The Machine Learning Column

BY

Pablo Barceló

Pontificia Universidad Católica de Chile
Avda. Vicuña Mackenna 4860
Santiago, Chile
pbarcelo@uc.cl

AND

David Saulpic

Institut de Recherche en Informatique Fondamentale (IRIF)
CNRS & Université Paris Cité
8 Place Aurélie Nemour
75013 Paris, France
david.saulpic@irif.fr

THE LANDSCAPE OF TCS FOR ML

Abstract

We introduce here a new column, TCS for ML. The idealistic goal of this column will be to examine what special directions, what special insight Theoretical Computer Science can bring to Machine Learning. In this initial edition, we attempt to examine how our TCS fields currently contribute to ML, from all side. This introduction will serve as the basis for deeper investigation in next columns, where we will try to highlight new theoretical questions arising from ML and its massive use.

1 Introduction

The past 10 years have seen the rise of Machine Learning (ML), which is now a recognized and dominant field - in terms of funding, public exposure, research progress, and applications. As many aspects of ML are directly related to computation, it would be natural that Theoretical Computer Science (TCS) contributes to this success-story. However, this is a relatively new field, growing extremely fast: it is not completely structured, and it is not easy to enter the field, nor to follow the different contributions. This is perhaps why editors of the *Bulletin* asked us to write a column on "TCS for ML".

As both TCS and ML are vast, fast-moving fields, we do not have a definitive picture of what this means exactly. Hence, we have asked -— more or less successfully —- experts from various TCS subfields to provide their own hindsight, in order to sketch an initial big picture. In this initial column, we present those expert perspectives along with our own views, and in the coming months we will dive into more specific aspects of TCS and ML, providing the *Bulletin* with interviews of researchers discussing their work. Our far-reaching objective will be to provide a global picture of the research directions followed by the TCS community to contribute to ML ; and, ideally, to identify open and interesting directions.

As a disclaimer, while we aim to capture a broad and honest picture of the landscape, we are aware that our perspective is limited and shaped by our own interests. We therefore welcome any comments or criticisms that could help improve this review.

TCS and ML: definitions. We considered as TCS the various communities represented by EATCS and showcased at ICALP. This spans a wide spectrum of areas—sometimes overlapping, sometimes far apart—but all connected by SIGACT's definition of TCS as "the formal analysis of efficient computation and computational processes".[1] Guided by this perspective, we focus here on works that present strong theoretical results or apply theoretical tools to rigorously analyze the capabilities and limitations of ML models.

ML, as we know, is an expansive discipline at the intersection of computer science, statistics, and numerous applied fields, dedicated to designing algorithms and representations that can infer patterns or decision rules from data. Beyond its practical successes, ML is a source of theoretical challenges for TCS. First, it calls to revisit fundamental questions about computation, efficiency, and representation. Second, it offers opportunities for interaction with traditional TCS communities, such as algorithms, logic, formal verification, databases, and knowledge representation, where decades of research provide principled ways to encode knowledge, enforce constraints, and explain or verify model behavior.

In this column we have organized our review by traditional TCS subfields (Theory of Computation, Ethical ML, Algorithms, Logic/ Formal Languages/ Discrete Structures, Foundations of Data Management, Formal Verification, and Knowledge Representation), highlighting foundational works and current research directions relating to ML in each area. But before diving into those subfields, it is useful to step back and group the interactions between TCS and ML more conceptually, following the SIGACT definition above. We identify three complementary roles:

- Formal analysis of ML models and their capabilities—understanding what learning architectures (e.g., Transformers, GNNs) can and cannot compute, their sample complexity, and their expressiveness.
- Formal analysis of ML outcomes — using TCS tools to verify, explain, and guarantee properties such as fairness, privacy, or logical consistency of predictions.
- Support for efficient computation—drawing on TCS techniques in algorithms, data management, and KR to scale ML pipelines, optimize primitives, and integrate explicit knowledge with data-driven learning.

These perspectives are not exhaustive, but they capture recurring themes across the subfields we survey. They also remind us that while ML often pushes TCS to confront new challenges, TCS offers a deep well of methods for making ML more principled, reliable, and interpretable.

[1] `https://www.sigact.org/`

Understanding the power and limits of neural networks. One of the central themes of TCS is understanding computation—specifically, what can be computed and with which model of computation. These questions can be directly reformulated in the context of ML: what can be learned, and what can a specific model learn? From the beginning of the field, several research areas have emerged to provide different perspectives on computation. As we will see, they all offer viewpoints that can explain—or at least begin to explain—the power and limits of neural networks.

First, Learning Theory emerged alongside ML precisely to build a theory of learnability. Here, "learning" means that an algorithm is given some labeled data as input and must infer the labels of new data points. The labels are functions of the data, and the goal is to understand which functions are "easy" or "hard" to learn. Difficulty is measured primarily in terms of the amount of input data necessary for inference or the computational complexity required. This general theory therefore aims to broadly characterize which functions can be learned, defining properties of functions (or dimensions) that measure the difficulty of learning them. One specific learning task that has recently been investigated—shedding TCS light on a broader debate—is the task of learning languages, which we will cover in more detail later.

Logic and formal languages provide a different view: instead of studying functions broadly, their tools and theory allow us to focus on particular network architectures. A fixed architecture, or a family of architectures, restricts the computations possible for a network; these fields examine the properties that can be recognized by specific architectures. A similar perspective comes from complexity theory, which focuses on specific learning tasks and derives lower bounds on the size of a network required to compute those tasks. Recent examples highlight the difficulty for LLMs to "compose" functions—a task they are neither explicitly designed nor trained for, but which they can sometimes handle surprisingly well and, at other times, fail at quite spectacularly.

In both cases, substantial modeling effort is required before answering such questions: one must model which class of functions networks attempt to learn or provide a formal model for the network architecture.

Formal analysis of ML's results. One of the major critiques of ML systems concerns the uncertain nature of their outcomes: given a new data point—different from those observed during the learning process—how can we be sure that the model will remain satisfactory? Here, formal guarantees are required regarding the model's ability to generalize. This is where TCS steps in, as proving guarantees is our daily bread.

These guarantees can sometimes be automatically established via formal veri-

fication: given a model, one may wish to automatically verify that its output satisfies certain desirable properties. Hardness results — even for simple models and properties — make this task quite challenging, which is why several relaxations have been investigated.

Another approach is to prove such guarantees manually, by establishing theorems about specific properties of the models. Beyond the correctness of predictions—which is the focus of Learning Theory—several ethical properties are being actively studied. A growing field of ethical AI and ML formally defines these properties and designs learning algorithms that enforce them, so that the resulting model can be called "fair" or "private," for instance. Here again, modeling plays a central role: mathematically defining "fairness" is not an easy task. Many definitions exist, are not necessarily compatible with each other, and may not provide practical leverage for algorithm design. As we will see in the dedicated section, this modeling can be guided by TCS insights.

