Research Statement

Rahul Yedida (Applicant for Fall 2024)

Increasingly, the use of machine learning is becoming pervasive across disciplines, and software engineering is no exception. The International Data Corporation (IDC) estimates that by 2026, worldwide spending on AI systems will surpass $300 billion[1]. This increased use of AI systems comes with various costs, especially to the environment. Indeed, the leading conference in machine learning (ML), *NeurIPS*, now encourages authors to provide information from a CO_2 emissions tracker[2]. It is therefore cruicial that we make our ML systems as efficient as possible. The use of computationally cheaper systems is not only better for the environment, but also for reproducibility: when papers use ML systems that require lots of compute power, other researchers who do not have access to equally large funding cannot replicate their work. Throughout my Ph.D., I have relied on the simplest neural networks: feedforward models–and repeatedly achieved state-of-the-art performance using them.

A central theme of my research is understanding theoretical properties of learning algorithms under minimal assumptions and applying the results of those properties to achieve strong performance. For example, I studied the application of the Lipschitz constant of the loss function to gradient-based optimization algorithms such as SGD and RMSprop. That study Yedida et al. (2021b) determined that standard learning rates, and even non-monotonic learning rate policies such as 1cycle, were often too conservative, especially for non-neural learning algorithms. For instance, on the California housing prices dataset, we were able to use a learning rate of 5,163.5 for linear regression, converging in only 2 epochs using SGD (a 10,000x improvement). We further showed that such high learning rates did not lead to divergence; rather, the model continued to converge. This result was also true for deep learners, albeit with lower recommended learning rates: on the MNIST, CIFAR-10, and CIFAR-100 datasets, using ResNet and DenseNet architectures, we demonstrated that models can converge using initial learning rates 12-24x higher than typically used.

During my doctoral studies, I have used hyper-parameter optimization (HPO), real analysis, and deep learning to achieve state-of-the-art results in defect prediction Yedida & Menzies (2021), issue lifetime prediction Yedida et al. (2021c), code smell detection Yedida & Menzies (2022), actionable static code warnings prediction Yedida et al. (2023a), and automated software partitioning Yedida et al. (2023b). Broadly, the overarching theme of my Ph.D. can be distilled into three core aspects:

1. *Landscape* improvements by addressing issues with the **loss function** and manipulating the decision boundary;
2. *Learner* improvements through sample-efficient hyper-parameter optimization;
3. *Generalization* by filtering for flat minima, which have been shown to correlate with generalization Dauphin et al. (2014); Keskar et al. (2016).

For example, in recent work Yedida et al. (2023a), I proposed various data transformation operators that increased the $\beta-$smoothness of the loss function by a median of 33.85%, which consequently led to a median AUC improvement of 100% (mean 140%). A visualization of this improvement in the smoothness of the loss landscape is shown in Figure 1. These figures use the landscape visualization technique of Li et al. (2018), and the xy-plane are two random directions. Currently, my work focuses on exploring the broader utility of the $\beta-$smoothness as a heuristic for hyper-parameter optimization. Across 6 Bayesmark datasets, this has shown to be significantly better than 5 other popular hyper-parameter optimization algorithms ($p < 0.01$). Moreover, on 12 software engineering datasets, this approach outperforms the state-of-the-art for those tasks.

[1] https://www.idc.com/getdoc.jsp?containerId=prUS49670322

[2] https://neurips.cc/Conferences/2022/PaperInformation/PaperChecklist

Two extensions of this work[3] studied the utility of directly optimizing for the smoothness and the strong convexity of the loss respectively. In both studies, we were able to outperform state-of-the-art HPO methods across a variety of datasets. Both studies derived analytical expressions for the smoothness and strong convexity respectively, and used them as heuristics to declare a subset of randomly chosen hyper-parameter configurations as worth investigating further. We showed that computing these values was incredibly computationally cheap (a few seconds), and showed that the error incurred by computing them in a mini-batch fashion could be bounded using a covering argument under mild assumptions. Moreover, we made the connection to generalization by observing that applying these heuristics biased our algorithm towards flatter minima, which is known to be correlated with better generalization. Experimentally, we demonstrated their efficacy on 15 and 24 datasets respectively.

