情報処理学会論文誌 Vol.60 No.2 651–659 (Feb. 2019)

深層学習を用いた巡回セールスマン問題の解法

三木 彰馬[1,a)] 榎原 博之[2,b)]

受付日 2018年2月28日, 採録日 2018年11月7日

概要：本論文では代表的な組合せ最適化問題の1つである巡回セールスマン問題（TSP）に注目し，深層学習を適用した解法を提案する．本手法では，畳み込みニューラルネットワークを用いて最適経路を画像として学習することで，最適経路に含まれうる辺の分布である優良エッジ分布を求め，これにより計算される辺の評価値である優良エッジ値を利用して近傍探索を行う．この提案手法の性能を調べるために実験を行い，解の精度向上において有効であることを示す．

キーワード：組合せ最適化問題，巡回セールスマン問題，深層学習，畳み込みニューラルネットワーク，近傍探索法

Solving Combinatorial Optimization Problems Using Deep Learning

Shoma Miki[1,a)] Hiroyuki Ebara[2,b)]

Received: February 28, 2018, Accepted: November 7, 2018

Abstract: In this paper, we focus on the traveling salesman problem (TSP) that is a typical combinatorial optimization problem, and propose a method for solving it with applying deep learning. This method features learning the image of the optimal tour by a convolutional neural network to acquire the Good-Edge Distribution whose edges could be included in the optimal solution. It also conducts neighborhood search by using Good-Edge Value that is an evaluation of each edge calculated from the distribution. We show experimentally that this method improves the quality of solutions.

Keywords: combinatorial optimization problem, traveling salesman problem, deep learning, convolutional neural network, neighborhood search

1. はじめに

組合せ最適化問題は計算機科学における基本的な問題の1つである．輸送や通信，製造，インフラ計画など，さまざまな分野における多くの課題は組合せ最適化問題として扱うことができ，現実社会での応用が期待されている．典型的な組合せ最適化問題の1つに巡回セールスマン問題（TSP: Traveling Salesman Problem）があげられる．TSPとは与えられたグラフにおいて，すべての頂点を1度だけ通るような巡回路のうちエッジ（辺）の距離の総和を最小とするものを求める問題である．とくに頂点が2次元平面上にあり，エッジの距離が頂点間のユークリッド距離として定義されるものを平面TSPと呼ぶ．TSPとその派生問題の応用として，荷物の配送計画や基板のドリル穴あけ問題などが考えられている．TSPは頂点数が増えるにつれて解の個数が指数関数的に増加し，また，NP困難に属する．

組合せ最適化問題の解法はその解の精度の違いによって厳密解法，近似アルゴリズム，ヒューリスティクスに大きく分けられる．厳密解法では列挙法や分枝限定法などを用いて最適解を求めるが，問題の規模が大きい場合には現実的な時間で最適解を求めることができない．また近似アルゴリズムでは解の近似度が保証されるが，経験的な知識と複雑な計算を要するため，良い近似度を保証する高速な近似アルゴリズムを得ることは難しい．一方ヒューリスティ

[1] 関西大学大学院
Graduate School of Kansai University, Suita, Osaka 564–8680, Japan
[2] 関西大学
Kansai University, Suita, Osaka 564–8680, Japan
[a)] k154911@kansai-u.ac.jp
[b)] ebara@kansai-u.ac.jp

クスでは解の精度が保証されないが短い時間で解を求められる場合が多く，その例として LK 法 [1] などの近傍探索法や遺伝的アルゴリズム [2] があげられる．これらの背景から，高速かつ高精度なヒューリスティクスの開発が重要視されている．

近年，機械学習および深層学習を用いた技術が活発に研究され，今までは困難であった課題を解決できる可能性がある手法として注目されている．深層学習では学習に長い時間を費やすことで問題の特徴量を事前に獲得し，高精度かつ高速な近似を実現する．このことから，組合せ最適化問題の解法において深層学習を利用することで，未知の入力への汎化と計算時間の削減ができるのではないかと考えられる．

畳み込みニューラルネットワーク（CNN: Convolutional Neural Network）は畳み込み演算を行う処理層で構成されるニューラルネットワークであり，画像認識をはじめ画像を入力とする問題において高い性能を示している．その例として Deep Convolutional Generative Adversarial Network（DCGAN）を用いた画像生成 [3], [4] などがあげられる．また囲碁 AI への応用例 [5] では，盤面の情報を CNN に入力し，盤面上の各位置の手と局面に対する評価値を近似する．これによりゲーム木の探索における評価関数の先読みを高速化し，人間のプロ棋士と同等またはそれ以上の棋力を実現している．

TSP を含む組合せ最適化問題に対して深層学習を適用する手法はいくつか研究されている．再帰型ニューラルネットワークを用いる手法 [6], [7] では，問題を構成するデータ系列を LSTM（Long Short-Term Memory）に入力することで特徴量を圧縮し，次に選ぶべき入力要素を順番に評価することで解を構築する．問題が持つグラフ構造を利用した手法 [8] では，グラフに従ってネットワークの処理層を結合し，隣接する頂点間で特徴量を伝播させることで，グラフ構造を効率良く反映した特徴量の抽出を図っている．しかしこの手法はすべての頂点が互いに隣接する TSP では精度の高い特徴抽出が難しく，特徴量を伝播する近傍を制限するなどの工夫が必要である．