Design of Efficient ML Algorithms. The aspect of TCS perhaps closest to practical applications is the design of efficient algorithms and database techniques that enable fast and memory-efficient computation. Since ML is inherently tied to big data, it requires algorithms and data-management tools that can scale: our field provides such tools, backed by solid theory and formal guarantees.

The communication between TCS and ML goes both ways. Ideas from TCS have been incorporated into big-data pipelines — such as streaming and sketching techniques in algorithms or in-database learning frameworks and functional aggregate queries (FAQs) from database theory, which allow training models directly within relational systems without expensive data extraction. Conversely, the ML industry continues to raise new computational questions: identifying and optimizing the critical tools or subroutines used in practice. For example, efficient gradient computation over relational data, fast query-based feature generation, and probabilistic database techniques for uncertainty handling are all areas where database theory informs ML workflows.

Finding which subroutines are ripe for algorithmic improvement is itself an art—one that requires expertise in both ML and TCS. This ongoing exchange not only optimizes existing pipelines but also opens opportunities for theoretical insights to drive entirely new approaches to scalable ML.

We will now dive, subfield by subfield, into more details. Again, we are not expert in all of those: we have tried to ask better-placed persons their opinion, and have probably overlooked some directions and over-emphasized some others. Nonetheless, we still believe those partial thoughts are an interesting first step, and will warmly welcome any feedback.

2 Theory of Computation

The interplay between theory of computation and ML has deep roots, from VC theory in statistics to PAC learning in theoretical computer science. Today, rapid advances in ML far outpace rigorous theoretical understanding, creating both a challenge and an opportunity. Theory is needed to explain the successes and limitations of current methods and to distinguish fundamental phenomena from transient trends. While ML theory can be messy—often involving toy models or impractical algorithms, it is precisely this principled, analytical perspective that ensures AI's development is grounded in solid foundations.

- **Learning Theory:** Learning theory is a well-established field whose primary goal is to understand which functions are *learnable* and which are not. Several definitions of learnability co-exist, most prominently PAC learning [113], and efficiency may be measured in terms of the number of samples or computational time. In other words, this field studies the theory of supervised machine learning and is therefore at the heart of our topic.

  To analyze learnability, no assumptions are made about the structure of the learning or prediction algorithms, so the results apply very broadly. The trade-off is that there are limitations on the types of problems studied. For instance, PAC learning traditionally assumes that input examples follow a distribution and that the test set *also* follows this distribution. In addition, positive results often require strong assumptions on this distribution, e.g., that it is Gaussian or a product distribution. Distribution-free learning is generally hard (see, for instance, [71, 30, 72]), and some recent works attempt to lift results from specific distributions to general ones [28, 29]. To address limitations regarding distributions, a theory of robustness has emerged (see, e.g., [88, 50, 80, 93]) to characterize what sorts of noise or adversarial conditions can be handled.

  Beyond fundamentally understanding what is learnable, concepts from learning theory are also used to evaluate algorithms in practice. For example, [54, 36] show that diffusion models can learn mixtures of Gaussians, extending a recent line of work on these models, and [116] evaluate the ability of transformers to learn k-fold composition functions.

  Recent work has studied "hallucinations" in LLMs [66, 65]. These results show that even an ideal, perfectly calibrated language model cannot entirely avoid hallucinations: it must assign nonzero probability to facts not seen during training, leading inevitably to some false outputs. Conceptually, these findings echo *No Free Lunch* theorems, as they highlight fundamental trade-offs between generality and accuracy.

- **Language generation in the limit:** The success of Large Language Models (LLMs) has intensified the need to understand the principles underlying their effectiveness and to identify the mathematical components that support it. Recent research has begun addressing these questions by studying the design of generative algorithms for language [78]. This line of work builds on a rich set of theoretical tools, most notably the *Gold–Angluin framework* for language identification, originally developed to analyze the learnability of formal languages. More formally speaking, imagine an adversary who keeps listing strings from some unknown language L, which we only know belongs to a possibly infinite list of candidate languages. A computational agent tries to learn how to generate strings from L. We say the agent *generates from L in the limit* if, after seeing enough of the adversary's examples, it can start producing new strings that (1) always belong to L, and (2) have never been shown before. The main result in this area shows that there exists such an agent for any countable list of possible languages. Subsequent work has examined the following key question: is it possible to design language generation algorithms that are not only *valid*—producing well-formed strings in the target language L—but also *broad*—producing a diverse set of strings from L [68, 100, 77, 67]. LLMs inherently face this validity–breadth trade-off: they must balance the need for accuracy and coherence (avoiding *hallucination*) with the need for variety and coverage (avoiding *mode collapse*).

- **Complexity, and Lower bounds for ML architectures:** Transformers are the ML architecture that lies at the core of LLMs. Transformers use a mechanism known as *self-attention* to weigh the importance of different parts of an input sequence. They also use *chain-of-thought* (CoT) reasoning as a way to break down a complex problem into a series of intermediate steps which the model explicitly generates before arriving at a final answer. By providing these intermediate steps, CoT effectively gives the Transformer model a form of "scratchpad memory" or "workspace" to perform computations.

  A key question in understanding the computational capabilities of Transformers is how many CoT steps they require to carry out function composition, a core operation in both symbolic reasoning and natural language understanding. Recent theoretical work has tackled this from complementary perspectives, depending on whether one considers *hard* or *soft* attention mechanisms. Hard attention makes a discrete, non-differentiable choice, focusing on a small, selected subset of input elements. In contrast, soft attention computes a weighted average of all elements in the input sequence. The first approach uses communication complexity to prove that single-

layer Transformers with soft attention cannot perform function composition when the function domains are large, and extend this to show that iterated composition requires a number of CoT steps that grows with the domain size [101]. The second approach uses the notion of *Ehrenfeucht-Haussler rank* of a Boolean function and the minimum number of Chain of Thought (CoT) steps required by a single-layer Transformer with hard attention to compute it, demonstrating that l-fold function composition requires exactly l CoT steps [18]. The third approach provides the first unconditional lower bounds for multi-layer Transformers with soft attention. This work introduces a new *multi-party autoregressive* communication model and uses it to obtain strong lower bounds for the number of CoT steps required to compute the composition of L functions [35].

3 Ethical ML

Another direction in which a principled and analytical perspective is required is that of ethical ML, in which we include urgent issues of AI safety, ethics, and trustworthiness. For these, formal guarantees are required—and sometimes even enforced by law. Theoretical insights not only clarify what is possible in these directions but may also help in modeling and proving these formal guarantees. The wonderful book by Kearns and Roth [70] dives deep into the topic; we will only briefly cover some of its aspects related to privacy and fairness.