However, the properties of learning algorithms that my work exploits are not always entirely based on formal learning theory. In a paper during my doctoral studies Yedida & Menzies (2021), I hypothesized that much of the misclassification in defect prediction (where there is significant class imbalance) was due to samples in the minority class being ignored by the standard loss function and the decision boundary of the learner being too close to the samples. We mitigated this by proposing fuzzy sampling, which concentrically adds samples around each minority class sample to push the decision boundary away (see Figure 2). That paper used hyper-parameter optimization with feedforward learners, noting that (a) the datasets were simple, as noted by the intrinsic dimensionality, Lipschitz constant, and $\beta-$smoothness (b) feedforward networks using the ReLU activation are universal approximators, and (c) the architecture of the feedforward network could be designed based on the results of Montufar et al. (2014). This yielded improvements in the F1 score of the prior state-of-the-art, which was based on a deep neural network, by up to 115%.

My work has also been useful in industry research. A collaboration with IBM Research led to two papers (Yedida et al. (2021a, 2023b)) systematically studying the state of automated microservice partitioning and proposing an improvement over the state-of-the-art. In one paper, we demonstrated how effective small changes could be towards significant improvements by leveraging advances in empirical machine learning to improve upon existing work. Specifically, we proposed the use of spectral clustering over k-means to better identify structural relationships in the data, and the 1cycle learning policy over exponential decay to exploit the super-convergence phenomenon. Further, we reduced a 3-objective problem to a single-objective problem using a weighted combination of the metrics as an objective. These weights were derived by analyzing the relationships between the various metrics, which we obtained by first performing a Monte

[3]These papers are currently under review.

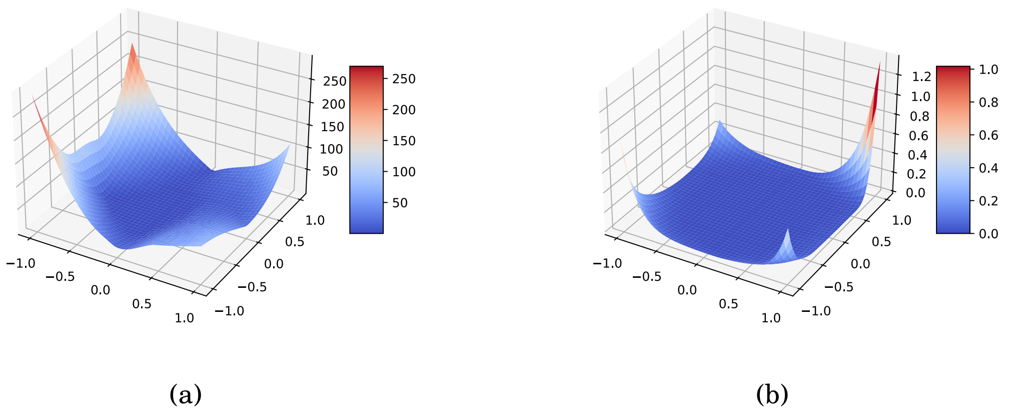

Figure 1: Loss landscape (a) before and (b) after applying data transformation operators from Yedida et al. (2023a).

Carlo sampling of the metrics and analyzing the Spearman correlation coefficients between them. Throughout this collaboration, I have demonstrated strong communication skills and the ability to work effectively with industry professionals. My approachable demeanor and technical expertise have enabled me to build strong relationships with industry partners and facilitate productive discussions. This experience has also provided me with invaluable knowledge about the funding landscape within the industry. By engaging with industry collaborators, I have gained a deep understanding of the priorities and needs of these stakeholders. This expertise will be an essential asset as I continue to pursue research that addresses critical industry challenges.

My research during my Ph.D. lays the groundwork for several future directions in the long term, which I detail below.

Generalization: My work on using smoothness and strong convexity as heuristics for hyper-parameter optimization showed that these are both effective and bias for generalization. However, it is currently unknown *why* they do. The large body of work studying the connection between sharpness and generalization establishes several metrics for flatness, including the Frobenius norm of the Hessian Wu & Su (2023) and variance of gradients. Recent work Dauphin & Cubuk (2020) showed that BatchNorm's regularization effects and consequent performance improvements can be explained by its ability to suppress explosive growth of the activations in the last layer. Our work on strong convexity-guided HPO showed that the strong convexity of a feedforward network is directly proportional to the norm of these activations. In our experiments, an unusually large value of this norm led to an extremely large value of strong convexity, which in turn was correlated with poor performance. This observation corroborates their theoretical results.

