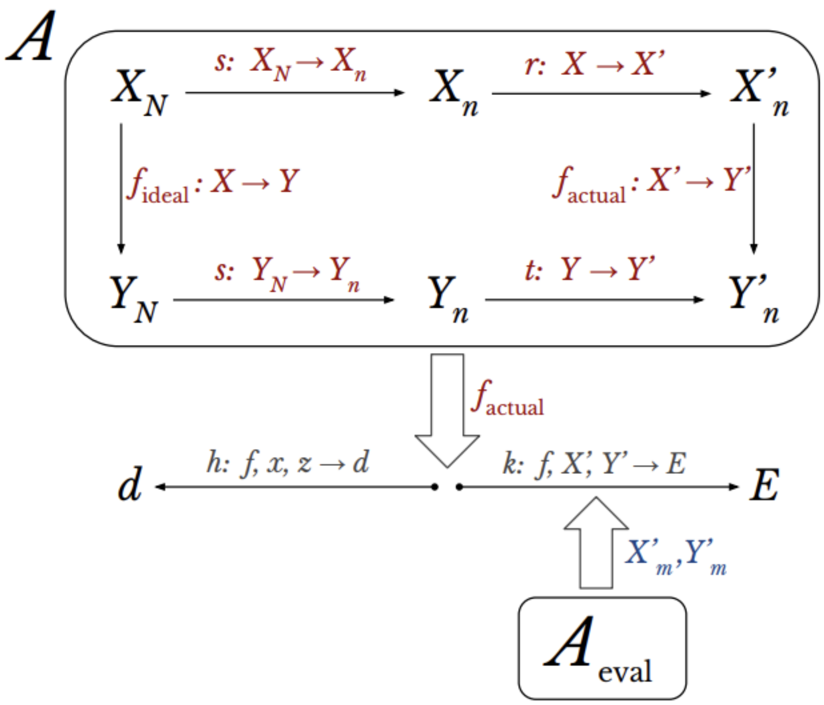

Fig. 5. A data generation and ML pipeline viewed as a series of mapping functions.

There is a growing body of work on "fairness-aware algorithms" that modify some part of the modelling pipeline to satisfy particular notions of "fairness." Consider the data transformations for a dataset as depicted in Figure 5. The upper part of the diagram of Figure 2 deals with data collection and model building, while the bottom half describes the evaluation and deployment process.

The data transformation sequence can be abstracted into a general process A. Let X and Y be the underlying feature and label constructs we wish to capture where $s : X_N \to X_n$ is the sampling function. X' and Y' are the measured feature and label proxies that are chosen to build a model, where r and t are the projections from constructs to proxies, i.e., $X \to X'$ and $Y \to Y'$.

The function $f\text{ideal} : X \to Y$ is the target function—learned using the ideal constructs from the target population—but $f\text{actual} : X' \to Y'$ is the actual function that is learned using proxies measured from the development sample. Then, the function k computes some evaluation metric(s) E for $f$actual on data X'_m, Y'_m (possibly generated by a different process, e.g., A_{eval} in Figure 2).

Finally, given the learned function $f$actual, a new input example x, and any external, environmental information z, a function h governs the real-world decision d that will be made (e.g., a human decision-maker taking a model's prediction and making a final decision).

5. MITIGATION TECHNIQUES

The aim of this section is to understand and motivate mitigation techniques in terms of their ability to target different sources of harm to get a better understanding when and why different approaches might help, and what hidden assumptions they make.

As an example, measurement bias is related to how features and labels are generated (i.e., how r and t are instantiated). Historical bias is defined by inherent problems with the distribution of X and/or Y across the entire population. Therefore, solutions that try to adjust s by collecting more data (that then undergoes the same transformation to X') will likely be ineffective for either of these issues. However, it may be possible to combat historical bias by designing s to systematically over- or under-sample X and Y, leading to a development sample with a different distribution that does not reflect the same undesirable historical biases. In the case of measurement bias, changing r and t through more thoughtful, context-aware measurement or annotation processes (e.g., as in Patton et al. [34]) may work.

