Undecidability of Underfitting in Learning Algorithms

Sonia Sehra
Cloud and AI
Microsoft
Redmond, WA, USA
sonia.sehra@microsoft.com

David Flores
AMISTAD Lab
Harvey Mudd College
Claremont, CA, USA
deflores@hmc.edu

George D. Montañez
AMISTAD Lab
Harvey Mudd College
Claremont, CA, USA
gmontanez@hmc.edu

***Abstract*—Using recent machine learning results that present an information-theoretic perspective on underfitting and overfitting, we prove that deciding whether an encodable learning algorithm will always underfit a dataset, even if given unlimited training time, is undecidable. We discuss the importance of this result and potential topics for further research, including information-theoretic and probabilistic strategies for bounding learning algorithm fit.**

***Index Terms*—undecidability, underfitting, machine learning**

I. Introduction

Overfitting and underfitting are often explained as symptoms of the bias-variance trade-off [1], [2], where overfitting describes when a learning method has low training but high test error, and underfitting occurs when a method has high training and high test error. Seeking robust definitions of overfitting and underfitting, we build on recent work in machine learning looking at both phenomena from an information-theoretic perspective [3].

Information-theoretic notions of overfitting reflect when algorithms go beyond learning true regularities in data, inadvertently capturing noise in their modeling processes. In contrast, underfitting describes when an algorithm "fails to capture enough" information to learn the true regularities in data. Marrying this intuitive notion of underfitting with existing definitions of underfitting can help us derive new insights and define specific circumstances into when and how algorithms underfit.

In this paper, we examine definitions of underfitting using notions of algorithmic capacity and dataset complexity developed by Bashir et al. [3], and further, prove that determining whether an arbitrary learning algorithm will always *underfit* a particular dataset is formally undecidable. We will demonstrate this by a reduction from the halting problem.

Our proof for the undecidability of the underfitting uses definitions from Bashir et al. [3], building on that work while focusing primarily on underfitting. In their aforementioned paper, Bashir et al. proved the formal undecidability of the overfitting problem, while leaving open the problem of establishing a similar proof for underfitting. Inspired by their method, we accomplish that task here.

II. Related Work

This paper builds on the larger algorithmic search framework for machine learning developed by Montañez [4], on which Bashir et al. also build [3]. Our notions of algorithm and dataset complexity tie into an existing body of work developed by Lauw et al. [5], while using a definition of underfitting most closely related to that of Bashir et al. [3]. These papers build on previous work characterizing the capacity of machine learning algorithms, such as the VC dimension [6], Rademacher complexity [7], and Labeling Distribution Matrices [8].

There has been a large body of research into underfitting in machine learning algorithms, and approaches that can be taken to detect and avoid underfitting algorithms. Gavrilov et al. studied causes of over- and underfitting, and methods to prevent it, in Convolutional Neural Nets [9]. Li et al. did similar work for decision trees [10], and Narayan et al. did similar work for Multilayer Perceptrons [11]. Many other examples could be given.

There has also been other recent work on decidability for problems related to machine learning. Ben-David et al. recently proved that the learnability problem is undecidable [12]. Building on that work, Gandolfi proved that the sample complexity of an algorithm, a measure of how many samples are needed to solve a problem, is decidable in some circumstances [13].

A. Relation to Bashir et al. [3]

As we have already noted, this paper is largely built around the definitions developed by Bashir et al. [3]. Our work accomplishes three main goals: (1) it expands on the brief description of underfitting presented in that paper, (2) gives a new model-specific definition of underfitting, and (3) proves the formal undecidability of underfitting under this definition.

While Bashir et al. introduced an information-theoretic definition for underfitting, their analysis primarily focused on overfitting rather than underfitting. Our paper analyzes their definition of underfitting, creates a complementary model-specific definition of underfitting, and finally presents rigorous conclusions which can be drawn from that definition. In particular, we prove that underfitting, like overfitting, is formally undecidable.

III. Background

For completeness, we reproduce several of the definitions developed by Bashir et al., and employ the same assumptions regarding algorithms, datasets, and hypothesis spaces that are used in that work.

We begin with Definition 4 from Bashir et al., a definition of time-indexed capacity. Let $\mathcal{G}$ denote a finite hypothesis space available to a learning algorithm $\mathcal{A}$ (or more generally, let $\mathcal{G}$ be a finite search space sampled by a search algorithm $\mathcal{A}$).

