DISTINGUISH HARD INSTANCES OF AN NP-HARD PROBLEM USING MACHINE LEARNING

ZHE WANG, TONG ZHANG AND YUHAO ZHANG

ABSTRACT. Graph properties suitable for the classification of instance hardness for the NP-hard MAX-CUT problem are investigated. Eigenvalues of the adjacency matrix of a graph are demonstrated to be promising indicators of its hardness as a MAX-CUT instance.

1. Introduction

Many combinatorial optimization problems arising in various areas are NP-hard in computational complexity. They are commonly believed to be intractable, given that all previous attempts of finding a polynomial time algorithm have failed irrespective of a considerable amount of effort. A typical strategy for tackling these problems is to use approximation or heuristic algorithms to find solutions relatively close to the optimal value. These algorithms usually demonstrate reasonable performance on realistic applications. However, without the knowledge of the hardness of these applications, it is questionable whether identical performance would be achieved in general .

This project is concerned with identifying hard instances of an NP-hard problem according to some features calculable in polynomial time using machine learning algorithms. The problem under study is the MAX-CUT problem. Given an unweighted graph $G = (V, E)$, where V and E denote the set of vertices and edges respectively, the MAX-CUT problem is to find a cut $(S, V \setminus S)$ such that the number of edges with one endpoint in S and the other in $V \setminus S$ is maximized over all possible cuts [1]. The best-known approximation algorithm for the MAX-CUT problem utilizes semidefinite programming and establishes the celebrated 0.87856 performance guarantee [2]. In practice, many heuristic algorithms based on randomized local search achieve better performance than the 0.87856-approximation algorithm [3, 4]. Since local search is subject to being trapped into local optima, it is intuitively plausible to consider instances with a large ratio of the number of local optima to global optima to be hard but those with a small ratio to be easy. However, since the computation time of directly evaluating the ratio increases exponentially with the problem size, it requires alternative efficient methods to estimate the instance hardness for large problems.

The paper is structured as follows. Section 2 describes the data sets under study and explains the method to determine the hardness of the MAX-CUT instances. Section 3 explores possible intrinsic properties of the graph to be utilized as indicators of its hardness. Section 4 presents the results of predicting instance hardness using the eigenvalues of the adjacency matrix of a graph. Finally, section 5 concludes the paper with remarks on possible future directions of the work.

2. Data Sets

The current study focuses only on cubic graphs for the following reasons. First, the MAX-CUT problem restricted to cubic graphs is itself NP-hard [5]. Therefore, hard instances of the problem are still included in the select data sets. Second, the existence of common attributes among cubic graphs easily excludes some properties as promising features. For example, since all bridgeless cubic graphs contain a perfect matching according to Petersen's theorem [6], the matching number is unlikely to be meaningful for the prediction of instance hardness. Third, finite number of graphs of the same order helps reduce the bias and the required number of instances in the training sets.

Table 1. Number of cubic graphs [7].

Order	4	6	8	10	12	14	16	18	20	22
Cubic Graphs	1	2	5	19	85	509	4060	41301	510489	7319447

The hardness of each instance is determined according to its ratio of the number of locally maximal cuts to globally maximal cuts. For a cubic graph of order n, it requires to exhaust $\mathcal{O}(2^n)$ number of possible cuts by brute force to obtain the ratio. Meanwhile, the total number of cubic graphs increases super-exponentially with the graph order, as shown in Table 1. Therefore, only graphs of orders up to 20 are examined due to the limited computational resources.

Since the ratio tends to increase with the graph order, the hardness is relatively measured within instances of the same order. The K-means method are utilized to divide the instances into two clusters, and those assigned to the centroid with a larger ratio are labeled as "hard" while the remains are considered to be "easy". Fig.1 displays the classification result for graphs of order 18 and the shift of the cluster centroid positions for different graph orders. The decision boundary is fitted well by an exponential curve, which captures the exponentially faster increase of the number of local optima compared to global optima with the graph order on average. If the fitted curve already indicates the true asymptotic behavior of the decision boundary, it provides information on the estimated lower bound for the number of local optima for hard instances of arbitrary order.