In contrast, representation bias stems either from the target population definition (X_N , Y_N) or the sampling function (s). In this case, methods that adjust r or t (e.g., choosing different features or labels) or g (e.g., changing the objective function) may be misguided. Importantly, solutions that do address representation bias by adjusting s implicitly assume that r and t are acceptable and that therefore, improving s will mitigate the harm.

Learning bias is an issue with the way f is optimized, and mitigations should target the defined objective(s) and learning process [24]. In addition, some sources of harm are connected: e.g., learning bias can exacerbate performance disparities on under-represented groups, so changing s to more equally represent different groups/examples could also help prevent it.

Deployment bias arises when h introduces unexpected behaviour affecting the final decision d. Dealing with deployment bias is challenging since the function h is usually determined by complex real-world institutions or human decision-makers. Mitigating deployment bias might involve instituting a system of checks and balances in which users balance their faith in model predictions with other information and judgements [26]. This might be facilitated by choosing an f that is human-interpretable, or by developing interfaces that help users understand model uncertainty and how predictions should be used. Evaluation and aggregation bias are discussed in more detail below.

To supplement user reporting, platforms have algorithms that flag content for human review. Several platforms currently use image recognition tools and natural language processing classifiers to help moderators filter and prioritize possible objectionable content for evaluation.

Such prompts have at least three virtues. First, they may help users pause and engage in what Daniel Kahneman calls “system 2” thinking—higher-level cognitive reflection. In fact, research has shown that pop-up warnings requiring user interaction to dismiss them can positively change user behavior. Second, if such self-moderation occurs, it would be in advance of posting, before potentially harmful material can spread. Finally, these prompts preserve users’ freedom of expression, as they allow users to ignore the warnings and post the questionable material anyway.

Finally, there is a risk of exploitation by bad actors. Those who intentionally and willfully post misleading or dangerous material will not be deterred by an algorithmic warning. Instead, they could use the warnings to help them craft harmful posts that fall just below the threshold of algorithmic detection.

User prompts are designed to reduce the spread of harmful content while respecting freedom of expression and are immediate and reasonably effective. The precedent for user prompts already exists, and the technology needed to expand them into new contexts is available. All that remains is for platforms to take action.

6. RECOMMENDATIONS

Bringing convolutional neural networks (CNNs) to any industry through means of AI algorithms—whether it be medical imaging, robotics, or some other application entirely—has the potential to enable new functionalities and reduce the compute requirements for existing processes as a single CNN can replace more computationally expensive image processing, denoising, and object detection algorithms. However, there might be some challenges and difficulties while moving an idea from conception to productization. Here is an overview of some challenges and potential solutions regarding the development and deployment of AI model.

Leverage existing models

As existing models already exist for almost every application, rather than reinventing the wheel, it's often much easier to start with a network based on one of these architectures. Moreover, starting with a known model will reduce the amount of time, data, and effort to train a model, since it's possible to retrain existing models in a process called 'transfer learning.'

Simple models are effective

For most applications, there is no need for a latest and greatest in CNN architectures. For example, if an application only requires detecting the difference between a few different objects with high certainty, even simple detectors can do the task. Users can benefit greatly once they realize that their applications can be solved for a fraction of the computational complexity with much simpler models than what's on the forefront of research. The goal is to not make the migration to CNNs any harder than it has to be.

Integrate quantization early

Quantizing a model down from multi-byte precisions to a single-byte can multiply inference speed with little to no degradation in accuracy. For example, frameworks such as PyTorch expose their own methods for quantizing models, but they're not always compatible with each other. Regardless of the approach taken, the aim should be to quantize from the outset of developing the model in a consistent way.

7. CONCLUSION

This paper provides a framework for understanding the sources of downstream harm caused by ML systems to facilitate productive communication around potential issues. By framing sources of downstream harm through the data generation, model building, evaluation, and deployment processes, we encourage application-appropriate solutions rather than relying on broad notions of what is fair. Fairness is not one-size-fits-all; knowledge of an application and engagement with its stakeholders should inform the identification of these sources.