本論文では平面 TSP に注目し，CNN を用いてエッジの評価値を計算する手法と，2-opt 法において距離の代わりにエッジの評価値を用いる手法 EV-2opt を提案し，それらの提案手法の性能を検証するために実験を行う．この手法では，TSP とその解を画像として扱い，CNN により最適経路の画像を近似した優良エッジ分布と，ここから得られる各エッジの評価値として優良エッジ値を計算する．そして，TSP に対する手法の 1 つである 2-opt 法において距離の代わりに優良エッジ値を使用し，経路上の優良エッジ値の総和が大きくなるようにエッジをつなぎかえることによって近傍探索を行う．実験では優良エッジ分布を学習し，EV-2opt を用いた解法の精度について調べる．

2. CNN を用いたエッジの評価

2.1 優良エッジ分布

平面 TSP では各頂点が 2 次元ユークリッド座標で与えられ，点および線の描画によってその問題例と解を画像として表現できる．ここである平面 TSP の問題例について，すべての頂点を描画した頂点画像 $N(x, y)$ と，その最適経路を描画した最適経路画像 $t(x, y)$ を定義する．ただしこれらの画素数を (S_1, S_2) とし，画像内の任意の画素の位置 (x, y) は $(x, y) \in \{1 .. S_1\} \times \{1 .. S_2\}$ を満たす．また，描画前の画素値を 0 とし，最大画素値 1 で点および線を描画する．

頂点画像 $N(x, y)$ を入力したときその問題例の最適経路画像 $t(x, y)$ を出力するようなモデルを考える．このよう

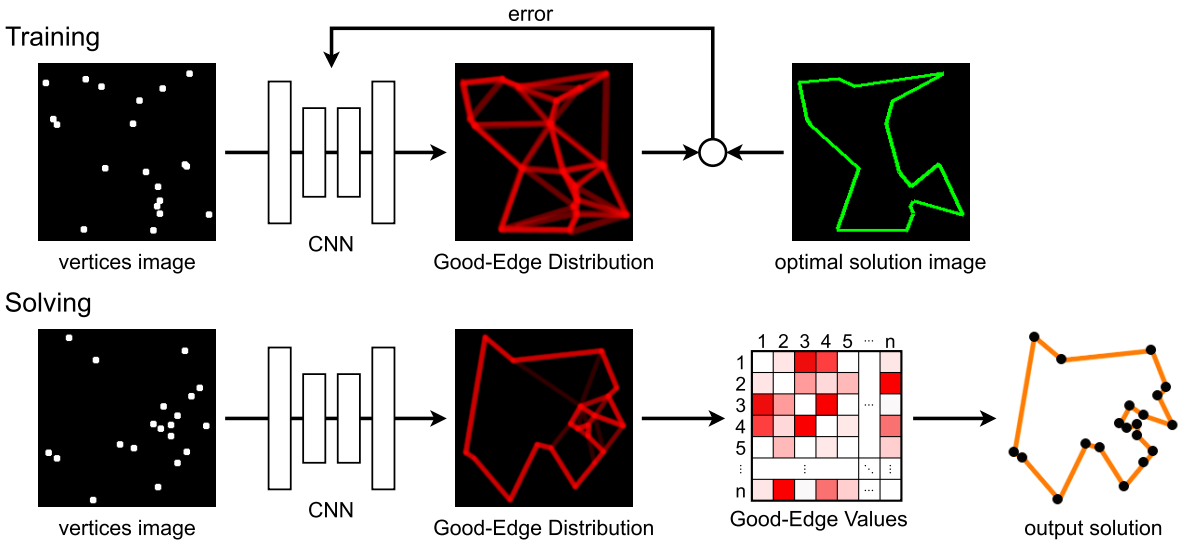


図 1 優良エッジ分布の学習方法と解法の概要

Fig. 1 Overview of the training and solving method with Good-Edge Distribution.

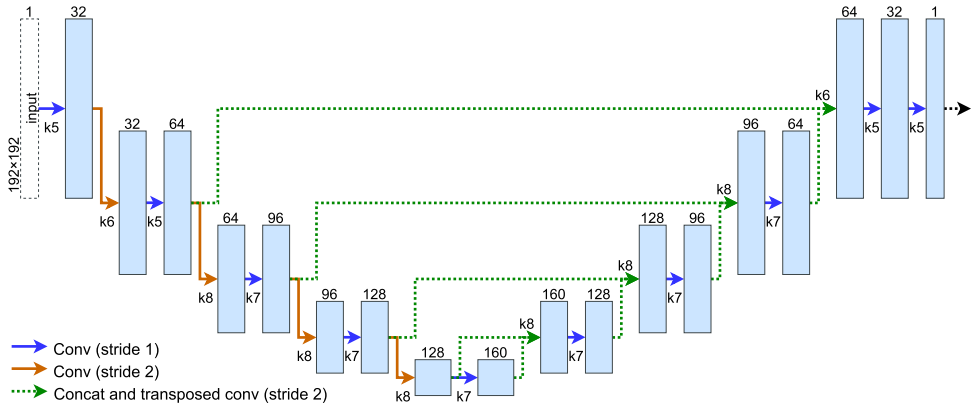


図 2 CNN の構造
Fig. 2 Structure of CNN.