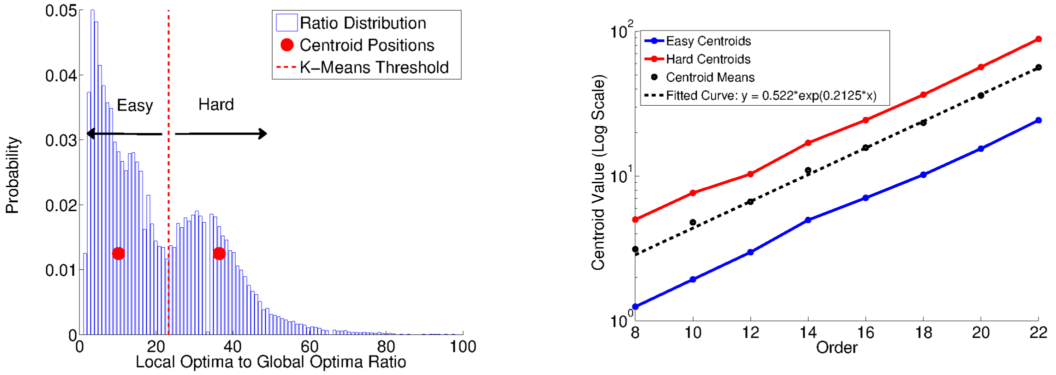

Figure 1. Left: probability distribution of the ratio of the number of local optima to global optima on cubic graphs of order 18; Right: positions of the cluster centroids and decision boundary for each graph order. The data points for order 22 are results of 5000 random cubic graphs.

3. Feature Selection

The possibility of using a particular graph property as a feature for the classification of instance hardness has been analyzed through calculating the mutual information between it and the instance hardness label, which is given by

$$\mathrm{MI}(x,y) = \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)}.$$

Three efficiently calculable properties of graphs have been investigated: global clustering coefficient (GCF) [8], diameter and eigenvalues of the adjacency matrix of the graph. Even though a highly-clustered graph is intuitively more difficult to solve than a sparsely-connected graph, the mutual information between GCF and the hardness label is almost 0. Thus, it is highly unlikely to be a relevant feature. Diameter is also excluded as a suitable feature for the same reason.

On the other hand, non-trivial mutual information is discovered between the eigenvalues of adjacency matrices of the graphs and the hardness label, as shown in Fig.2. A cubic graph of order n has eigenvalues $3 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq -3$. All of them may potentially function as useful features except for the largest eigenvalue λ_1 since it is a constant for all instances. The relatively higher mutual information achieved by λ_2 and λ_n than the remains can be explained from their relations to the structures of the graph. The edge expansion of a graph is bounded by functions of λ_2 as shown in Cheeger's inequality [9], and the bipartiteness ratio of a graph is in close relation with λ_n [10]. In addition, an upper bound of the maximum cut for a d-regular graph is given by $(1 + |\lambda_n|/d)nd/4$ [11].

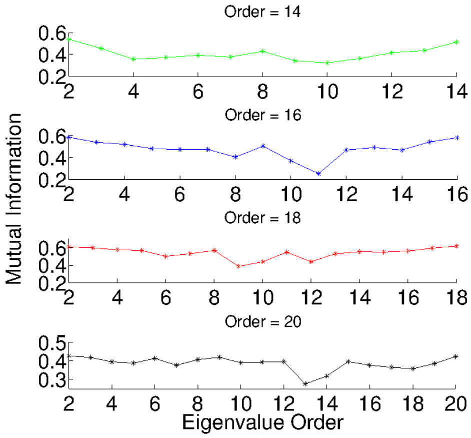

FIGURE 2. Mutual information between the eigenvalues of the cubic graphs and the hardness label.