Definition III.1 (Time-indexed Capacity, from [3])**.** Let P_i denote the (stochastic) probability distribution over $\mathcal{G}$ at time i. $\mathcal{A}$'s capacity *at time i* is the maximum amount of information $\mathcal{A}$ may transfer from a dataset $D \sim \mathcal{D}$ to $G_i \sim \mathbb{E}[P_i|D]$,

$$C^i_{\mathcal{A}} = \sup_{\mathcal{D}} I(G_i; D),$$

where $\mathcal{A}$ is a learning algorithm and $\mathcal{G}$ is the algorithm's hypothesis space.

Definition III.1 gives us a limit on how much information a learning algorithm is able to extract from a dataset at time i in reference to a finite hypothesis space $\mathcal{G}$. It does so by making use of the information-theoretic quantity of mutual information, which captures the level of dependence between two random variables.

Next, we present Definitions 6 and 7 from Bashir et al., for Dataset Turing Complexity and Dataset Complexity. These definitions allow us to characterize the (algorithmic) information content of arbitrary datasets.

Definition III.2 (Dataset Turing Complexity, from [3])**.** Given a fixed Turing Machine M that accepts a string p and feature vector x as input and outputs a label y, the *data complexity* of a dataset D is

$$C_{D,M} = L(\langle M \rangle) + L(p),$$

where $L(p) = \min\{|p| : \forall(\mathbf{x}, y) \in D, M(p, \mathbf{x}) = y\}$. That is, the data complexity $C_{D,M}$ is the length of the shortest program that can correctly map every input in the dataset $\mathcal{D}$ to its corresponding output.

Before we present the definition of Dataset Complexity from Bashir et al., we need to introduce the quantity C'_D.

Definition III.3 (C'_D, from [3])**.**

$$C'_D = \sum_{i=1}^{n} b(z_i),$$

where $b(z_i)$ is the number of bits required to encode the feature-label pair z_i from dataset $D = (z_1, \ldots, z_n)$, without any compression.

As noted in Bashir et al., C'_D represents the amount of information needed to memorize a dataset D without compression. Now, we define Dataset Complexity, following Definition 7 of Bashir et al.:

Definition III.4 (Dataset Complexity, from Bashir et al. [3])**.** $C_D = \min\{C_{D,M}, C'_D\}$.

Note that $C_D \geq C'_D$, allowing the dataset complexity C_D to be easily upper bounded, which is helpful since $C_{D,M}$ is generally uncomputable.

Having provided some background definitions from Bashir et al., we define underfitting and present several related definitions. Note, we assume that $D \sim \mathcal{D}$ whenever the distribution of D is not stated explicitly in the definitions that follow.

First, Definition 9 of Bashir et al. defines underfitting at iteration i as:

Definition III.5 (Underfitting, from Bashir et al. [3])**.** An algorithm $\mathcal{A}$ **underfits** at iteration i if

$$C^i_{\mathcal{A}} < \mathbb{E}_{\mathcal{D}}[C_D]$$

i.e., after training for i timesteps, $\mathcal{A}$ has time-indexed capacity strictly less than $\mathbb{E}_{\mathcal{D}}[C_D]$.

As noted in [3], underfitting *"could be the result of insufficient capacity, insufficient training, or insufficient information retention, all of which are captured by $C^i_{\mathcal{A}}$."*

The pointwise information transfer from a single dataset to a single model is also defined, which is useful in defining overfitting and underfitting of a particular model (hypothesis) g on a particular dataset d.

Definition III.6 (Pointwise Information Transfer [3])**.** For a given dataset d and specific hypothesis g, the **pointwise information transfer** by algorithm $\mathcal{A}$ from d to g is the pointwise mutual information (lift),

$$C_{\mathcal{A}}(g, d) = \log_2 \frac{p(g, d)}{p(g)p(d)} = \log_2 \frac{p(g|d)}{p(g)} = \log_2 \frac{p(d|g)}{p(d)}.$$

In the same way that Bashir et al. considered overfitting for a single model and dataset, using pointwise information transfer we can define model underfitting.

Definition III.7 (Model Underfit)**.** $\mathcal{A}$'s **model** g **underfits** d if $C_{\mathcal{A}}(g, d) < C_d$.