In this field, TCS provides technical tools and analyses, but a technical solution is not enough. While this is beyond our survey, we refer, for instance, to the works by the FAccT community (organized around the ACM conference on Fairness, Accountability and Transparency) or STS scholars (e.g., [92] and the journal Big Data and Society), which are key to studying ethical questions. We emphasize that, in our view, TCS only provides a helping hand, not a complete answer.

- **Privacy:** Certain ML applications rely on sensitive training data, which must be carefully protected during training. This protection is required by the European GDPR; privacy concerns are therefore brought into legal debates, in which TCS arguments are used to argue for the privacy of some algorithms. Over the past decade, differential privacy has emerged as a foundational tool to address this challenge. A major line of research focuses on designing differentially private algorithms for empirical risk minimization (ERM), particularly when the loss function is Lipschitz or strongly convex [23]. This work introduced several algorithms and established tight theoretical error bounds, providing optimal risk guarantees. Building on these ideas, differential privacy has been extended to deep learning. Differentially Private Stochastic Gradient Descent (DP-SGD) offers a principled

framework for algorithm design while carefully controlling privacy budgets [1], whereas Private Aggregation of Teacher Ensembles (PATE) leverages the teacher-student paradigm to achieve strong utility alongside differential privacy guarantees [98]. More recent research has focused on efficient algorithms for differential learning and on handling non-convex loss functions [21, 22]. Other work investigates how to add just enough noise to preserve privacy without compromising accuracy, introducing mechanisms that carefully balance privacy and utility [55]. On the other hand, what differential privacy actually means for practical privacy is somewhat unclear—in particular because of parameter choices [48]—and some works focus on attacks in order to better understand the limits of this model for privacy [14].

- **Fairness:** The dramatic analysis of the COMPAS system by the journalists of ProPublica [64] has shown that ML predictions are prone to massive bias—sometimes replicating bias from the training data, sometimes inherent to the task. In order to measure this bias, several notions of fairness coexist, with two main categories: individual fairness tries to measure how much predictions differ for individuals that are similar [46], while group fairness measures whether the predictions are fair between different population groups [20]. While the Statistics community is designing fair procedures—i.e., that have small bias according to one of these measures—for problems such as regression or classification, with the objective of bounding risk or learning rate under some statistical assumption, TCS naturally contributed to designing fair and efficient algorithms and to designing a toolbox for fair algorithms [38, 47]; but also to understanding the relations between different fairness notions and to formalizing impossibility results [79, 105, 52].

Here, our survey has some shortcomings: the story for Ethical AI is not limited to these two topics, but we did not manage to cover more. In particular, it seems there are exciting developments around the notion of trustworthy ML, that we would have liked to be able to discuss.

We note that, in all cases, the solutions proposed by computer scientists are not sufficient as standalone tools. Quantifying privacy and fairness is not easy, and perhaps not even possible; and the number of different fairness notions may blur the debate, as mentioned on Wikipedia: "the different and sometimes competing notions of fairness left little room for clarity on when one notion of fairness may be preferable to another" [117]. However, these tools may be helpful when used in combination with other sociotechnical approaches.

4 Algorithms

One of the goals of algorithm design is to invent algorithms that make better use of memory and time. Quite naturally, some of the algorithmic techniques developed in the past now find applications in ML and diffuse ideas into that domain; conversely, new problems arising in ML are extensively studied and optimized by the algorithmic community.

- **Algorithms for data-intensive tasks:** Prominent examples of past algorithms finding application today are related to the "big data" side of ML, with huge memory and efficiency constraints. To address this, the study of metric embeddings (e.g., embedding into structured trees [49]) and dimensionality reduction (so-called Johnson-Lindenstrauss [83] or Locality-Sensitive Hashing [8]) helps simplify the input space and reduce the impact of the curse of dimensionality. The streaming model, with its emphasis on memory, led to the invention of sketches that allow data compression while still enabling computation of certain statistics or information [6]. Other extensively studied problems include basic unsupervised learning algorithms, namely clustering and regression. These have been studied through different formulations (e.g., metric-based with k-means [11, 40] or graph-based with sparsest cut [10]) and algorithmic settings, with emphasis on time efficiency [45] or memory efficiency [41, 42].

- **Computation of ML primitives:** More recently, part of the community has focused on efficiently computing primitives often used in contemporary ML. Two examples are optimal transport and its variants, investigated through an approximation-algorithm perspective [27], computational geometry tools [3], or with an emphasis on linear-time complexity [13]; and kernel-density estimation, with memory-efficient [104] or fast algorithms [34, 33]. Finding other problems relevant to ML and data analysis where the algorithmic toolbox may yield substantial improvements is not easy, as it requires expertise in both domains. Nevertheless, it remains one of the major questions for the subfield of algorithms that focuses on improving practical algorithms.

5 Logic, Formal Languages, and Discrete Structures

Studying the capabilities and limitations of ML architectures involves examining the properties they can express and the structures they can distinguish. This line of research is deeply rooted in logic and formal languages, which provides a framework for analyzing the properties of inputs that an architecture can represent. It

also draws on discrete algorithms as a tool for understanding how these architectures differentiate between various inputs. Over the past decade, a surge of studies has focused on understanding the capabilities of two fundamental ML architectures: *Transformers*, which we have seen before, and *Graph Neural Networks* (GNNs), which learn by propagating information across the edges of a graph.

- **Transformers:** The ability of Transformers to process sequences can be formally analyzed by drawing on the rich traditions of logic and formal language theory. Since Transformer inputs can be viewed as strings over a finite alphabet, researchers can investigate which classes of languages these models are capable of recognizing. The resulting characterization, however, critically depends on the architectural features under consideration. The languages recognized by Transformers with hard attention have been shown to be closely tied to those expressible in First-Order Logic (FO) and its extensions with counting [17, 120]. In turn, while equipped with certain expressive features, soft attention Transformers can be shown to recognize all FO-definable languages, and even extensions [119, 121]. However, when restricting the model to more practical architectures used in real-world applications, their ability to recognize the entire class of FO-definable languages becomes limited [84].

  Interesting results have also been obtained regarding the expressive limitations of certain Transformer models. For instance, Transformers equipped with a restricted form of hard attention, known as *unique* hard attention, are limited to recognizing languages within the complexity class AC^0 [61]. This limitation has significant consequences. Specifically, models with unique hard attention are provably incapable of recognizing languages that fall outside this class. A prime example is the Parity language, which determines whether a binary string has an even or odd number of ones. Because the Parity language is not in AC^0, these restricted Transformers are unable to solve this seemingly simple task [60].

  Yet another line of work has analyzed the ability to recognize languages for Transformers extended with CoT reasoning. This capability has been shown to boost the model's expressive power; in fact, several models of Transformers with CoT and hard attention have been shown to be Turing-complete, meaning that they are capable of recognizing all decidable languages [102, 90, 51].