4. Classification Results

Machine learning algorithms such as logistic regression, naive Bayes and support vector machine (SVM) have been applied to predict instance hardness using the eigenvalues of the adjacency matrix of a graph. The SVM algorithm with the gaussian kernel

$$K(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{||\mathbf{x} - \mathbf{z}||^2}{2\sigma^2}\right)$$

yields the best classification results. Fig.3 visualizes the outcomes of using the second largest eigenvalue λ_2 and the smallest one λ_n to categorize the hardness of instances within the same order. It can be easily seen that hard and easy instances appear to form clusters on the plane defined by λ_2 and λ_n. Meanwhile, the clusterings for graphs of different orders and decision boundaries are similar in some degree, which suggests the possibility of using learning models obtained from graphs of lower orders to predict the hardness of instances of higher orders.

To evaluate the effectiveness of using eigenvalues to identify hard instances, training sets consisting of 50% hard and 50% easy instances are randomly generated for cubic graphs of order from 16 to 20. It is verified that larger training sets only produce very limited improvements on accuracy, therefore, training sets with 1000 instances for graphs of order 16, and 4000 instances for graphs of order 18 and 20 are mainly used for the investigation.

The performance of the SVM algorithm is tuned by adjusting the number of features and the parameter σ in the gaussian kernel. Eigenvalues at the head and tail of the sequence are mainly considered because they have relatively larger mutual information with the hardness label. If the

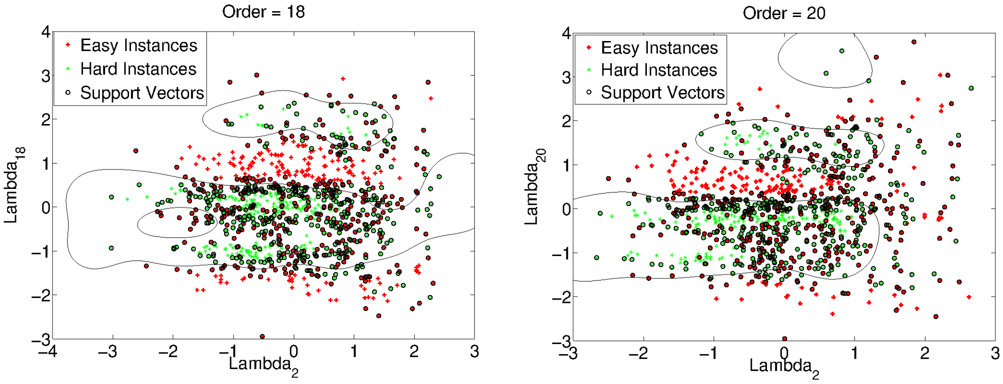

FIGURE 3. SVM classification results for randomly selected 500 hard and 500 easy instances of order 18 and 20 using λ_2 and λ_n when $\sigma = 0.8$ and the cost constant $C = 1$. The eigenvalues are normalized to quantities with mean 0 and variance 1.

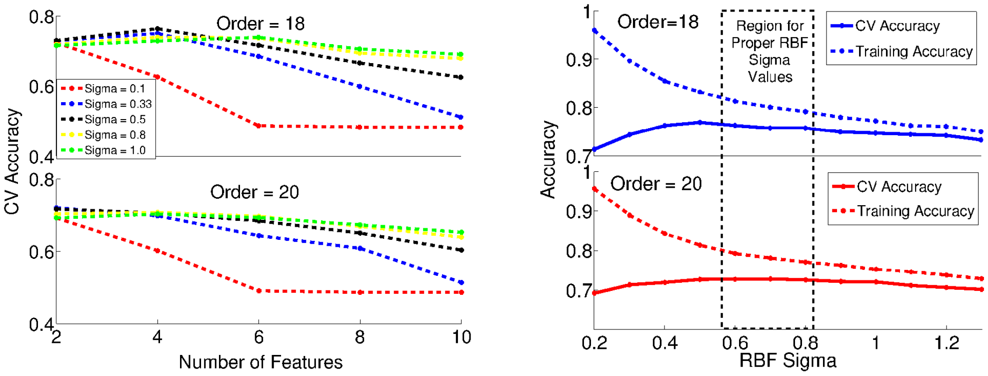

FIGURE 4. Prediction accuracy using different number of features and σ's in the gaussian kernel. The values are obtained by performing k-fold cross validation on training sets consisting of 4000 randomly selected cubic graphs of the same order when $k = 5$.