These definitions will allow us to prove the formal undecidability of the underfitting problem, which we turn to next.

IV. Undecidability of Underfitting

We now demonstrate the undecidability of underfitting as Bashir et al. did for overfitting. Whether an iteratively trained learning algorithm underfits, fits, or overfits is often a matter of just how much training has been allowed. Some algorithms initially produce models that underfit, but with more training eventually fit the data well. Other learning methods suffer from such low representational capacity that they can never produce a model that fits, even with unlimited training time. We prove here that determining whether an algorithm will eventually produce a model that does not underfit is formally undecidable. Like Bashir and collaborators, we do so by a reduction from the halting problem.

While we derive our proof using the model-specific notion of underfitting from Definition III.7, without too much work the proof can be modified to employ the expectation-centric definition of underfitting from Definition III.5. This suggests that even determining if an algorithm will always underfit relative to a distribution on datasets is also formally undecidable, since we can always create an algorithm whose maximum capacity changes whenever a Turing machine M halts on an input w. Thus, it doesn't matter if we're comparing the pointwise algorithm capacity against a fixed dataset complexity or the maximum algorithm capacity against an expected dataset complexity; it is the changing algorithm capacity in response to algorithmic halting that does the work in the proof. We present our main result next.

Theorem 1 (The Undecidability of Underfitting)**.** *Let S be the set of all encodable learning algorithms and let $\langle \mathcal{A} \rangle$ denote the encoded form of algorithm $\mathcal{A}$. Then, for any dataset d,*

$$L_{underfit} = \{\langle \mathcal{A} \rangle, d | \mathcal{A} \in S, \mathcal{A} \text{ underfits } d\, textatalliterations\}$$

is undecidable.

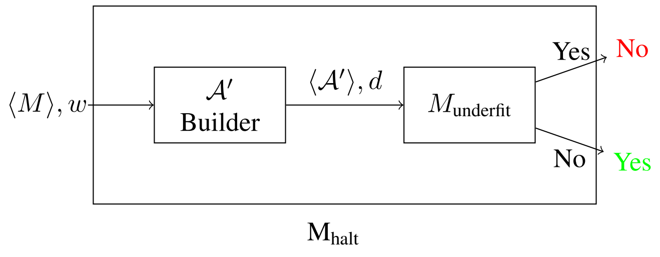

Fig. 1. M_{halt} constructed using M_{underfit}.

Proof. We show that L_{underfit} is undecidable using a reduction from the halting problem. By way of contradiction, assume that L_{underfit} is decidable. There then exists a Turing machine, denoted M_{underfit}, which halts for all inputs of the form $\langle \mathcal{A} \rangle$,$d$ and determines whether $\mathcal{A}$ will always produce models that underfit d, that is, which do not transfer enough information from the dataset to the model to capture the relationship of the training data. In formal terms, a model g^i at iteration i underfits dataset d whenever $C_{\mathcal{A}}(g^i, d) < C_d$, in accordance with Definition III.7.

We now use M_{underfit} to construct a decider for L_{halt} with the given steps (this decider is also shown in Figure 1):

We create an auxiliary machine called $\mathcal{A}'$ *builder*, which takes as input a Turing machine encoding and input string, $\langle M \rangle, w$. $\mathcal{A}'$ builder then creates an encoded algorithm $\mathcal{A}'$ representing an iterative learning method, exporting the encoded algorithm together with a training dataset d. The dataset d consists of a single training example with $k-1$ binary features and a single binary label, for some positive integer $k > 1$. We construct $d = \{(x_1, y_1)\}$ as follows, where x_1 denotes features and y_1 the label. The features are chosen uniformly at random, as is the label; simply put, we generate a k-length binary string by flipping a fair coin, and take the first $k-1$ bits as the features x_1, and the final bit as y_1. This produces one of 2^k possible datasets with equal probability, where $C'_d = k$ for each one (since the k-length binary string fully encodes the feature-label pair, without compression). Thus, $p(d) = 2^{-k} = 2^{-C'_d}$. Also note that $C_d > 0$ because d is nonempty and so any encoded Turing machine that produces it must consist of one or more bits.