  An excellent survey on different aspects of this topic has recently been published [109].

- **Graph Neural Networks:** The foundational work in understanding the expressive power of Graph Neural Networks (GNNs) began with the semi-

nal result that their ability to distinguish graphs coincides with that of the Weisfeiler-Leman (WL) test [118, 96]. The WL test is a widely studied polynomial-time heuristic for checking graph isomorphism. This initial result opened a large avenue of research exploring the relationship between different flavors of GNNs and various higher-order versions of the WL test. The WL connection has also allowed developing a clear understanding of which substructures of graphs can be detected, or counted, with GNNs [37, 15, 82]. Finally, the WL test has also been related to the VC dimension of GNNs, thus providing a critical benchmark for graph representation learning [95].

A parallel line of work has focused on understanding which properties of graphs can be recognized or defined by GNNs. Early results in this area provided a logical characterization of the class of FO definable graph properties that can be captured by GNNs, which coincides with the expressive power of *graded modal logic* (GML) [16]. This characterization requires GNNs to be equipped with a specific class of piecewise linear activation functions, which are expressive enough to simulate counting up to fixed bounds. On the other hand, it has been shown that if the activation functions are restricted to be linear or polynomial, the expressive power of GNNs drops significantly, and they can no longer capture even basic GML-definable properties [73]. More recently, the full expressive power of GNNs—beyond FO-definable properties—has been characterized for a restricted but expressive class of activation functions (known as *eventually constant* functions) in terms of an extension of GML enriched with Presburger arithmetic constraints [26]. In contrast, when activation functions are not eventually constant, the landscape becomes less well understood. In this more general setting, only upper and lower bounds are known for the expressive power of GNNs in logical terms [57]. Recently, the expressive power of recurrent GNNs—that is, GNN architectures where the same message-passing layer is applied repeatedly over multiple rounds—has also begun to be systematically investigated [4, 103, 32].

Several of these topics are presented in depth in a recent survey [56].

6 Foundations of Data Management

Database theory offers rigorous frameworks for organizing, querying, and managing large-scale structured data, which forms the backbone of many ML applications. By integrating principles from database theory, ML systems can achieve more efficient data access, better feature extraction, and more interpretable models

grounded in logical formalisms. This synergy enhances the scalability and reliability of ML algorithms and opens new avenues for understanding model behavior through formal queries. Below, we highlight several key ideas and approaches proposed by the database theory community that contribute to advancing these goals.

- **In-database learning:** A framework that enables the training of a wide array of statistical models directly inside a relational database has been proposed [74]. This is achieved by representing features as sparse tensors and expressing gradient computations as *functional aggregate queries* [75], a powerful and general framework for expressing a wide range of database queries that involve aggregation. The proposed method leverages the inherent join structure and functional dependencies of the database to significantly reduce per-iteration costs to a sub-linear level. By eliminating the need to extract data, the system achieves substantial speedups compared to conventional ML workflows, highlighting the power of applying database-theoretic optimization to modern ML pipelines.

- **Logic-based feature generation:** This line of research investigates how data management techniques can be exploited to learn sophisticated ML features specified by logical queries. A concrete example is the use of the well-known class of *unions of conjunctive queries* (UCQs)—which coincides with the existential-positive fragment of first-order logic—to separate classification data via a linear model whose features are given by such UCQs [76]. Learning such query-defined features is particularly important in settings with relational or graph-structured data, where predictive signals often stem from patterns spanning multiple entities and relationships. Their formal semantics also make them interpretable and amenable to theoretical analysis of expressiveness and generalization. This setting has inspired renewed interest in the long-studied problem of learning database queries from examples, leading to recent advances in both theoretical frameworks and algorithmic techniques [111, 112].

- **Querying ML models:** One of the central concerns in the foundations of data management is how to efficiently and effectively extract answers to logical queries over a dataset. A notable development in recent years is the realization that ML models can themselves be viewed as datasets—albeit implicit and compact ones—and thus can be queried to extract meaningful information about their input-output behavior [9, 58]. For example, querying an ML model can help generate explanations for its predictions, or verify whether it satisfies certain desirable properties. The computational com-

plexity of extracting such explanations over different classes of ML models has become a recurring topic in recent literature [19, 44, 97].

- **Dealing with uncertain data:** Modern ML models are powered by vast amounts of noisy data, which means that the information extracted from them often comes with varying degrees of confidence. A natural question, then, is how to measure such confidence in a principled way. Database theory has developed a foundational framework for addressing this challenge through the notion of *probabilistic data* [110]. At its core, a probabilistic database assigns a probability to each tuple, representing the likelihood that the tuple is present. These probabilities can be subject to constraints or correlations, enabling the modeling of rich and complex uncertainty scenarios. A central computational task in this setting is to determine the probability that a given query is true across all possible "worlds" encoded by the probabilistic data—an answer that directly quantifies our confidence in the query's validity given the noisy dataset. This framework has led to major theoretical advances, most notably a landmark dichotomy theorem for unions of conjunctive queries (UCQs): some UCQs can be evaluated in polynomial time with respect to data complexity, while others are #P-hard, revealing a sharp boundary between tractable and intractable cases [43]. Query evaluation over probabilistic databases is closely linked to another central problem in AI: knowledge compilation, which studies how to efficiently perform reasoning tasks—such as satisfiability—over propositional knowledge bases. This field has produced an impressive body of results, particularly in the design and use of classes of tractable Boolean circuits that enable efficient solutions to problems arising in the management and analysis of probabilistic data (see [7] for a survey on this topic).

7 Formal Verification

The theory of verification—concerned with rigorously proving that systems behave as intended—offers powerful tools for increasing the reliability and trustworthiness of ML models. As ML systems are deployed in safety-critical domains such as healthcare, transportation, and finance, ensuring their correctness, robustness, and fairness becomes essential. Verification theory contributes formal methods to specify desired properties of models, systematically check them against all possible inputs, and identify counterexamples when guarantees fail. This integration not only strengthens the safety and accountability of ML but also deepens our theoretical understanding of its behavior, enabling the design of models that are both powerful and provably reliable. Next, we highlight several key

ideas that have emerged at the intersection of formal verification and ML over the past decade.