number of features are determined to be 2, λ_2 and λ_n will be utilized, and if the number is decided to be 4, λ_2, λ_3 and λ_{n-1} and λ_n will be selected, and so on. Fig.4 shows that the algorithm performs reasonably well with a small number of selected features and σ in the interval $[0.4, 0.8]$. The prediction accuracy decreases with the number of eigenvalues, which might be caused by the fact that data points appear to be sparse in a higher dimensional space. Meanwhile, it can also be observed that when σ is chosen to be too small, even though the learning model can achieve above 90% accuracy on the training sets, it introduces large generalization error due to overfitting. On the other hand, when σ is too large, the model becomes not accurate enough even on the training sets themselves.

Table 2 lists the prediction accuracy of the learning models on instances of both the same and different orders as those in the training sets. In the former case, the accuracy is approximately 75% on all randomly selected training sets. Although the absolute value does not appear so attractive, the achieved result is still considered to be reasonably good given that eigenvalues and the ratio of the number of local optima to global optima are two seemingly uncorrelated quantities. Meanwhile, it is not surprising that the accuracy decreases when learning models obtained from training sets of other orders are utilized. As demonstrated in Fig.3, the decision boundaries seem to be shifting

TABLE 2. Prediction accuracy of the SVM algorithm with the gaussian kernel using 4 eigenvalues as features when $\sigma = 0.6$ and $C = 1.0$. The same-order prediction results are the average accuracy obtained from the training sets using k-fold cross validation, whereas the cross-order prediction uses the learning model obtained from the training set of order 18 to classify instances in training sets of other orders.

Instance Order	16	18	20
Same-order prediction	0.7460	0.7630	0.7270
Cross-order prediction	0.6110	-	0.6150

slightly for different orders. It requires more work in the future to understand the variation of the boundaries in order to improve the cross-order prediction accuracy.

5. Summary

Eigenvalues of the adjacency matrix of a graph are discovered to be promising features to classify instance hardness for the MAX-CUT problem. Hard and easy instances are shown to form clusters in the space spanned by the eigenvalues. For the studied instances, the learning models based on the SVM algorithm can achieve up to about 75% accuracy in predicting the hardness of instances of the same order. Since the clusterings are similar for different graph orders, it is expected that better understanding of the decision boundaries will help increase the accuracy of cross-order prediction.

References

[1] R. M. Karp, in *Complexity of computer computations*, edited by R. E. Millera and J. W. Thatcher (Plenum Press, 1972) pp.85-103.

[2] M. X. Goemans and D. P. Williams, J. of the ACM **42**, 1115-1145 (1995).

[3] S. Burer, R. D. C. Monteiro, and Y. Zhang, SIAM J. on Optimization **12**, 503-521 (2001).

[4] P. Festa, P. M. Pardalos, M. G. C. Resende, and C. C. Ribeiro, Optim. Methods Software **7**, 1033-1058 (2002).

[5] M. Yannakakis, in *Proceedings of the tenth annual ACM symposium on Theory of Computing* (IEEE Computer Society, San Diego, CA, 1978) pp.253-64.

[6] J. Petersen, Acta Mathematica **15**, 193220 (1891).

[7] G. Royle, http://www.easybib.com/reference/guide/mla/website (Oct. 1996)

[8] Duncan J. Watts and Steven H. Strogatz, Nature **393** 440-442(1998).

[9] N. Alon, Combinatorics, **6**(2), 83-96 (1986)

[10] L. Trevisan, in *Proceedings of the 41st ACM Symposium on Theory of Computing* (New York, NY, 2009) pp. 263-272.

[11] C. Delorme and S. Poljak, Mathematical Programming **62**, 557-574 (1993)