$\mathcal{A}'$ works as follows: On its initial iteration, it produces a learning model that outputs a constant zero value for all labels, independent of the data. We consider this point in time t_1 and let g^{t_1} represent this initial constant model. Thus, $\mathcal{A}$ has a pointwise information transfer of zero, $C_{\mathcal{A}}(g^{t_1}, d) = 0$, since $p(g^{t_1}|d) = p(g^{t_1})$, as the algorithm will produce this initial model with probability 1, independent of d. Therefore,

$$0 = C_{\mathcal{A}}(g^{t_1}, d) < C_d,$$

and we see that at time point t_1 all of $\mathcal{A}$'s models produced so far (namely, the single model g^{t_1}) will **underfit** by Definition III.7.

After this, $\mathcal{A}'$ simulates M on w. If M halts on input w, $\mathcal{A}'$ then updates its internal model as follows. For any input with features equal to x_1 it predicts the label y_1, effectively memorizing the datapoint. For all other inputs, it produces the label $1 - y_1$, that is, it negates the label y_1. For example, if $y_1 = 1$ and $x_1 = (0, 0, 1, 1)$, then the model will have exactly one response equal to 1, namely for input $(0, 0, 1, 1)$, and all other inputs will map to a response label of 0. If $y_1 = 0$, we would have exactly one response with value of 0, and all other inputs would map to 1. Thus, at time point t_2 we will have one of 2^k possible models. Note that a bijective mapping exists between the model g^{t_2} and the dataset d; changing a single bit in the dataset d results in a different model. Therefore, $p(g^{t_2}|d) = 1$ (the algorithm's choice is deterministic given d), and

$$\begin{aligned} p(g^{t_2}) &= \sum_{d'} p(g^{t_2}|d')p(d') \\ &= 1 \cdot p(d) + \sum_{d' \neq d} 0 \cdot p(d') \\ &= 2^{-C'_d} \end{aligned}$$

by construction. Using Definition III.6 we obtain

$$\begin{aligned} C_{\mathcal{A}}(g^{t_2}, d) &= \log_2 p(g^{t_2}|d)/p(g^{t_2}) \\ &= \log_2 2^{C'_d} \\ &= C'_d \\ &\geq C_d \end{aligned}$$

and $\mathcal{A}'$ produces a model that does **not underfit** at time point t_2, according to Definition III.7.

If M does not halt on input w, then the algorithm only completes a single iteration, leaving the original underfitting model intact. Thus, $\mathcal{A}'$ will underfit d at all iterations if and only if machine M does not halt on input w.

Under the assumption that M_{underfit} exists, we can pass the outputs of $\mathcal{A}'$ builder to this machine and ask M_{underfit} if $\mathcal{A}'$ underfits d at all iterations. The way it answers will tell us whether M halts on w, as it will always underfit only in the case that M does not halt on input w. The outputs from M_{underfit} are then swapped and routed to the output of machine M_{halt}, creating a decider for L_{halt}, which is a contradiction. Having reached a contradiction, our initial assumption that M_{underfit} exists cannot hold, and L_{underfit} is therefore undecidable. □

V. Discussion

We have proven that, under a rigorous and reasonable information-theoretic definition of underfitting, the problem of determining whether an arbitrary learning algorithm will eventually fit a dataset given enough training time is formally undecidable. While a perfect algorithm cannot exist, this does not rule out underfitting detection in special cases, such as for fixed-capacity algorithms. However, our theorem guarantees no underfitting predictor can be universally applicable: it will either apply only to a subset of algorithms, will sometimes produce incorrect results, or will fail to terminate when applied to some algorithm and dataset pairs.

Opportunities for future work remain. Just as Bashir et al. established bounds for algorithm capacity and distributional algorithm capacity [3], it should be possible to create bounds using time-indexed capacity. Creating such bounds may allow for new insights on when algorithms might underfit. Research into the special case of fixed-capacity learning algorithms seems especially promising. Another direction to explore is whether the degree to which an algorithm underfits, captured by $\mathbb{E}_{\mathcal{D}}[C_D] - C^i_{\mathcal{A}}$, can be used to bound a learning method's generalization error.

VI. Conclusion

Underfitting remains a problem when training iterative algorithms. In trying to understand and describe this phenomenon, we may be tempted to create a list of criteria to predict exactly when an algorithm will underfit. While being able to determine with certainty that an arbitrary algorithm will underfit would be useful, we show that in general this cannot be done, for if it could, we would also be able to decide the halting problem.