なモデルが得られれば，その出力画像に表されたエッジを選ぶことで最適経路を求めることができる．提案手法では CNN を用いることでこのモデルを近似し，その出力 $p(x,y)$ を優良エッジ分布（Good-Edge Distribution）と呼ぶ．優良エッジ分布の学習方法とこれを用いた解法の概要を**図 1** に示す．

優良エッジ分布はその出力 $p(x,y)$ を教師信号 $t(x,y)$ に近づけることを目的として学習を行う．1 つの問題例について式 (1) のように 2 乗誤差で表される損失関数を与え，これを最小化するように勾配降下法および誤差逆伝播法を用いて CNN の重みを更新する．

$$\text{loss} = \sum_{x=1}^{S_1}\sum_{y=1}^{S_2}(t(x,y) - p(x,y))^2 \tag{1}$$

2.2 優良エッジ値

学習により優良エッジ分布が得られたとき，式 (2) のように各エッジ (i,j) 上における優良エッジ分布の平均を求めることで，そのエッジが最適経路に含まれる尤度を計算する．これを優良エッジ値 v_{ij} と呼ぶ．ただし $l_{ij}(x,y)$ はエッジ (i,j) を描画した画像，τ_{ij} は後述する交差ペナルティである．

$$v_{ij} = \frac{1}{1+\tau_{ij}} \cdot \frac{\sum_{x=1}^{S_1}\sum_{y=1}^{S_2} p(x,y)\, l_{ij}(x,y)}{\sum_{x=1}^{S_1}\sum_{y=1}^{S_2} l_{ij}(x,y)} \tag{2}$$

交差ペナルティ $\tau_{ij} \in \mathbb{R}_{\geq 0}$ はエッジ (i,j) がそれ以外のいくつの頂点と画像上で重なっているかを表す指標であり，他の頂点とまったく重なっていないとき 0，他の k 個の頂点のみと完全に重なるとき k の値となる．また，画像上で部分的に重なる（エッジの線分が中心を通過しない）ような頂点が存在する場合，交差ペナルティの値域は非負の実数になる．これは頂点集合の画像とエッジ (i,j) の画像について，その画素単位の積の総和を求めることにより

表 1 計算機の構成
Table 1 Machine environment.

項目	概要
CPU	Intel Core i7-6900K CPU @ 3.20 GHz
GPU	NVIDIA GeForce GTX TITAN X
メインメモリ	125.7 GiB
OS	CentOS 7

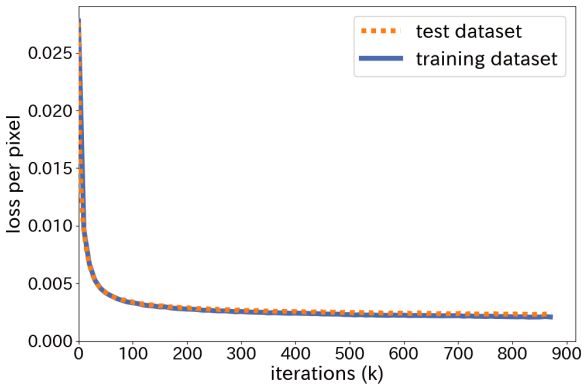


図 3 損失関数の推移
Fig. 3 Average losses for each dataset.

計算する．あるエッジが評価の高い他のエッジと多くの画素を共有している場合，そのエッジを本来よりも高く評価してしまう．そこでこの現象を緩和するために交差ペナルティを導入し，他の頂点との交差を持つようなエッジの優良エッジ値を低く見積もる．

3. 優良エッジ値を用いた近傍探索

優良エッジ値は各エッジが最適解に含まれる尤度であり，優良エッジ値が大きいエッジを選ぶことにより最適経路に近い解が得られると考えられる．そこで TSP の近傍探索法の 1 つである 2-opt 法において距離の代わりに優良エッジ値を用いる手法を提案し，これを EV-2opt 法（Edge Value 2-opt）と呼ぶ．

従来の 2-opt 法では 2 つのエッジをつなぎかえ，経路の

(a) 最適経路
(a) Optimal tour.

(b) 優良エッジ分布
(b) Good-Edge Distribution.

(c) SEP による初期解（誤差率 18.064%）
(c) SEP ($\varepsilon = 18.064\%$).

(d) EV-2opt（誤差率 0.157%）
(d) EV-2opt ($\varepsilon = 0.157\%$).

(e) EV-2opt+2opt（誤差率 0.122%）
(e) EV-2opt+2opt ($\varepsilon = 0.122\%$).

図 4 優良エッジ分布と解の出力例（rand50-A）
Fig. 4 Output of the Good-Edge Distribution and solutions (rand50-A).

(a) 最適経路
(a) Optimal tour.

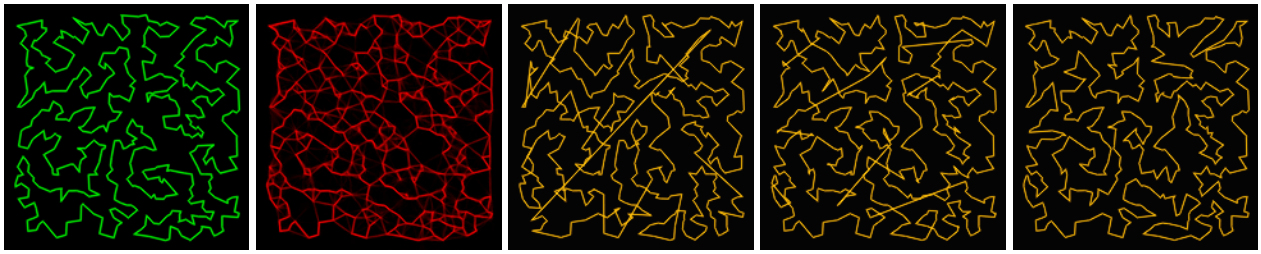
(b) 優良エッジ分布
(b) Good-Edge Distribution.

(c) SEP による初期解（誤差率 23.695%）
(c) SEP ($\varepsilon = 23.695\%$).

(d) EV-2opt（誤差率 9.469%）
(d) EV-2opt ($\varepsilon = 9.469\%$).

(e) EV-2opt+2opt（誤差率 2.818%）
(e) EV-2opt+2opt ($\varepsilon = 2.818\%$).

図 5 優良エッジ分布と解の出力例（rand100-A）
Fig. 5 Output of the Good-Edge Distribution and solutions (rand100-A).



(a) 最適経路
(a) Optimal tour.