- **Classical verification:** The verification of ML models has mostly centered around (deep) neural networks [5]. Languages for specifying desirable properties of the input–output behavior of neural networks are often based on logical formalisms such as Hoare logic or linear real arithmetic, allowing one to express statements of the form: whenever an input x to a network N satisfies certain constraints, the corresponding output $y = N(x)$ also satisfies certain constraints. A closely related *reachability* problem asks whether there exists an input in a given set that leads to an output in another given set. From a complexity-theoretic standpoint, exact verification of even simple properties is computationally intractable in the worst case. For feed-forward ReLU networks, the reachability problem is NP-complete [69], and this hardness persists even for shallow architectures with a single hidden layer [108]. Going beyond reachability, robustness verification—deciding whether all points in a perturbation set around a given input yield outputs in a safe region—has also been shown NP-complete [115]. Other properties can reach higher complexity: for instance, deciding surjectivity of ReLU networks can be Σ_2^{P}-complete [53]. These results delineate precise tractability frontiers, showing that certain restrictions—such as monotonic activations, fixed topology, or bounded input dimension—are necessary for polynomial-time verification. Together, these hardness classifications form the theoretical foundation for why approximate or restricted verification methods are often unavoidable in practice [69, 62].

- **Advanced verification:** Recently, research has increasingly focused on more sophisticated verification models and tasks. One prominent line of work investigates the decidability of verifying neural networks with *smooth* activation functions, such as sigmoid or tanh. It has been shown that this problem is equivalent to the decidability of the FO theory of the reals extended with the exponential function—a long-standing open problem in model theory known as *Tarski's exponential function problem* [63]. Another area of study is #NN-Verification, the counting analogue of standard neural network verification. The objective here is to determine the number of inputs that violate a given safety property. This problem is #P-complete, motivating the design of exact algorithms and randomized approximation techniques [89]. Building on this, a generalized probabilistic verification framework has been proposed, aiming to compute bounds on the probability of property violations under arbitrary input distributions. This probabilistic variant is also #P-hard, and a branch-and-bound approach has been

developed to address it, with formal proofs guaranteeing both soundness and completeness [31].

8 Knowledge Representation

Theoretical research in knowledge representation (KR) is essential for the advancement of ML because it provides a complementary perspective that helps build AI systems that are both powerful and interpretable. KR has been part of AI since its inception, giving the community deep insights into the strengths and limitations of purely data-driven methods, as well as a rich set of tools for addressing key challenges in ML. While the limitations of ML—such as difficulty in reasoning over structured knowledge or capturing complex relational dependencies—are well-known, the field of KR offers concrete technical approaches to address them. Properly integrated, KR can enhance ML by enabling the encoding of explicit domain knowledge, enforcing constraints, explaining predictions in human-understandable terms, and supporting reasoning over learned models. The subsequent sections illustrate some of these technical solutions, including statistical relational learning, probabilistic and differentiable logic, and knowledge graph embeddings, which collectively demonstrate how KR can complement and extend standard ML techniques.

- **Statistical relational learning:** Over the past decade, statistical relational learning (SRL) has made major theoretical advances that sharpen our understanding of the trade-offs between expressivity and tractability, while strengthening the links between logic, probability, and ML. SRL is important for AI because many real-world domains are both relational (entities and their relationships matter) and uncertain (data is noisy, incomplete, or stochastic). SRL provides the formal machinery to jointly model these properties, enabling AI systems to learn from data while reasoning with structured, semantically rich knowledge. A key unifying development has been the view of many SRL formalisms through the lens of *probabilistic circuits* (PCs), where structural constraints guarantee polynomial-time inference and closedness under key transformations, thus reframing inference complexity in relational models in circuit-theoretic terms [39]. Complementing this, there has been work on *lifted inference*, which is the task of doing probabilistic inference over a model without grounding all the variables in it. Lifted inference can be rephrased as the *model counting* problem: Given a FO formula ϕ and an integer n, compute the number of models of ϕ of size n. A precise characterization of which FO formulas admit tractable model counting has been obtained [24]. In addition, several extensions of this tractable fragment have been found by applying combinatorial

techniques [81, 114, 86]. SRL has also expanded beyond purely discrete domains through the theory of *model integration* [25, 94].

- **Probabilistic and differentiable logic:** Probabilistic and differentiable logic frameworks have become crucial for enabling reasoning under uncertainty while still supporting gradient-based optimization. This allows symbolic rules to be learned jointly with neural components. Over the last decade, several theoretical advances have shaped this area. For instance, differentiable logic systems such as DeepProbLog [87] have introduced end-to-end differentiable reasoning frameworks that extend classical logical semantics with gradient-based learning, together with analyses of complexity and expressivity [85]. This has been complemented by frameworks like *Logic Tensor Networks* (LTNs), which introduced a many-valued, end-to-end differentiable FO logic that unifies learning and reasoning in a single formalism [12]. Further advances include the development of *Logical Neural Networks* (LNNs)—a differentiable neuro-symbolic model where each neuron corresponds to a component of a logical formula, enabling resilience to logical contradictions and support for open-world semantics via real-valued truth bounds [107].

- **Knowledge graph embeddings:** Knowledge graph embeddings (KGEs) have become a central research topic at the intersection of knowledge representation and ML because they provide a scalable way to bridge symbolic relational knowledge with continuous vector-space learning. KGEs operate by mapping entities and relations into geometric spaces where reasoning tasks—such as link prediction, query answering, and ontology completion—can be carried out efficiently. Their design raises fundamental theoretical questions: what kinds of logical patterns, rules, and graphs can different embedding families represent, and what are their inherent limitations? In recent years, substantial theoretical progress has been made in addressing these questions. For example, a geometry-based framework was proposed demonstrating that widely used KGE models fail to capture certain existential rule patterns, and introducing embeddings based on convex regions that can faithfully encode a particular class of existential rules while ensuring both logical consistency and deductive closure [59]. Another important contribution is BoxE, a fully expressive embedding model in which entities are points and relations are boxes in latent space, supporting higher-arity relations and principled incorporation of logical rules [2]. Its temporal extension, BoxTE, preserves full expressivity in temporal knowledge graphs while exhibiting strong inductive generalization [91]. More recently, research has focused on enhancing both the expressivity and interpretability

of KGEs. For instance, ExpressivE represents entities as points and relations as hyper-parallelograms in a 2D Euclidean space [99]. This geometric design enables the model to capture a rich set of inference patterns, including composition and hierarchy, while providing an intuitive geometric interpretation of these patterns. An interesting survey on different aspects of lind prediction and query answering over incomplete knowledge graphs has recently been published [106].

9 Final Remarks

We thought of this initial column as a wide review on how different areas of TCS — both the questions they pose and the tools they develop — have contributed to recent advances in ML, or how they could contribute. And on the flip side, whether developments in ML are starting to shape or inspire work in those fields in any meaningful way.

Those questions appear ubiquitous in TCS: the future publications of the column will dive deeper into topics we glanced over. It will consist of interviews, reviews of specific topics or position papers: we would appreciate any suggestion, thoughts or feedback you might have!

Acknowledgements

We are extremely grateful to Cristobal Guzman, Javier Esparza, Wim Martens, Magdalena Ortiz, Santosh Vempala, and David Woodruff for their invaluable comments and ideas during the preparation of this article.