Finally, the paper illustrates that there are important choices being made throughout the broader data generation and ML pipeline that extend far beyond just model training. In practice, ML is an iterative process with a long and complicated feedback loop. This paper highlighted problems that manifest through this loop, from historical context to the process of benchmarking models to their final integration into real-world processes.

REFERENCES

[1] Agre, P. E. (1994). Surveillance and capture: Two models of privacy. The Information Society, 10(2), 101–127.
[2] Allen, J. (2016). Topologies of power. Beyond territory and networks. Routledge.
[3] Bratton, B. (2015). The Stack: On software and sovereignty. MIT Press.
[4] Bucher, T. (2018). If...then: Algorithmic power and politics. Oxford University Press.
[5] Castañeda, L., & Selwyn, N. (2018). More than tools? Making sense of the ongoing digitizations of higher education. International Journal of Educational Technology in Higher Education, 15(1).
[6] Decuypere, M. (2019a). Open Education platforms: Theoretical ideas, digital operations and the figure of the open learner. European Educational Research Journal, 18(4), 439–460.
[7] Decuypere, M. (2019b). Researching educational apps: ecologies, technologies, subjectivities and learning regimes. Learning, Media and Technology, 44(4), 414–429.
[8] Decuypere, M. (2019c). STS in/as education: where do we stand and what is there (still) to gain? Some outlines for a future research agenda. Discourse: Studies in the Cultural Politics of Education, 40(1), 136–145
[9] Dieter, M., Gerlitz, C., Helmond, A., Tkacz, N., Vlist, F., Der, V., & Weltevrede, E. (2018). Store, interface, package, connection : Methods and propositions for multi-situated app studies. CRC Media of Cooperation Working Paper Series No 4.
[10] Drucker, J. (2020). Visualization and Interpretation: Humanistic Approaches to Display. MIT Press. Journal of New Approaches in Educational Research, 10(1)
[11] Mathias, Decuypere The Topologies of Data Practices: A Methodological Introduction Fedorova, K. (2020). Tactics of Interfacing. Encoding Affect in Art and Technology. MIT Press. Goriunova, O. (2019). The Digital Subject: People as Data as Persons. Theory, Culture & Society, 36(6), 125–145.
[12] & Ruppert, E. (2020). Population Geometries of Europe: The Topologies of Data Cubes and Grids. Science, Technology, & Human Values, 45(2), 235–261.
[13] Gulson, K. N., Lewis, S., Lingard, B., Lubienski, C., Takayama, K., & Webb, P. T. (2017). Policy mobilities and methodology: a proposition for inventive methods in education policy studies. Critical Studies in Education, 58(2), 224–241.
[14] Gulson, K. N., & Sellar, S. (2019). Emerging data infrastructures and the new topologies of education policy. Environment and Planning D: Society and Space, 37, 350–366.
[15] Hartong, S. (2020). The power of relation-making: insights into the production and operation of digital school performance platforms in the US. Critical Studies in Education, 00(00), 1–16.
[16] Hartong, S., & Förschler, A. (2019). Opening the black box of data-based school monitoring: Data infrastructures, flows and practices in state education agencies. Big Data & Society, 6(1),
[17] Lash, S. (2012). Deforming the Figure: Topology and the Social Imaginary. Theory, Culture & Society, 29(4-5), 261–287.
[18] Latour, B. (1986). Visualization and cognition: Thinking with eyes and hands. Knowledge & Society, 6, 1–40. Retrieved from http://hci.ucsd.edu/10/readings/Latour(1986).pdf
[19] Law, J. (2004). After Method: Mess in Social Science Research. Psychology Press.
[20] Lewis, S. (2020). Providing a platform for "what works": Platform-based governance and the reshaping of teacher learning through the OECD's PISA4U. Comparative Education, 56(4).
[21] Lewis, S., & Hardy, I. (2017). Tracking the Topological: The Effects of Standardised Data Upon Teachers' Practice. British Journal of Educational Studies, 65(2), 219–238.
[22] Light, B., Burgess, J., & Duguay, S. (2018). The walkthrough method: An approach to the study of apps. New Media and Society, 20(3), 881–900.
[23] Lindh, M., & Nolin, J. (2016). Information We Collect: Surveillance and Privacy in the Implementation of Google Apps for Education. European Educational Research Journal, 15(6),
Lury, C., & Day, S. (2019). Algorithmic Personalization as a Mode of Individuation. Theory, Culture & Society, 36(2), 17–37.

