Is It Worth to Approximate Fitness by Machine Learning?: Investigation on the Extensibility According to Problem Size

Dong-Pil Yu
Dept. Comp. Sci, Kwangwoon Univ
Seoul, Republic of Korea
yoodongphil@naver.com

Yong-Hyuk Kim
Dept. Comp. Sci, Kwangwoon Univ
Seoul, Republic of Korea
yhdfly@kw.ac.kr

ABSTRACT

It is usual to need an approximate model in evolutionary computation when fitness function is deemed to be abstract or expected to have a long computation time. In these cases, research on possibility of fitness approximation should proceed before applying an evolutionary algorithm in real-world problems. In this paper, it was found that we could train machine learning algorithms with the sampled solutions when problem size is large, if there is a possibility of fitness approximation at small problem sizes.

CCS CONCEPTS

• **Computing methodologies** → **Genetic algorithms**; *Machine learning*;

KEYWORDS

Genetic algorithm, fitness approximation, machine learning

ACM Reference Format:
Dong-Pil Yu and Yong-Hyuk Kim. 2018. Is It Worth to Approximate Fitness by Machine Learning?: Investigation on the Extensibility According to Problem Size. In *GECCO '18 Companion: Genetic and Evolutionary Computation Conference Companion, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3205651.3208773

1 INTRODUCTION

In order to assess the quality of a solution in evolutionary computation, each solution is endowed fitness. The fitness functions that calculate fitness may not be clear depending on the given problems, and obtaining fitness can be enormously expensive from a computational standpoint. In these cases, It is vital to build a model that approximates fitness and perform research on possibility of approximation [1–3]. In this paper, we used three machine learning algorithms to make approximate models for four problems. Various problem sizes from small to large ones were considered in our experiments. The findings showed that it is possible to train machine learning algorithms with the sampled solution when problem size is large, if there is a possibility of fitness approximation at small problem sizes. The rest of this paper is organized as follows. Three approximate models are presented in Section 2. Test problems are introduced in Section 3. In Section 4, approximation error and performance are presented. Conclusions are drawn in Section 4.

GECCO '18 Companion, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5764-7/18/07.
https://doi.org/10.1145/3205651.3208773

2 APPROXIMATE MODELS

The algorithms used to make a model are linear regression (LR), support vector regression (SVR), and deep neural networks (DNN). In our experiments, "LinearRegression" and "SMOreg" algorithm of WEKA[1] was used. TensorFlow[2] was used to make a neural network model.

3 TEST PROBLEMS

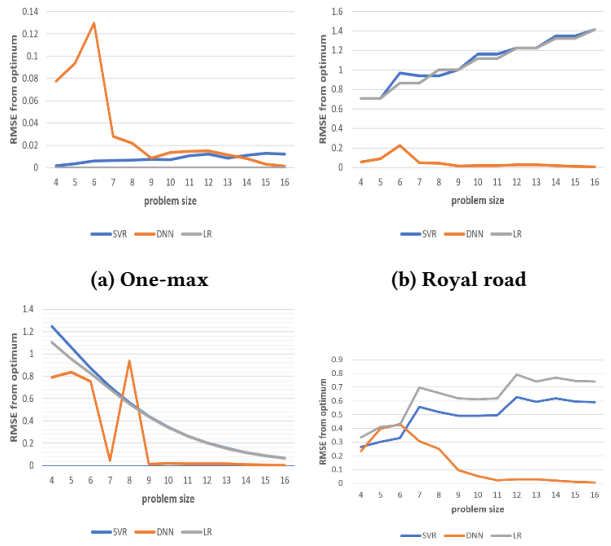
The one-max problem is to maximize the number of genes each of which has a value of 1. The royal road function was used to examine how the schema is processed in the evolutionary process of genetic algorithms. The objective function can be defined as $f(x_1, x_2, ..., x_n) = \sum_i c_i \sigma_i(x_1, x_2, ..., x_n)$ in which c_i is a constant corresponding to schema s_i.$\sigma_i : \{0, 1\} \rightarrow \{0, 1\}$ is a function where 1 was returned if the solution was included in the schema s_i, and 0 was returned if the solution was not included in the schema. In the experiment, c_i was fixed as 2.

An *NK*-landscape model was constructed to define a fitness function with various dimensions and epistasis. The fitness function is tuned by two parameters n and k, where n defines the dimensions of the problem space, and k determines the degree of epistasis between the genes making up chromosomes. The value of k was set to 2 in our experiments. The deceptive problem is designed to force a genetic algorithm to converge to a local optimum rather than a global optimum.

4 EXPERIMENTS AND ANALYSIS

To compare the performance of four problems, the parameters were identically designed. All of the problems used tournament selection and one-point crossover. Bitwise mutation was used with probability of 0.7, and a generational genetic algorithm was used. The population size and number of generations was 50. The performance of the approximated and calculated objective functions were evaluated using an average value with $n = 50$. In small problem size, The root mean square error (RMSE) between the value predicted by the approximate model trained with all solutions and the optimal fitness was calculated. The results are shown in Figure 1. Based on these findings, we can arrive at the conclusion that when the problem size is small, a model trained with all solutions can be considered as a reasonable approximate.