References

[1] Martín Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *CCS*, pages 308–318, 2016.

[2] Ralph Abboud, Ismail Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. Boxe: A box embedding model for knowledge base completion. In *NeurIPS*, pages 9649–9661, 2020.

[3] Pankaj K. Agarwal, Hsien-Chih Chang, Sharath Raghvendra, and Allen Xiao. Deterministic, near-linear ϵ-approximation algorithm for geometric bipartite matching. In *STOC*, pages 1052–1065. ACM.

[4] Veeti Ahvonen, Damian Heiman, Antti Kuusisto, and Carsten Lutz. Logical characterizations of recurrent graph neural networks with reals and floats. In *NeurIPS*, 2024.

[5] Aws Albarghouthi. *Introduction to Neural Network Verification*. verifieddeeplearning.com, 2021. `http://verifieddeeplearning.com`.

[6] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. In *STOC*, pages 20–29, 1996.

[7] Antoine Amarilli and Florent Capelli. Tractable circuits in database theory. *SIGMOD Rec.*, 53(2):6–20, 2024.

[8] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *FOCS*, pages 459–468. IEEE Computer Society, 2006.

[9] Marcelo Arenas, Daniel Báez, Pablo Barceló, Jorge Pérez, and Bernardo Subercaseaux. Foundations of symbolic languages for model interpretability. In *NeurIPS*, pages 11690–11701, 2021.

[10] Sanjeev Arora, Elad Hazan, and Satyen Kale. $o(\sqrt{(}\log(n))$ approximation to sparsest cut in $\tilde{o}(n^2)$ time. *SIAM J. Comput.*, 39(5):1748–1771, 2010.

[11] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In Nikhil Bansal, Kirk Pruhs, and Clifford Stein, editors, *SODA*, pages 1027–1035. SIAM, 2007.

[12] Samy Badreddine, Artur S. d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks. *Artif. Intell.*, 303:103649, 2022.

[13] Ainesh Bakshi, Piotr Indyk, Rajesh Jayaram, Sandeep Silwal, and Erik Waingarten. Near-linear time algorithm for the chamfer distance. In *NeurIPS*, 2023.

[14] Borja Balle, Giovanni Cherubin, and Jamie Hayes. Reconstructing training data with informed adversaries. In *sp*, pages 1138–1156. IEEE, 2022.

[15] Pablo Barceló, Floris Geerts, Juan L. Reutter, and Maksimilian Ryschkov. Graph neural networks with local graph parameters. In *NeurIPS*, pages 25280–25293, 2021.

[16] Pablo Barceló, Egor V. Kostylev, Mikaël Monet, Jorge Pérez, Juan L. Reutter, and Juan Pablo Silva. The logical expressiveness of graph neural networks. In *ICLR*, 2020.

[17] Pablo Barceló, Alexander Kozachinskiy, Anthony Widjaja Lin, and Vladimir V. Podolskii. Logical languages accepted by transformer encoders with hard attention. In *ICLR*, 2024.

[18] Pablo Barceló, Alexander Kozachinskiy, and Tomasz Steifer. Ehrenfeucht-haussler rank and chain of thought. *ICML*, 2025.

[19] Pablo Barceló, Mikaël Monet, Jorge Pérez, and Bernardo Subercaseaux. Model interpretability through the lens of computational complexity. In *NeurIPS*, 2020.

[20] Solon Barocas, Moritz Hardt, and Arvind Narayanan. Fairness and machine learning. *Recommender systems handbook*, 1:453–459, 2020.

[21] Raef Bassily, Cristóbal Guzmán, and Michael Menart. Differentially private stochastic optimization: New results in convex and non-convex settings. In *NeurIPS*, pages 9317–9329, 2021.

[22] Raef Bassily, Cristóbal Guzmán, and Anupama Nandi. Non-euclidean differentially private stochastic convex optimization. In *COLT*, volume 134, pages 474–499, 2021.

[23] Raef Bassily, Adam D. Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *FOCS*, pages 464–473, 2014.

[24] Paul Beame, Guy Van den Broeck, Eric Gribkoff, and Dan Suciu. Symmetric weighted first-order model counting. In *PODS*, pages 313–328, 2015.

[25] Vaishak Belle, Andrea Passerini, and Guy Van den Broeck. Probabilistic inference in hybrid domains by weighted model integration. In *IJCAI*, pages 2770–2776. AAAI Press, 2015.

[26] Michael Benedikt, Chia-Hsuan Lu, Boris Motik, and Tony Tan. Decidability of graph neural networks via logical characterizations. *CoRR*, abs/2404.18151, 2024.

[27] Lorenzo Beretta and Aviad Rubinstein. Approximate earth mover's distance in truly-subquadratic time. In *STOC*, pages 47–58. ACM, 2024.

[28] Guy Blanc, Jane Lange, Ali Malik, and Li-Yang Tan. Lifting uniform learners via distributional decomposition. In *STOC*. ACM, 2023.

[29] Guy Blanc, Jane Lange, Carmen Strassle, and Li-Yang Tan. A distributional-lifting theorem for PAC learning. In *COLT*, volume 291, pages 375–379. PMLR, 2025.

[30] Avrim Blum and Ronald L. Rivest. Training a 3-node neural network is np-complete. *Neural Networks*, 5(1):117–127, 1992.

[31] David Boetius, Stefan Leue, and Tobias Sutter. Probabilistic verification of neural networks using branch and bound. *CoRR*, abs/2405.17556, 2024.

[32] Jeroen Bollen, Jan Van den Bussche, Stijn Vansummeren, and Jonni Virtema. Halting recurrent gnns and the graded μ-calculus. *CoRR*, abs/2505.11050, 2025.

[33] Moses Charikar, Michael Kapralov, Navid Nouri, and Paris Siminelakis. Kernel density estimation through density constrained near neighbor search. In *FOCS*, pages 172–183. IEEE, 2020.

[34] Moses Charikar and Paris Siminelakis. Hashing-based-estimators for kernel density in high dimensions. In *FOCS*, pages 1032–1043, 2017.

[35] Lijie Chen, Binghui Peng, and Hongxun Wu. Theoretical limitations of multi-layer transformer. *FOCS*, 2025.

[36] Sitan Chen, Vasilis Kontonis, and Kulin Shah. Learning general gaussian mixtures with efficient score matching. In *COLT*, volume 291, pages 1029–1090. PMLR, 2025.

[37] Zhengdao Chen, Lei Chen, Soledad Villar, and Joan Bruna. Can graph neural networks count substructures? In *NeurIPS*, 2020.

[38] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair clustering through fairlets. *NeurIPS*, 30, 2017.

[39] Yitao Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic modeling. *arXiv:2006.10183*, 2020.