[24] Mathias, Decuypere The Topologies of Data Practices: A Methodological Introduction Lury, C., Fensham, R., Heller-Nicholas, A., & Lammes, S. (2018). Routledge Handbook of Interdisciplinary Research Methods. Routledge.

[25] Lury, C., Parisi, L., & Terranova, T. (2012). Introduction: The Becoming Topological of Culture. Theory, Culture & Society, 29(4-5), 3–35.

[26] Lury, C., Tironi, M., & Bernasconi, R. (2020). The Social Life of Methods as Epistemic Objects: Interview with Celia Lury. Diseña, 16, 32–55.

[27] Lury, C., & Wakeford, N. (2012). Introduction: A perpetual inventory. Inventive Methods (pp. 15–38). Routledge.

[28] Martin, L., & Secor, A. J. (2014). Towards a post-mathematical topology. Progress in Human Geography, 38(3), 420–438.

[29] Piattoeva, N., & Saari, A. (2020). Rubbing against data infrastructure(s): methodological explorations on working with(in) the impossibility of exteriority. Journal of Education Policy, 00(00), 1–21.

[30] Plantin, J. C., Lagoze, C., Edwards, P. N., & Sandvig, C. (2018). Infrastructure studies meet platform studies in the age of Google and Facebook. New Media and Society, 20(1), 293–310.

[31] Prince, R. (2017). Local or global policy? Thinking about policy mobility with assemblage and topology. Area, 49(3), 335–341.

[32] Ratner, H. (2019). Topologies of Organization: Space in Continuous Deformation. Organization Studies, 1–18.

[33] Ratner, H., & Gad, C. (2019). Data warehousing organization: Infrastructural experimentation with educational governance. Organization, 26(4), 537–552.

[34] Ratner, H., & Ruppert, E. (2019). Producing and projecting data: Aesthetic practices of government data portals. Big Data & Society, 6(2), 1–16.

[35] Ruppert, E., Law, J., & Savage, M. (2013). Reassembling Social Science Methods: The Challenge of Digital Devices. Theory, Culture & Society, 30(4), 22–46.

[36] Suchman, L. (2012). Configuration. In C. Lury & N. Wakeford (Eds.), Inventive Methods: The Happening of the Social (pp. 48–60). Taylor and Francis.

[37] Thompson, G., & Cook, I. (2015). Becoming-topologies of education: deformations, networks and the database effect. Discourse: Studies in the Cultural Politics of Education, 36(5), 732–748.

[38] Thompson, G., & Sellar, S. (2018). Datafication, testing events and the outside of thought. Learning, Media and Technology, 43(2), 139–151.

[39] van de Oudeweetering, K., & Decuypere, M. (2019). Understanding openness through (in)visible platform boundaries: a topological study on MOOCs as multiplexes of spaces and times. International Journal of Educational Technology in Higher Education, 16(1).

[40] van de Oudeweetering, K., & Decuypere, M. (2020). In between hyperboles: forms and formations in Open Education. Learning, Media and Technology, Advance online publication, 1–18.

[41] Williamson, B. (2017). Learning in the “platform society”: Disassembling an educational data assemblage. Research in Education, 98(1), 59–82.

AUTHORS

Ayse received her MSc in Internet Studies in University of Oxford in 2006. She participated in various research projects for UN, Nato and the EU regarding HCI (human-computer interaction). She completed her doctorate degree in user experience design in Oxford while working as an adjunct faculty member at Bogazici University in her home town Istanbul. Ayse has also a degree in Tech Policy from Cambridge University. Currently, Ayse lives in Silicon Valley where she works as a visiting scholar for Google on human-computer interaction design.