As a faculty member, I will work to elucidate the relationship between established sharpness metrics and smoothness and strong convexity. Subsequent work, will then focus on establishing new metrics for sharpness that are computationally cheap to compute, similar to the computations for smoothness and strong convexity in my recent work.

Hyper-parameter optimization: While much of my research has endorsed the value of HPO, only recently did I propose the sample-efficient method described above. This method currently works by sampling 50 random configurations, computing their strong convexity/smoothness, and training only the top 10 according to these heuristics. This is only 33% of the effort involved as compared to other state-of-the-art methods. Although I did attempt to replace the random sampling with Bayesian optimization, I found that it very quickly converges to a single, often sub-optimal, configuration. Therefore, I will work to devise strategies that incorporate the search from Bayesian optimization with diversity of the samples so that different, but promising areas of the hyper-parameter space can be explored.

Another avenue I will explore is to incorporate user metrics in my HPO method described. Currently, the method uses either smoothness or strong convexity as a heuristic, but cannot use user-specified metrics to guide hyper-parameter search. A first step towards resolving this is to note that the Pareto frontier of options is a subset of the convex hull of all options. To find con-

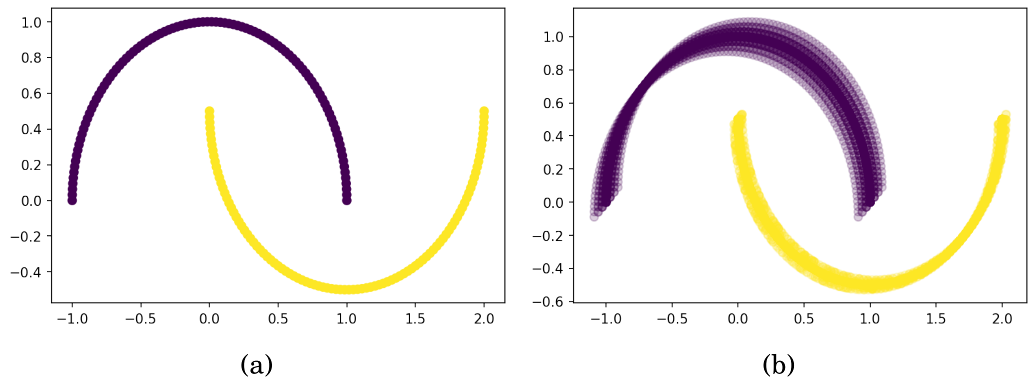
(a) (b)

Figure 2: Fuzzy sampling demo (a) before and (b) after application.

figurations that do well on some metric, we traverse the Pareto frontier and compute quantized[4] convex combinations of adjacent points and also test them. This adds computational cost, so the next step would be to incorporate search into this idea, as discussed in the first paragraph.

However, improvements upon the work above is only a short-term research goal. In the longer term, I will work on highlighting the underlying principles that make some HPO algorithms better than others, whether that be search capabilities via better acquisition functions (e.g., BOHB), pruning the space for better loss landscapes (as I have done), or other metrics such as variability of the loss, which a recent paper Yu & Zhang (2021) has shown to be a good predictor for trainability. As such, this longer-term goal will work towards understanding the relationships between trainability, generalization, and the various metrics that researchers have studied over the years.

Applied machine learning: Over the course of my doctoral program, my work has repeatedly achieved state-of-the-art results for various software engineering tasks including defect prediction, automated microservice partitioning, and predicting actionable static code warnings. Each dataset brings unique challenges such as class imbalance (for defect prediction), very small (< 10 samples) datasets (for static code warnings), and nonstandard data structures (for microservice partitioning), which my work has addressed in appropriate ways. For example, static code warnings datasets sometimes had less than 10 samples, but we were able to learn using a combination of label smoothing, hyper-parameter optimization, boundary engineering, and instance engineering.

I was also involved in a research project in computational medicine Yedida et al. (2022), whose goal was to extract novel drug repurposing hypotheses as well as known drug-disease relations from unstructured text (in this project, we used transcripts from a medical radio show). We then used a biomedical knowledge graph, ROBOKOP, to show the clinical outcome pathways (COPs) for the predicted relations. For example, our model was able to identify a relationship between testosterone deficiency and obesity, which have been linked in a bidirectional manner.