[40] Vincent Cohen-Addad, Fabrizio Grandoni, Euiwoong Lee, Chris Schwiegelshohn, and Ola Svensson. A (2+ϵ)-approximation algorithm for metric k-median. In Michal Koucký and Nikhil Bansal, editors, *STOC*, pages 615–624. ACM, 2025.

[41] Vincent Cohen-Addad, David Saulpic, and Chris Schwiegelshohn. A new coreset framework for clustering. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC*, pages 169–182. ACM, 2021.

[42] Vincent Cohen-Addad, David P. Woodruff, and Samson Zhou. Streaming euclidean k-median and k-means with o(log n) space. In *FOCS*, pages 883–908. IEEE, 2023.

[43] Nilesh N. Dalvi and Dan Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. *J. ACM*, 59(6):30:1–30:87, 2012.

[44] Guy Van den Broeck, Anton Lykov, Maximilian Schleich, and Dan Suciu. On the tractability of SHAP explanations. *J. Artif. Intell. Res.*, 74:851–886, 2022.

[45] Ilias Diakonikolas, Weihao Kong, and Alistair Stewart. Efficient algorithms and lower bounds for robust linear regression. In *SODA*, pages 2745–2754. SIAM, 2019.

[46] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *ITCS*, pages 214–226, 2012.

[47] Cynthia Dwork and Christina Ilvento. Individual fairness under composition. *FAccT*, 2018.

[48] Cynthia Dwork, Nitin Kohli, and Deirdre Mulligan. Differential privacy in practice: Expose your epsilons! *Journal of Privacy and Confidentiality*, 9(2), 2019.

[49] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004.

[50] Uriel Feige, Yishay Mansour, and Robert E. Schapire. Learning and inference in the presence of corrupted inputs. In *COLT*, volume 40 of *JMLR Workshop and Conference Proceedings*, pages 637–657, 2015.

[51] Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: A theoretical perspective. In *NeurIPS*, 2023.

[52] Sorelle A Friedler, Carlos Scheidegger, and Suresh Venkatasubramanian. The (im) possibility of fairness: Different value systems require different mechanisms for fair decision making. *Communications of the ACM*, 64(4):136–143, 2021.

[53] Vincent Froese, Moritz Grillo, and Martin Skutella. Complexity of injectivity and verification of relu neural networks, 2025.

[54] Khashayar Gatmiry, Jonathan A. Kelner, and Holden Lee. Learning mixtures of gaussians using diffusion models. In *COLT*, volume 291 of *Proceedings of Machine Learning Research*, pages 2403–2456. PMLR, 2025.

[55] Quan Geng, Wei Ding, Ruiqi Guo, and Sanjiv Kumar. Tight analysis of privacy and utility tradeoff in approximate differential privacy. In *AISTATS*, volume 108, pages 89–99. PMLR, 2020.

[56] Martin Grohe. The logic of graph neural networks. In *LICS*, pages 1–17, 2021.

[57] Martin Grohe. The descriptive complexity of graph neural networks. *TheoretiCS*, 3, 2024.

[58] Martin Grohe, Christoph Standke, Juno Steegmans, and Jan Van den Bussche. Query languages for neural networks. In *ICDT*, volume 328, pages 9:1–9:18, 2025.

[59] Víctor Gutiérrez-Basulto and Steven Schockaert. From knowledge graph embedding to ontology embedding? an analysis of the compatibility between vector space representations and rules. In *KR*, pages 379–388, 2018.

[60] Michael Hahn. Theoretical limitations of self-attention in neural sequence models. *Trans. Assoc. Comput. Linguistics*, 8:156–171, 2020.

[61] Yiding Hao, Dana Angluin, and Robert Frank. Formal language recognition by hard attention transformers: Perspectives from circuit complexity. *Trans. Assoc. Comput. Linguistics*, 10:800–810, 2022.

[62] Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. Safety verification of deep neural networks. In *CAV*, pages 3–29, 2017.

[63] Omri Isac, Yoni Zohar, Clark W. Barrett, and Guy Katz. DNN verification, reachability, and the exponential function problem. In *CONCUR*, pages 26:1–26:18, 2023.

[64] Lauren Kirchner Jeff Larson, Surya Mattu and Julia Angwin. How we analyzed the COMPAS recidivism algorithm. Technical report, ProPublica, 2016.

[65] Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala, and Edwin Zhang. Why language models hallucinate. *CoRR*, abs/2509.04664, 2025.

[66] Adam Tauman Kalai and Santosh S. Vempala. Calibrated language models must hallucinate. In *STOC*, pages 160–171, 2024.

[67] Alkis Kalavasis, Anay Mehrotra, and Grigoris Velegkas. On the limits of language generation: Trade-offs between hallucination and mode-collapse. In *STOC*, pages 1732–1743, 2025.

[68] Amin Karbasi, Omar Montasser, John Sous, and Grigoris Velegkas. (im)possibility of automated hallucination detection in large language models. *CoRR*, abs/2504.17004, 2025.

[69] Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *CAV*, pages 97–117, 2017.

[70] Michael Kearns and Aaron Roth. *The ethical algorithm: The science of socially aware algorithm design*. Oxford University Press, 2019.

[71] Michael J. Kearns, Ming Li, Leonard Pitt, and Leslie G. Valiant. On the learnability of boolean formulae. In *STOC*, pages 285–295. ACM, 1987.

[72] Michael J. Kearns and Leslie G. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *J. ACM*, 41(1):67–95, 1994.

[73] Sammy Khalife. Graph neural networks with polynomial activations have limited expressivity. *CoRR*, abs/2310.13139, 2023.

[74] Mahmoud Abo Khamis, Hung Q. Ngo, XuanLong Nguyen, Dan Olteanu, and Maximilian Schleich. In-database learning with sparse tensors. In *PODS*, pages 325–340. ACM, 2018.

[75] Mahmoud Abo Khamis, Hung Q. Ngo, and Atri Rudra. FAQ: questions asked frequently. In *PODS*, pages 13–28. ACM, 2016.

[76] Benny Kimelfeld and Christopher Ré. A relational framework for classifier engineering. In *PODS*, pages 5–20. ACM, 2017.

[77] Jon Kleinberg and Fan Wei. Density measures for language generation, 2025.

[78] Jon M. Kleinberg and Sendhil Mullainathan. Language generation in the limit. In *NeurIPS*, 2024.

[79] Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. In Christos H. Papadimitriou, editor, *ITCS*, 2017.

[80] Adam R. Klivans, Pravesh K. Kothari, and Raghu Meka. Efficient algorithms for outlier-robust regression. In *COLT*, volume 75 of *Proceedings of Machine Learning Research*, pages 1420–1430. PMLR, 2018.