[1] WEKA (https://www.cs.waikato.ac.nz/ml/weka)
[2] TensorFlow (https://www.tensorflow.org)



(a) One-max (b) Royal road

(c) Deceptive (d) *NK*-landscape

Figure 1: Approximation error at small problem sizes ($n \leq 16$)

In large problem size, The RMSE between the value predicted by approximate model and the optimal fitness are shown in Figure 2. The performance comparison between the approximate models and actual calculations are shown in Table 1. A total of 10,000 solutions were arbitrarily selected, and RMSEs derived from 5-fold cross validation were compared. The RMSE of the deceptive problem was not zero for the small problem size ($n \leq 16$), while RMSE was 0 when problem size grew larger, indicating that sampled solutions did not include the optimal solutions. When the calculated and approximated objective functions are compared, the performance was similar in most cases. Surprisingly, the approximation approach was even more effective in the deceptive problem. In order to determine statistical significance, t-tests were conducted and the result was shown to be statistically significant. After comparing the approximation models, DNN showed better performance in the *NK*-landscape problem than LR and SVR, which is similar to the case when DNN had the smallest RMSE when the problem size was small ($n \leq 16$). In the other problems, DNN was expected to have outstanding performance. But as the other problems were considered rather easy, the differences were not conspicuous.

5 CONCLUSIONS

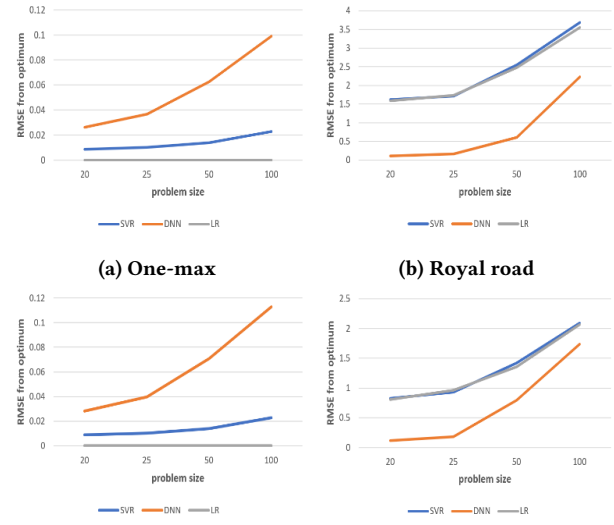
In this study, we confirmed that probability of approximation at small problem sizes and performance at large problem sizes are highly correlated. So it was possible to train machine learning algorithms with the sampled solutions when problem size is large if there is possibility of fitness approximation at small problem sizes. In future research, additional experiments will examine real-world problems with complicated fitness computation

ACKNOWLEDGMENTS

This research was supported by a grant [KCG-01-2017-05] through the Disaster and Safety Management Institute funded by Korea Coast Guard of Korean government.

REFERENCES

[1] Y. Jin. 2005. A comprehensive survey of fitness approximation in evolutionary computation. *Soft computing* 9, 1 (2005), 3–12.
[2] Y. Jin, H. Michael, O. Markus and B. Sendhoff. 2005. Neural networks for fitness approximation in evolutionary optimization. In *Knowledge Incorporation in Evolutionary Computation*. 281–306.
[3] Y. Jin, M. Olhofer and B. Sendhoff. 2002. A framework for evolutionary optimization with approximate fitness functions. *IEEE Transactions on evolutionary computation* 6, 5 (2002), 481–494.



(a) One-max (b) Royal road

(c) Deceptive (d) *NK*-landscape

Figure 2: Approximation error at large problem sizes ($n > 16$)

Table 1: Comparison of performance

Problem	Objective function	SVR approximation		DNN approximation		LR approximation	
	Ave/SD	Ave/SD	p val	Ave/SD	p val	Ave/SD	p val
Onemax_20	19.860/0.351	19.940/0.240	1.89e-01	19.980/0.141	2.92e-02	19.860/0.351	1.00e+00
Onemax_25	24.540/0.613	24.820/0.388	8.76e-03	24.760/0476	5.06e-02	24.540/0.613	1.00e+00
Onemax_50	44.400/1.726	45.380/1.665	5.69e-03	45.580/1.527	6.85e-04	44.400/1.726	1.00e+00
Onemax_100	77.680/3.074	80.180/2.529	4.95e-05	79.220/3.190	1.75e-02	77.680/3.074	1.00e+00
Royal_20	19.740/0.527	19.980/0.141	3.10e-03	19.760/0.517	2.92e-02	19.920/0.274	3.71e-02
Royal_25	24.320/0.913	24.140/0.783	2.95e-01	24.080/0.853	5.06e-02	24.360/0.749	8.12e-01
Royal_50	44.680/1.647	45.320/1.609	5.49e-02	43.900/2.541	6.85e-04	45.320/1.755	6.59e-02
Royal_100	77.800/3.213	79.100/3.151	4.64e-02	78.620/2.806	1.75e-02	79.740/2.791	2.23e-03
Deception_20	18.840/0.442	18.940/0.240	1.51e-01	18.880/0.435	8.49e-01	18.960/0.198	7.46e-02
Deception_25	23.400/0.904	23.820/0.388	3.97e-03	23.740/0.443	1.81e-01	23.840/0.370	2.48e-03
Deception_50	43.640/3.373	44.600/1.552	7.35e-02	43.800/1.485	7.46e-02	44.420/1.762	1.54e-01
Deception_100	75.600/5.588	78.580/2.400	1.10e-03	79.260/3.069	1.80e-01	77.400/3.245	5.44e-02
NK_20_2	13.781/0.725	13.393/0.523	3.41e-03	14.380/0.527	6.43e-01	13.414/0.614	8.64e-03
NK_25_2	17.035/0.773	17.269/0.715	1.22e-01	17.760/0.519	2.07e-02	17.250/0.673	1.44e-01
NK_50_2	32.683/1.360	31.626/1.364	3.07e-04	33.033/1.244	7.6e-01	31.058/1.209	7.16e-08
NK_100_2	60.300/1.761	56.658/2.110	1.37e-12	59.173/2.302	1.73e-04	56.822/2.153	8.50e-12

* SD : standard deviation