(b) 優良エッジ分布
(b) Good-Edge Distribution.

(c) SEP による初期解（誤差率 18.791%）
(c) SEP ($\varepsilon = 18.791\%$).

(d) EV-2opt（誤差率 15.328%）
(d) EV-2opt ($\varepsilon = 15.328\%$).

(e) EV-2opt+2opt（誤差率 5.328%）
(e) EV-2opt+2opt ($\varepsilon = 5.328\%$).

図 6 優良エッジ分布と解の出力例（rand500-A）
Fig. 6 Output of the Good-Edge Distribution and solutions (rand500-A).

総距離が短くなる組合せを選択することで近傍探索を行うが，EV-2opt 法では経路の優良エッジ値の総和が大きくなる近傍解を選択する．ここでは局所探索の戦略として即時移動戦略を採用し，その EV-2opt 法の手順を以下に示す．

(1) すべてのエッジ (i, j) について優良エッジ値 v_{ij} を計算する．
(2) 初期経路を設定する．
(3) 経路中の隣り合わないエッジの組 (i, j), (k, h) のうち，$v_{ij} + v_{kh} < v_{ik} + v_{jh}$ を満たすものを 1 つ探す．なければ現在の経路を出力して終了する．
(4) 2 つのエッジ (i, j), (k, h) を (i, k), (j, h) としてつなぎかえて経路を作成し，現在の経路とする．
(5) (3) に戻る．

4. 評価実験

本章では，提案手法の性能を評価するために実施した計算機実験の方法とその結果について説明する．

実験プログラムのプログラミング言語として Python を，

機械学習用のライブラリとして TensorFlow を使用する．実験に使用する計算機の構成を**表 1** に示す．

4.1 優良エッジ分布の評価

学習に用いる平面 TSP の問題例として，頂点数 20〜100 の問題例を 20 万個，各頂点の座標を一様乱数により設定することで生成し，TSP ソルバー Concorde [9] を用いてその最適解を求める．教師信号および学習する CNN の入出力の画素数は $(S_1, S_2) = (192, 192)$ とし，頂点の描画半径を 1.5，エッジの描画幅を 1 としてそれぞれの問題例の入力信号 $N(x, y)$ と教師信号 $t(x, y)$ を作成する．汎化能力を向上させるため，頂点画像および最適経路画像を 90 度単位で回転させたものを訓練用データに追加し，データ拡張を行う．

優良エッジ分布の学習に使用する CNN の構造を**図 2** に示す．ただし図中において k に続く整数は畳み込みのカーネルサイズを表している．この CNN は U-Net [3], [10], [11] と類似の構造を持ち，画像の縮小を行うエンコーダと拡大を行うデコーダの各層が skip connection によって結合されている．処理層は 14 層の畳み込み層と 4 層の転置畳み込み層で構成され，これらの層のストライドを 2 以上に設定することで画像を縮小および拡大する．出力層以外の層ではスロープ係数 0.2 の Leaky ReLU [12] を，出力層では恒等写像を活性化関数として設定する．学習時のミニバッチ数は 32 とし，学習アルゴリズムには Adam 法 [13] を使用する．

優良エッジ分布の学習時の損失関数の推移を**図 3** に示す．黄破線，青実線はそれぞれテストデータと訓練データのバッチ全体に対する損失関数の平均を表している．今回は実験時間の都合上，ある程度学習が進んだと見なした時点で学習を停止した．学習時間は約 300 時間であり，以降の実験にはイテレーションが 800 k の時点で得られたモデルを使用する．

学習したモデルによって得られる優良エッジ分布と，4.2 節の解の評価実験にて得られた解の出力例を，**図 4**，**図 5**，**図 6** に示す．各図の (a) は与えた問題例の最適経路を，(b) はその問題例に対する優良エッジ分布の出力を表す．(c) から (e) は EV-2opt+2opt の処理過程における解の例を示している．

図 4 (b)，図 5 (b)，図 6 (b) より，優良エッジ分布が最適経路に含まれるエッジを多く持ち，最適経路の概形を表現できていることが分かる．ただし頂点が密集するような領域では，最適ではないエッジ上でも大きな値を出力することが多く，最適解として有効なエッジを絞りきることができていない．これは頂点数が多い問題例に多くみられ，解を求める際に精度を下げる要因となる．この傾向を改善する手法として，より大きな頂点数の問題例を訓練データとして使用することや，データ拡張として問題例をより小さい領域に描画することで細部の近似性能を向上させるなどの方法が考えられる．

表 2 ランダムな問題例に対する平均誤差率（%）
Table 2 Average error ratios for random instances.

問題例	2opt	**EV-2opt**	**EV-2opt+2opt**
rand20-A	1.389	**0.000**	**0.000**
rand20-B	**0.000**	**0.000**	**0.000**
rand50-A	5.207	0.220	**0.170**
rand50-B	2.449	**0.405**	**0.405**
rand75-A	**2.196**	14.779	2.369
rand75-B	7.034	4.115	**0.685**
rand90-A	2.925	1.090	**0.419**
rand90-B	7.680	6.633	**0.993**
rand100-A	5.485	7.584	**2.727**
rand100-B	7.149	**0.022**	**0.022**
rand100-C	5.118	4.732	**1.965**
rand150-A	5.708	3.861	**0.604**
rand200-A	5.015	9.216	**2.620**
rand300-A	4.630	15.269	**4.449**
rand400-A	4.857	13.631	**4.635**
rand500-A	5.277	15.470	**5.261**
総平均	4.508	6.064	**1.708**

表 3 ランダムな問題例に対する平均計算時間（s）
Table 3 Average solving times for random instances.

