深層強化学習によるナップサック問題の解法に関する研究

A Study on Solving Knapsack Problem by Deep Reinforcement Learning

末吉　璃子・システム分科会・情報セキュリティ大学院大学

This study focuses on the Knapsack problem, which is used for trapdoors in Knapsack cryptography, and proposes an architecture for solving the problem using deep reinforcement learning. Various solution methods have been proposed in the past, but in recent years, with the development of machine learning, there has been an increase in the number of studies applying deep reinforcement learning to optimization problems. In this study, we propose an architecture for solving trapdoors in KnapSack cryptography based on the work of Goole Brain, which applied a Pointer network to the traveling salesman problem (TSP). Specifically, we consider random, hyper-increasing, even-odd, composite, and modulo transformed trapdoors and obtain approximate and exact solutions, and compare the exact solution with the LLL algorithm. The results show that reinforcement learning slightly detects trapdoors in the approximate solution. Furthermore, the exact solution was able to solve up to 30 dimensions, and depending on the problem, it was found to be more accurate than the LLL algorithm. In addition, we found that training on a more difficult problem than the test problem in the exact solution improves the test results.

研究の背景 ・ 目的

■近年、深層強化学習を用いて最適化問題を解く研究が増えている。中でも、Knapsack問題（NP完全問題）を深層強化学習を用いて解く手法も提案されている。(先行研究①)

■暗号におけるKnapsack問題を解く意義

・Knapsack問題を始めとする組み合わせ最適化問題は耐量子計算機暗号の安全性を支える重要な問題であり、LLLアルゴリズム以外の別のアプローチも必要。

■本研究では、Pointer networkを用いて(村上ら(先行研究②))が提案したトラップドア(超増加性、偶奇性、複合)の近似解を求め(トラップドアを見破っているかどうかの検証も)、さらにそのトラップドアの厳密解を求める。

先行研究/提案手法

■先行研究①

NEURAL COMBINATORIAL OPTIMIZATION WITH REINFORCEMENT LEARNING (Google Brain 2017)

・2017年にGoogle BrainでTSPを解くためのPointer Networkと強化学習(Actor-Critic)を用いたアーキテクチャを提案

・さらにSampling searchとActive searchという厳密解を求める手法を提案

→Active search は高い性能を示している

■先行研究②

トラップドア複合型高密度ナップザック暗号の提案

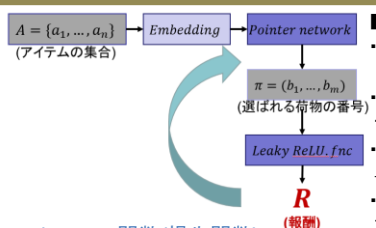
(名迫、村上 2006年 日本応用数理学会)

・Knapsack暗号について、従来の超増加性トラップドアに加え新しい偶奇性、複合トラップドアを提案

超増加性＋偶奇性　↓トラップドアの隠蔽

複合トラップドア(秘密鍵) ＋ モジュラ変換 ⟶ よりセキュアな難しい問題(公開鍵)

提案手法



■提案手法

・google brainの研究をベースに

・Pointer Network を用いてナップサックに入れるべき荷物を選ぶ

・選ばれた荷物集合に報酬を設定 →強化学習

・報酬の計算はLeakyReLU関数で行っている

LeakyReLU関数(損失関数)

・もし荷物の総量を超えたらハイパーパラメータ×荷物の総量のペナルティを課す。

$$L(\pi) = -R(\pi) = |LeakyReLU(W - \Sigma_{i=1}^{m} a_{\pi(i)}, \alpha = 2.0))|$$

■問題設定

・ランダム/モジュロ複合/モジュロ偶奇性/モジュロ超増加性/複合/偶奇性/超増加性

近似解の結果

■25/50,50/100,150/300で実験を行った150/300での結果

・複合、超増加性は最適値(1.0)から離れた値に

・モジュラ変換を施した問題は最適値(1.0)に近い

→モジュラ変換を施す事は強化学習にとって有利に働く可能性

■トラップドアを見抜いているかどうか

モジュロ超増加性/モジュロ偶奇性/モジュロ複合においてモジュラ変換ありで学習とランダムデータで学習した場合を比較。赤字が最適値(1.0)に近い事を示す。モジュロ超増加性の結果を示す。

モジュロ超増加性トラップドア			
		モジュロ変換で学習	ランダムで学習
50/100	1.50515	0.999989935	0.999977026
	0.9996	1.000007629	0.999759085
150/300	1.50515	1.00002497	0.999807421
	0.9924	0.998988203	0.99972271

・モジュロ偶奇性、モジュロ複合では150/300ではモジュラ変換を施した方が良い結果となった。→モジュロ超増加性と高次元ではトラップドアを見抜いている

厳密解の結果

	問題	密度	荷物の範囲	Sampling 正解率	Active 正解率	LLL 正解率
ランダム	20_10	1.505149978	10000	97%	64%	19%
		1.003433319	1000000	81%	8%	3%
		0.974823664	15×10^5	98%	17%	1%
	30_15	1.505149978	1000000	5%	1%	5%
		1.003433319	1000000000	0%	0%	1%
		0.903089987	1E+10	0%	0%	0%
超増加性	20_10	1.505149978	10000	10%	5%	21%
		1.003433319	1000000	12%	2%	1%
		0.974823664	15×10^5	2%	4%	0%
	30_15	1.505149978	1000000	1%	0%	9%
		1.003433319	1000000000	0%	–	0%
		0.903089987	1E+10	0%	0%	0%
ュロ超増	20_10	1.505149978	10000	7%	3%	23%
		1.003433319	1000000	1%	3%	3%
		0.974823664	15×10^5	5%	1%	3%
	30_15	1.505149978	1000000	2%	–	6%
		1.003433319	1000000000	0%	–	0%
		0.903089987	1E+10	–	–	1%

■ランダム/超増加性/モジュロ超増加性で10/20,15/30で実験を行った。

・正解率は100回中何回正解だったかを表している

・厳密解は30次元密度1.0台まで解く事が出来た。

・LLLアルゴリズムとの比較では、基本的に高次元低密度ではLLLアルゴリズムよりSampling searchやActive searchが低い結果となった。

■学習時にテストよりも大きな問題でトレーニングする事で大幅に性能が上がる事も分かった。

	テスト問題	密度	使用したモデル	Sampling search	Active search	LLL_Algorithm
ランダム	20_10	0.974823664	ランダム30_15 密度1.50515	92%	13%	1%
超増加性	20_10	0.974823664	超増加性30_15 密度1.50515	6%	4%	0%
モジュロ超増加性	20_10	0.974823664	モジュロ超増加性30_15 密度1.50515	72.00%	13%	3%