Although it is impossible to always determine whether an algorithm will underfit, this does not rule out probabilistic bounds on the likelihood of underfitting nor exact determination for specific classes of learning algorithms. Future work may include bounding the probability of underfitting given an algorithm and a dataset. Investigating these questions may also lead to a better understanding of why common solutions for underfitting work, by linking these strategies to information-theoretic notions of algorithm capacity and mutual information.

Author Contributions

S. Sehra is responsible for the initial formulation of the paper and proof and for the key insight that the underfitting problem may be formally undecidable. D. Flores contributed to the Related Work section, much of the Introduction and Conclusion, and the Discussion section. G. Montañez contributed Definition III.7 and formally proved Theorem 1, with support in figure creation and layout from Sehra and Flores. Montañez contributed a majority of the text from the preamble of Section IV. All authors contributed to the prose of the manuscript, and all authors participated in editing the final manuscript, sharing responsibility for the conclusions and content contained herein.

References

[1] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural computation*, vol. 4, no. 1, pp. 1–58, 1992.

[2] R. Kohavi, D. H. Wolpert *et al.*, "Bias plus variance decomposition for zero-one loss functions," in *ICML*, vol. 96, 1996, pp. 275–83.

[3] D. Bashir, G. D. Montañez, S. Sehra, P. Segura Sandoval, and J. Lauw, "An Information-Theoretic Perspective on Overfitting and Underfitting," *Australasian Joint Conference on Artificial Intelligence (AJCAI 2020)*, 2020. [Online]. Available: http://arxiv.org/abs/2010.06076

[4] G. D. Montañez, "Why Machine Learning Works," Ph.D. dissertation, Carnegie Mellon University, 2017.

[5] J. Lauw, D. Macias, A. Trikha, J. Vendemiatti, and G. D. Montañez, "The Bias-Expressivity Trade-off," in *Proceedings of the 12th International Conference on Agents and Artificial Intelligence, ICAART 2020, Volume 2, Valletta, Malta, February 22-24, 2020*, A. P. Rocha, L. Steels, and H. J. van den Herik, Eds. SCITEPRESS, 2020, pp. 141–150. [Online]. Available: https://doi.org/10.5220/0008959201410150

[6] V. N. Vapnik and A. Y. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," in *Measures of Complexity*. Springer, 2015, pp. 11–30.

[7] M.-F. Balcan, "Rademacher Complexity," 2011, [Online; accessed 26-February-2019]. [Online]. Available: http://www.cs.cmu.edu/ ninamf/ML11/lect1117.pdf

[8] P. Sandoval Segura, J. Lauw, D. Bashir, K. Shah, S. Sehra, D. Macias, and G. D. Montañez, "The Labeling Distribution Matrix (LDM): A Tool for Estimating Machine Learning Algorithm Capacity," in *Proceedings of the 12th International Conference on Agents and Artificial Intelligence, ICAART 2020, Volume 2, Valletta, Malta, February 22-24, 2020*, A. P. Rocha, L. Steels, and H. J. van den Herik, Eds. SCITEPRESS, 2020, pp. 980–986. [Online]. Available: https://doi.org/10.5220/0009178209800986

[9] A. D. Gavrilov, A. Jordache, M. Vasdani, and J. Deng, "Preventing Model Overfitting and Underfitting in Convolutional Neural Networks," *International Journal of Software Science and Computational Intelligence (IJSSCI)*, vol. 10, no. 4, pp. 19–28, 2018.

[10] J. Li, S. Fong, S. Mohammed, J. Fiaidhi, Q. Chen, and Z. Tan, "Solving the under-fitting problem for decision tree algorithms by incremental swarm optimization in rare-event healthcare classification," *Journal of Medical Imaging and Health Informatics*, vol. 6, no. 4, pp. 1102–1110, 2016.

[11] S. Narayan and G. Tagliarini, "An analysis of underfitting in MLP networks," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks*, vol. 2. IEEE, 2005, pp. 984–988.

[12] S. Ben-David, P. Hrubeš, S. Moran, A. Shpilka, and A. Yehudayoff, "Learnability Can Be Undecidable," *Nature Machine Intelligence*, vol. 1, no. 1, p. 44, 2019.

[13] A. Gandolfi, "Decidability of Sample Complexity of PAC Learning in Finite Setting," *arXiv preprint arXiv:2002.11519*, 2020.