問題例	2opt	**EV-2opt**	**EV-2opt+2opt**
rand20-A	0.001	0.083	0.083
rand20-B	0.001	0.065	0.065
rand50-A	0.019	0.296	0.298
rand50-B	0.009	0.359	0.360
rand75-A	0.030	0.648	0.685
rand75-B	0.045	0.627	0.637
rand90-A	0.046	0.903	0.913
rand90-B	0.089	0.938	0.964
rand100-A	0.134	1.159	1.196
rand100-B	0.108	1.124	1.131
rand100-C	0.141	1.159	1.197
rand150-A	0.325	2.753	2.875
rand200-A	0.526	4.778	5.141
rand300-A	1.926	11.643	13.787
rand400-A	5.812	25.046	29.733
rand500-A	10.834	37.301	47.479

4.2 解の評価

テスト用問題例について各解法により得られた解の平均誤差率を比較する．EV-2opt 法を利用した提案手法（EV-2opt+2opt と表記）では，まず貪欲法ベースのアルゴリズムによって初期解を生成し，これに対して EV-2opt 法を適用した後，さらに通常の距離を用いた 2-opt 法を適用する．一方，比較手法（2opt と表記）では，初期解に対して距離による 2-opt 法のみを適用する．初期解の生成に用いるアルゴリズムでは，現在の部分解に対して部分閉路を作らな

表 4 TSPLIB の問題例に対する平均誤差率（%）
Table 4 Average error ratios for TSPLIB instances.

問題例	2opt	S2V-DQN	**EV-2opt**	**EV-2opt +2opt**
eil51	3.462	3.052	11.162	**2.958**
berlin52	9.382	**0.000**	0.350	**0.000**
st70	3.993	3.111	1.489	**0.452**
eil76	4.089	4.833	0.372	**0.037**
pr76	4.293	**0.265**	6.648	2.188
rat99	4.013	5.698	5.925	**1.936**
kroA100	2.407	2.890	1.415	**0.477**
kroB100	2.238	2.489	3.426	**0.660**
kroC100	3.162	1.566	4.170	**0.104**
kroD100	5.071	3.794	4.013	**0.986**
kroE100	**2.696**	3.829	8.449	2.708
rd100	4.149	3.148	2.996	**1.320**
eil101	6.065	4.769	5.469	**1.932**
lin105	3.418	4.479	5.709	**0.807**
pr107	1.943	1.828	8.474	**1.750**
pr124	2.698	4.393	7.042	**2.506**
bier127	6.416	2.785	8.553	**2.681**
ch130	4.865	**2.619**	9.814	3.205
pr136	9.426	2.792	10.905	**2.221**
pr144	1.507	1.536	20.047	**1.198**
ch150	7.027	7.001	7.354	**2.500**
kroA150	3.579	5.143	10.163	**2.672**
kroB150	5.987	4.129	5.678	**2.577**
pr152	3.036	2.173	8.800	**1.254**
u159	5.617	7.968	0.291	**0.107**
rat195	**5.390**	11.106	15.189	5.568
d198	5.445	4.265	17.931	**2.817**
kroA200	4.893	5.438	7.388	**2.747**
kroB200	5.682	7.660	14.944	**3.326**
ts225	**2.099**	7.627	6.448	2.742
tsp225	3.807	6.078	10.409	**3.390**
pr226	4.753	**1.871**	9.276	4.601
gil262	5.200	6.686	7.881	**2.096**
pr264	7.277	6.572	28.820	**1.551**
a280	5.737	11.167	13.722	**5.572**
pr299	6.877	7.686	24.754	**4.663**
lin318	5.565	7.961	16.362	**4.186**
rd400	**3.866**	—	12.274	3.923
fl417	5.370	—	13.318	**3.526**
pr439	**6.431**	—	28.604	7.120
pcb442	5.639	—	11.380	**4.241**
d493	**4.514**	—	20.373	4.917
総平均（lin318 まで）	4.683	4.606	8.969	**2.230**
総平均（全問題例）	4.740	—	9.947	**2.529**

いエッジのうち，最も距離が短いものを経路に追加していくことで巡回路を構築する．本論文ではこの手法を最短辺優先法（short edge priority method: SEP）と呼ぶ．2-opt 法の解の精度と計算速度はその初期解の精度に大きく依存するため，事前に最短辺優先法である程度質の良い解を求めることでこの欠点を補っている．

表 5 TSPLIB の問題例に対する平均計算時間（s）
Table 5 Average solving times for TSPLIB instances.

問題例	2opt	**EV-2opt**	**EV-2opt+2opt**
eil51	0.016	0.373	0.383
berlin52	0.019	0.309	0.315
st70	0.030	0.623	0.628
eil76	0.025	0.662	0.666
pr76	0.070	0.644	0.669
rat99	0.086	1.108	1.132
kroA100	0.099	1.071	1.091
kroB100	0.131	1.174	1.206
kroC100	0.096	1.071	1.091
kroD100	0.112	1.059	1.094
kroE100	0.114	1.161	1.217
rd100	0.111	1.092	1.121
eil101	0.102	1.227	1.256
lin105	0.113	1.248	1.320
pr107	0.158	1.387	1.480
pr124	0.069	1.666	1.726
bier127	0.233	1.944	2.146
ch130	0.212	1.956	2.081
pr136	0.232	2.225	2.385
pr144	0.152	2.390	2.569
ch150	0.343	2.627	2.757
kroA150	0.399	2.594	2.774
kroB150	0.464	2.840	2.969
pr152	0.323	2.715	2.972
u159	0.339	2.931	2.962
rat195	0.435	4.131	4.490
d198	0.965	5.519	6.798
kroA200	1.001	5.289	5.628
kroB200	0.908	5.032	5.786
ts225	0.319	5.646	5.810
tsp225	0.621	6.554	7.221
pr226	1.205	7.166	7.675
gil262	1.170	9.006	9.827
pr264	0.669	11.738	15.513
a280	1.464	9.988	11.007
pr299	3.005	13.278	16.573
lin318	2.770	13.873	16.809
rd400	4.431	22.281	26.322
fl417	2.309	32.772	40.458
pr439	6.689	33.354	47.265
pcb442	6.336	26.308	30.803
d493	11.035	43.078	61.138