[81] Ondrej Kuzelka. Weighted first-order model counting in the two-variable fragment with counting quantifiers. *J. Artif. Intell. Res.*, 70:1281–1307, 2021.

[82] Matthias Lanzinger and Pablo Barceló. On the power of the weisfeiler-leman test for graph motif parameters. In *ICLR*, 2024.

[83] Kasper Green Larsen and Jelani Nelson. Optimality of the johnson-lindenstrauss lemma. In Chris Umans, editor, *FOCS*, pages 633–638. IEEE Computer Society, 2017.

[84] Jiaoda Li and Ryan Cotterell. Characterizing the expressivity of transformer language models, 2025.

[85] Jaron Maene, Vincent Derkinderen, and Luc De Raedt. On the hardness of probabilistic neurosymbolic learning. In *ICML*. OpenReview.net, 2024.

[86] Sagar Malhotra, Davide Bizzaro, and Luciano Serafini. Lifted inference beyond first-order logic. *Artif. Intell.*, 342:104310, 2025.

[87] Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Neural probabilistic logic programming in deepproblog. *Artif. Intell.*, 298:103504, 2021.

[88] Yishay Mansour, Aviad Rubinstein, and Moshe Tennenholtz. Robust probabilistic inference. In *SODA*, pages 449–460. SIAM, 2015.

[89] Luca Marzari, Davide Corsi, Ferdinando Cicalese, and Alessandro Farinelli. The #dnn-verification problem: Counting unsafe inputs for deep neural networks. In *IJCAI*, pages 217–224, 2023.

[90] William Merrill and Ashish Sabharwal. The expressive power of transformers with chain of thought. In *ICLR*, 2024.

[91] Johannes Messner, Ralph Abboud, and İsmail İlkan Ceylan. Temporal knowledge graph completion using box embeddings. In *AAAI*, pages 7779–7787, 2022.

[92] Brent Daniel Mittelstadt, Patrick Allo, Mariarosaria Taddeo, Sandra Wachter, and Luciano Floridi. The ethics of algorithms: Mapping the debate. *Big Data & Society*, 3(2):2053951716679679, 2016.

[93] Omar Montasser, Steve Hanneke, and Nathan Srebro. VC classes are adversarially robustly learnable, but only improperly. In *COLT*, volume 99, pages 2512–2530. PMLR, 2019.

[94] Paolo Morettin, Andrea Passerini, and Roberto Sebastiani. Efficient weighted model integration via smt-based predicate abstraction. In *IJCAI*, pages 720–728, 2017.

[95] Christopher Morris, Floris Geerts, Jan Tönshoff, and Martin Grohe. WL meet VC. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *ICML*, volume 202, 2023.

[96] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI*, pages 4602–4609, 2019.

[97] Sebastian Ordyniak, Giacomo Paesani, Mateusz Rychlicki, and Stefan Szeider. Explaining decisions in ML models: A parameterized complexity analysis. In *KR*, 2024.

[98] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *ICLR*, 2017.

[99] Aleksandar Pavlovic and Emanuel Sallinger. Expressive: A spatio-functional embedding for knowledge graph completion. In *ICLR*, 2023.

[100] Charlotte Peale, Vinod Raman, and Omer Reingold. Representative language generation. *CoRR*, abs/2505.21819, 2025.

[101] Binghui Peng, Srini Narayanan, and Christos H. Papadimitriou. On limitations of the transformer architecture. *CoRR*, abs/2402.08164, 2024.

[102] Jorge Pérez, Pablo Barceló, and Javier Marinkovic. Attention is turing-complete. *J. Mach. Learn. Res.*, 22:75:1–75:35, 2021.

[103] Maximilian Pflueger, David Tena Cucala, and Egor V. Kostylev. Recurrent graph neural networks and their connections to bisimulation and logic. In *AAAI*, pages 14608–14616, 2024.

[104] Jeff M. Phillips and Wai Ming Tai. Near-optimal coresets of kernel density estimates. *Discret. Comput. Geom.*, 63(4):867–887, 2020.

[105] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On fairness and calibration. *NeurIPS*, 30, 2017.

[106] Hongyu Ren, Mikhail Galkin, Zhaocheng Zhu, Jure Leskovec, and Michael Cochez. Neural graph reasoning: A survey on complex logical query answering. *Trans. Mach. Learn. Res.*, 2024, 2024.

[107] Ryan Riegel, Alexander G. Gray, Francois P. S. Luus, Naweed Khan, Ndivhuwo Makondo, Ismail Yunus Akhalwaya, Haifeng Qian, Ronald Fagin, Francisco Barahona, Udit Sharma, Shajith Ikbal, Hima Karanam, Sumit Neelam, Ankita Likhyani, and Santosh K. Srivastava. Logical neural networks. *CoRR*, abs/2006.13155, 2020.

[108] Marco Sälzer and Martin Lange. Reachability is np-complete even for the simplest neural networks. In *RP*, pages 149–164, 2021.

[109] Lena Strobl, William Merrill, Gail Weiss, David Chiang, and Dana Angluin. What formal languages can transformers express? A survey. *Trans. Assoc. Comput. Linguistics*, 12:543–561, 2024.

[110] Dan Suciu, Dan Olteanu, Christopher Ré, and Christoph Koch. *Probabilistic Databases*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.

[111] Balder ten Cate and Víctor Dalmau. The product homomorphism problem and applications. In *ICDT*, pages 161–176, 2015.

[112] Balder ten Cate, Victor Dalmau, Maurice Funk, and Carsten Lutz. Extremal fitting problems for conjunctive queries. In *PODS*, pages 89–98. ACM, 2023.

[113] Leslie G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984.

[114] Timothy van Bremen and Ondrej Kuzelka. Lifted inference with tree axioms. In *KR*, pages 599–608, 2021.

[115] Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *NeurIPS*, pages 3839–3848, 2018.

[116] Zixuan Wang, Eshaan Nichani, Alberto Bietti, Alex Damian, Daniel Hsu, Jason D. Lee, and Denny Wu. Learning compositional functions with transformers from easy-to-hard data. In *COLT*, volume 291, pages 5632–5711. PMLR, 2025.

[117] Wikipedia contributors. Fairness (machine learning) — Wikipedia, the free encyclopedia, 2024. [Online; accessed 24-September-2024].

[118] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*. OpenReview.net, 2019.

[119] Andy Yang and David Chiang. Counting like transformers: Compiling temporal counting logic into softmax transformers. *CoRR*, abs/2404.04393, 2024.

[120] Andy Yang, David Chiang, and Dana Angluin. Masked hard-attention transformers recognize exactly the star-free languages. In *NeurIPS*, 2024.

[121] Andy Yang, Lena Strobl, David Chiang, and Dana Angluin. Simulating hard attention using soft attention. *CoRR*, abs/2412.09925, 2024.