Software analytics: My strong research background will allow me to work further on software engineering problems and bring my theoretical knowledge to the field, which is currently a minority of the papers in SE. Specifically, I will work on a closed-loop approach to applied ML: understand *why* algorithms that perform well do so, and refine algorithms to further improve the state-of-the-art. As an example of work where I did this, in Yedida et al. (2023a), a key challenge was that the labels of the dataset were potentially untrustworthy. Therefore, one of the operators I proposed to smooth the landscape in Figure 1 was "label smoothing", which after clustering using a kd-tree, queried that tree to relabel samples. Our ablation study showed that this (and all the other) techniques yielded statistically significant improvements in test performance. While this method worked for that one case study, I will work to test these techniques across multiple SE tasks and datasets. The data collected from these experiments will enable me in the long term to present a general ML pipeline that software practitioners can reliably apply in their work, without worrying about the details of the algorithms themselves. My goal here is to allow software developers, DevOps engineers, and related practitioners to focus on their *software* problems while being aided by ML tools such as static code analyzers and defect predictors, without worrying about the *reliability* and *trustworthiness* of those ML systems.

References

Dauphin, Y., & Cubuk, E. D. (2020). Deconstructing the regularization of batchnorm. In *International Conference on Learning Representations*.

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., & Bengio, Y. (2014). Identifying

[4]This is only necessary if the space is not dense in $\mathbb{R}^d$; for example, Z is nowhere dense in $\mathbb{R}$ so that for a space $\mathbb{R}^n \times \mathbb{Z}$, quantization would be necessary.

and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in neural information processing systems*, *27*.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. T. P. (2016). On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*.

Li, H., Xu, Z., Taylor, G., Studer, C., & Goldstein, T. (2018). Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, *31*.

Montufar, G. F., Pascanu, R., Cho, K., & Bengio, Y. (2014). On the number of linear regions of deep neural networks. *Advances in neural information processing systems*, *27*.

Wu, L., & Su, W. J. (2023). The implicit regularization of dynamical stability in stochastic gradient descent. *arXiv preprint arXiv:2305.17490*.

Yedida, R., Beasley, J.-M., Korn, D., Abrar, S. M., Melo-Filho, C. C., Muratov, E., Graedon, J., Graedon, T., Chirkova, R., & Tropsha, A. (2022). Text mining of the people's pharmacy radio show transcripts can identify novel drug repurposing hypotheses. *medRxiv*, (pp. 2022–02).

Yedida, R., Kang, H. J., Tu, H., Yang, X., Lo, D., & Menzies, T. (2023a). How to find actionable static analysis warnings: A case study with findbugs. *IEEE Transactions on Software Engineering*.

Yedida, R., Krishna, R., Kalia, A., Menzies, T., Xiao, J., & Vukovic, M. (2021a). Lessons learned from hyper-parameter tuning for microservice candidate identification. In *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, (pp. 1141–1145). IEEE.

Yedida, R., Krishna, R., Kalia, A., Menzies, T., Xiao, J., & Vukovic, M. (2023b). An expert system for redesigning software for cloud applications. *Expert Systems with Applications*, *219*, 119673.

Yedida, R., & Menzies, T. (2021). On the value of oversampling for deep learning in software defect prediction. *IEEE Transactions on Software Engineering*, *48*(8), 3103–3116.

Yedida, R., & Menzies, T. (2022). How to improve deep learning for software analytics: (a case study with code smell detection). In *Proceedings of the 19th International Conference on Mining Software Repositories*, (pp. 156–166).

Yedida, R., Saha, S., & Prashanth, T. (2021b). Lipschitzlr: Using theoretically computed adaptive learning rates for fast convergence. *Applied Intelligence*, *51*, 1460–1478.

Yedida, R., Yang, X., & Menzies, T. (2021c). Old but gold: Reconsidering the value of feedforward learners for software analytics. *arXiv preprint arXiv:2101.06319*.

Yu, Y., & Zhang, Y. (2021). Multi-layer perceptron trainability explained via variability. *arXiv e-prints*, (pp. arXiv–2105).