テスト用問題例には 16 個のランダムな問題例と，TSPLIB [14] に含まれる 42 個の問題例を使用する．TSPLIB の問題例に関しては，S2V-DQN（Stracture2Vec Deep Q-Learning）による実験結果 [8] も同様に比較する．求めた解の精度を評価する指標として，最適解の経路長 L^* に対して，経路長 L の解の誤差率 ε を式 (3) に従って計算する．

表 6 初期解アルゴリズムによる平均誤差率の変化（ランダム）(%)
Table 6 Difference of average error ratios by initial solutions for random instances.

問題例	SEP			Farthest Insertion			Christofides		
	初期解	2opt	**EV-2opt+2opt**	初期解	2opt	**EV-2opt+2opt**	初期解	2opt	**EV-2opt+2opt**
rand20-A	6.282	1.389	**0.000**	1.304	1.057	**0.000**	10.922	4.858	**0.000**
rand20-B	0.958	**0.000**	**0.000**	3.246	2.048	**0.000**	7.186	2.395	**0.000**
rand50-A	18.016	5.207	0.170	5.787	4.241	**0.152**	6.175	0.815	0.288
rand50-B	8.097	2.449	**0.405**	4.312	4.069	**0.405**	13.360	5.587	**0.405**
rand75-A	8.644	2.196	2.369	6.681	6.025	**1.074**	14.343	3.004	1.157
rand75-B	17.239	7.034	**0.685**	7.459	6.232	0.761	14.085	6.103	0.795
rand90-A	9.837	2.925	**0.419**	5.668	4.678	0.421	8.473	2.995	0.868
rand90-B	21.874	7.680	0.993	11.202	10.019	1.677	11.668	4.194	**0.732**
rand100-A	23.695	5.485	2.727	7.007	5.767	**2.718**	13.164	5.030	2.848
rand100-B	13.934	7.149	0.022	6.899	5.864	0.171	9.780	4.693	**0.011**
rand100-C	23.130	5.118	**1.965**	7.743	6.390	2.269	11.623	3.219	3.847
rand150-A	16.549	5.708	**0.604**	8.763	7.014	0.801	11.203	4.094	0.675
rand200-A	10.192	5.015	2.620	8.940	8.120	**1.993**	13.227	6.223	2.054
rand300-A	15.210	4.630	4.449	8.484	7.251	3.996	11.400	4.481	**3.713**
rand400-A	16.371	4.857	4.635	10.144	8.974	4.917	13.421	7.301	**4.432**
rand500-A	18.791	5.277	5.261	10.300	8.875	5.125	11.838	**3.410**	4.023
総平均	14.301	4.508	1.708	7.121	6.039	1.655	11.367	4.275	**1.615**

$$\varepsilon = \frac{L - L^*}{L^*} \times 100\ (\%) \tag{3}$$

EV-2opt+2opt および 2opt を用いて，テスト用問題例に対して解を求める操作をそれぞれ 20 回試行する．これにより得られた解の平均誤差率を**表 2**，**表 4** に示す．また，EV-2opt+2opt の処理過程における解の例を図 4，図 5，図 6 のそれぞれ (c) から (e) に示し，(c) は最短辺優先法により得られた初期解，(d) はこれに EV-2opt を適用したもの，(e) はさらに 2opt を適用した後の解を表している．ただし問題例の名前に含まれる数字は頂点数を表している．表 2，表 4 より，ランダムな問題例と TSPLIB の両方において，EV-2opt+2opt による平均誤差率の総平均は 2opt および S2V-DQN よりも低くなった．また各問題例ごとに比較した場合でも，ほとんどの問題例において平均誤差率は低い値となった．このことから，提案手法を用いることによる解の精度向上を見ることができる．この結果について，2-opt 法を適用する前に EV-2opt 法を用いることでより最適解に近い解を求め，より質の良い解から 2-opt 法による探索を始めることができたため，誤差率が改善したと考えられる．

頂点数と精度の関係をみたとき，200 程度の頂点数を境にして精度の改善の効果は弱くなる傾向にある．これは訓練データとして頂点数が 100 以下の問題例を使用していることや，4.1 項で述べたように，頂点数が密集する領域における優良エッジ分布の性質に大きく起因すると思われる．より大きい頂点数の問題例を学習データとして用いることで有効な頂点数を向上できることが見込まれるが，頂点が膨大な数になると画像上でつぶれてしまい表現できなくなる．そのため，優良エッジ分布により精度良く扱うことができる頂点数の上限値は，画素数などに強く依存すると思われる．

4.3 計算時間評価

問題例が与えられてから解を 1 つ求めるまでの平均計算時間を**表 3**，**表 5** に示す．EV-2opt+2opt の計算時間は優良エッジ分布から優良エッジ値を求める処理が支配的であり，この処理の計算時間は頂点数 n に対して $\mathcal{O}(n^2)$ に従う．表 3，表 5 の実験結果からは，各手法の計算時間がおよそ $\mathcal{O}(n^2)$ に従っていることが推測される．

本論文の提案手法は，優良エッジ分布により問題の大域的な構造をとらえることで解の精度を向上するものと考えられる．一方，解を求める手法として局所探索法を採用しているが，局所探索法では遷移できる解が現在の解の近傍に制限されるため，優良エッジ分布が持つ大域的特徴を高速に反映することが難しいということが懸念される．これに関して，大域的特徴を効率良く反映し，アルゴリズムを高速化するためには，初期解を構築する部分で優良エッジ分布を利用することが妥当であると思われる．

4.4 初期解アルゴリズムによる比較

4.2 項の実験では初期解を求めるアルゴリズムとして最短辺優先法（SEP）を使用していたが，この手法は局所的に偏った解を与えることが多く，高精度なアルゴリズムとはいえない．これに対して，farthest insertion や Christofides アルゴリズムなど [15]，より精度が高く大域的な問題の特徴を考慮するような TSP の解法を用いた場合，解の精度がどのように変化するかについて検証する．

今回の実験では，最短辺優先法の代わりに farthest in-

表 7 初期解アルゴリズムによる平均誤差率の変化（TSPLIB）(%)
Table 7 Difference of average error ratios by initial solutions for TSPLIB instances.

問題例	SEP			Farthest Insertion			Christofides		
	初期解	2opt	**EV-2opt+2opt**	初期解	2opt	**EV-2opt+2opt**	初期解	2opt	**EV-2opt+2opt**
eil51	24.648	3.462	2.958	5.528	5.246	**2.359**	8.451	3.991	2.477
berlin52	31.941	9.382	**0.000**	7.248	5.642	**0.000**	13.498	8.134	**0.000**
st70	11.111	3.993	0.452	5.889	5.059	**0.074**	14.222	4.007	0.970
eil76	8.736	4.089	0.037	7.825	6.599	0.195	13.011	6.320	**0.009**
pr76	36.370	4.293	**2.188**	6.635	3.743	2.640	7.882	4.960	2.267
rat99	18.910	4.013	1.936	9.174	7.820	1.920	15.029	3.604	**1.367**
kroA100	14.120	2.407	0.477	6.438	4.709	**0.425**	9.449	6.615	0.491
kroB100	16.585	2.238	**0.660**	5.681	5.204	1.602	8.450	2.533	2.731
kroC100	12.270	3.162	**0.104**	6.024	4.999	0.146	9.653	4.743	0.186
kroD100	14.812	5.071	0.986	6.482	5.828	0.959	11.679	3.298	**0.889**
kroE100	12.588	2.696	2.708	6.074	4.791	2.432	7.935	3.756	**2.376**
rd100	16.979	4.149	**1.320**	9.834	8.601	1.782	12.592	5.069	1.564
eil101	24.483	6.065	1.932	7.925	7.107	**0.747**	12.401	6.113	0.851
lin105	16.601	3.418	0.807	7.438	6.087	**0.135**	14.660	8.574	0.178
pr107	7.318	1.943	1.750	2.886	2.421	**1.527**	8.133	2.388	1.768
pr124	10.110	2.698	2.506	6.315	5.758	1.174	9.551	1.364	**0.521**
bier127	19.493	6.416	2.681	7.844	6.833	2.643	12.706	6.259	**2.314**
ch130	18.216	4.865	3.205	6.909	5.847	2.424	11.964	4.492	**2.422**
pr136	19.919	9.426	**2.221**	8.548	7.731	2.459	7.233	2.240	2.298
pr144	12.483	1.507	**1.198**	6.291	4.961	1.339	20.625	1.768	1.626
ch150	19.623	7.027	2.500	8.860	7.746	1.487	10.018	3.144	**1.362**
kroA150	20.238	3.579	2.672	7.668	6.517	2.489	10.685	2.229	**2.212**
kroB150	20.272	5.987	2.577	6.696	6.039	**2.258**	14.171	3.989	3.251
pr152	15.931	3.036	**1.254**	3.574	2.433	1.839	7.522	1.652	1.355
u159	17.845	5.617	0.107	10.892	9.923	**0.055**	11.445	6.735	0.304
rat195	13.991	5.390	5.568	11.784	10.620	5.155	14.378	5.613	**4.864**
d198	21.337	5.445	**2.817**	5.911	4.342	3.292	9.670	3.123	3.462
kroA200	17.659	4.893	**2.747**	7.066	6.500	3.298	12.609	6.598	3.495
kroB200	22.210	5.682	3.326	7.822	6.588	3.292	11.438	3.208	**2.740**
ts225	5.383	2.099	2.742	10.121	8.388	4.342	5.242	**1.907**	3.236
tsp225	9.661	3.807	**3.390**	11.222	9.810	6.936	12.795	5.294	6.440
pr226	21.441	4.753	4.601	2.610	**1.735**	2.073	14.886	3.099	2.224
gil262	15.181	5.200	**2.096**	10.019	8.631	2.241	13.457	3.905	2.338
pr264	11.884	7.277	**1.551**	11.897	10.290	4.965	10.970	5.059	4.270
a280	19.775	5.737	5.572	13.470	11.958	6.313	13.920	**3.527**	4.585
pr299	31.421	6.877	4.663	10.205	8.720	5.679	9.975	**4.276**	5.683
lin318	18.723	5.565	**4.186**	8.858	7.677	4.201	12.901	7.324	4.814
rd400	13.029	**3.866**	3.923	9.807	8.504	3.930	12.637	5.052	4.565
fl417	9.021	5.370	3.526	7.910	6.459	**2.757**	10.218	4.572	3.717
pr439	20.080	6.431	7.120	12.493	10.456	6.028	12.256	**4.148**	4.934
pcb442	20.280	5.639	4.241	12.474	11.127	3.950	8.045	**2.992**	3.775
d493	18.172	4.514	4.917	9.090	8.183	5.660	10.221	**3.953**	5.620
総平均	17.401	4.740	**2.529**	8.034	6.848	2.601	11.395	4.324	2.537

sertion または Christofides アルゴリズムを用いて初期解を求め，これに対して 4.2 節と同様に 2opt と EV-2opt+2opt をそれぞれ適用した場合の解の誤差率を比較する．この実験の結果を**表 6**，**表 7** に示す．これらの結果より，提案手法 EV-2opt+2opt による誤差率には初期解を与えるアルゴリズムによる大きな差異はみられず，いずれのアルゴリズムでも通常の 2-opt 法だけを行った場合（2opt）より精度が改善されていることが分かる．以上のことから，提案手法による解の精度の向上を確認することができる．

5. 結論

本論文では，平面 TSP における距離に代わる指標とし

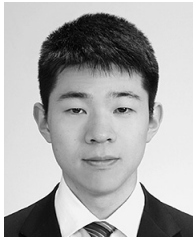
て，畳み込みニューラルネットワークにより最適経路画像を近似した優良エッジ分布と，ここから計算される優良エッジ値を提案した．また，優良エッジ値を利用した局所探索法としてEV-2opt法を提案し，その性能を検証するために実験を行った．その結果，比較手法よりも解の誤差率を改善したことを確認し，平面TSPの解法として有効な手法となりうる可能性を示した．

提案手法では訓練用問題例の最適解の画像を教師信号として学習を行ったが，規模が大きい問題例の最適解を求める計算コストは大きく，大量の訓練データを事前に用意することは難しい．そこで強化学習を用いて，解を生成しながらその目的関数の値に従ってネットワークを学習し，汎化能力を向上させることが今後の課題として考えられる．また，本手法は解が画像で表現される問題に適用可能であると考える．たとえば配送計画問題や最適配置問題，ガントチャートで表されるスケジューリング問題などがその適用先としてあげられる．

謝辞 本研究の一部は，JSPS科研費18K11484と，JSPS科研費17K01309，関西大学大学院理工学研究科高度化推進研究費，関西大学先端科学技術推進機構「緊急救命避難支援のための情報通信技術に関する研究開発」研究グループの助成を受けている．

参考文献

[1] Helsgaun, K.: General k-opt submoves for the Lin–Kernighan TSP heuristic, *Mathematical Programming Computation*, Vol.1, No.2, pp.119–163 (online), DOI: 10.1007/s12532-009-0004-6 (2009).
[2] Nagata, Y. and Kobayashi, S.: A Powerful Genetic Algorithm Using Edge Assembly Crossover for the Traveling Salesman Problem, *INFORMS Journal on Computing*, Vol.25, No.2, pp.346–363 (online), DOI: 10.1287/ijoc.1120.0506 (2013).
[3] Isola, P., Zhu, J.-Y., Zhou, T. and Efros, A.A.: Image-to-Image Translation with Conditional Adversarial Networks, *CVPR 2017*, arXiv:1611.07004 (2016).
[4] Radford, A., Metz, L. and Chintala, S.: Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, arXiv preprint arXiv:1511.06434 (2015).
[5] Silver, D., Huang, A., Maddison, C.J., et al.: Mastering the game of Go with deep neural networks and tree search, *Nature*, Vol.529, pp.484–489 (2016) (online), available from ⟨http://dx.doi.org/10.1038/nature16961⟩.
[6] Vinyals, O., Fortunato, M. and Jaitly, N.: Pointer Networks, arXiv preprint arXiv:1506.03134 (2015).
[7] Bello, I., Pham, H., Le, Q.V., Norouzi, M. and Bengio, S.: Neural Combinatorial Optimization with Reinforcement Learning, arXiv preprint arXiv:1611.09940 (2016).
[8] Dai, H., Khalil, E.B., Zhang, Y., Dilkina, B. and Song, L.: Learning Combinatorial Optimization Algorithms over Graphs, arXiv preprint arXiv:1704.01665 (2017).
[9] Applegate, D.L., Bixby, R.E., Chvatal, V. and Cook, W.J.: Concorde TSP Solver (2006), available from ⟨http://www.math.uwaterloo.ca/tsp/concorde/⟩ (accessed 2018-08-08).
[10] Ronneberger, O., Fischer, P. and Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation, arXiv preprint arXiv:1505.04597 (2015).
[11] Yonetsuji, T.: PaintsChainer (2017), available from ⟨https://github.com/pfnet/PaintsChainer⟩ (accessed 2018-08-08).
[12] Maas, A.L., Hannun, A.Y. and Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models, *ICML Workshop on Deep Learning for Audio, Speech and Language Processing* (2013).
[13] Kingma, D.P. and Ba, J.: Adam: A Method for Stochastic Optimization, arXiv preprint arXiv:1412.6980 (2014).
[14] Reinelt, G.: TSPLIB—A Traveling Salesman Problem Library, *ORSA Journal on Computing*, Vol.3, No.4, pp.376–384 (online), DOI: 10.1287/ijoc.3.4.376 (1991).
[15] Applegate, D.L., Bixby, R.E., Chvátal, V. and Cook, W.J.: *The traveling salesman problem: A computational study*, Princeton University Press (2011).



三木 彰馬

2017年関西大学システム理工学部電気電子情報工学科卒業．同年同大学大学院理工学研究科システム理工学専攻電気電子情報工学分野アルゴリズム工学研究室に配属．深層学習を用いた組合せ最適化問題の解法等の研究に従事．

榎原 博之 （正会員）

1982年大阪大学工学部通信工学科卒業．1987年同大学大学院博士（通信）課程修了．同年同大学工学部助手．1994年関西大学工学部専任講師となり現在，教授．組合せ最適化問題，計算幾何学，並列アルゴリズム等の研究に従事．工学博士．IEEE，ACM各会員．