事例研究［論文］

タクシーの流し営業における強化学習を用いた顧客獲得ナビ

岩田　真奈，桑原　惇，石塚　湖太，倉又　迪哉，清原　明加，中田　和秀

1. はじめに

現在，タクシー業界では，低収入・長労働が問題視されており，ドライバーの数も右肩下りである [1]. 2020 年の通常国会では，タクシー運転の際に必要となる二種免許の取得に関して年齢や運転歴の受験資格を緩和する法案が成立する [2] など，新規ドライバーを増やすための動きも活発である．また，タクシーの営業形態としては，道路を走行しながら顧客を探す「流し」，タクシー乗り場など特定の場所で顧客を待つ「付け待ち」，配車された場所に向かう「無線配車」などの形態がある．大都市では流しが 61％を占めており，主要な営業形態となっている [3]．しかしながら，新人ドライバーや土地勘のない場所を運転するドライバーにとって，顧客にすぐに出会えるようにタクシーを流すことは容易ではない．このような現状を踏まえて，本研究では流し営業において顧客を獲得できる可能性が高い運転ルートをリアルタイムで推薦するモデルを提案する．

本研究では，まず与えられた GPS 時系列データから，どの道路にいるのかを表すルート情報を生成するためにマップマッチングの手法を導入した．その際に，大量の GPS 時系列データを高速に処理する手法を提案し，実用性を確認した．また，長期的なスパンでの顧客獲得という観点でルート推薦をするために強化学習を用いる．今回の問題設定に合った強化学習モデルとして R2D3 [4] を選択し，適切に環境を設定したうえで学習を行った．そして，実データを用いたシミュレーションにより，教師あり学習モデルや実際のドライバーよりも効率的なルートが選択できることを確認した．

いわた　まな，くわばら　しゅん，いしづか　こうた，くらまた　みちや，きよはら　はるか，なかた　かずひで
東京工業大学工学院経営工学系
〒 152–8552　東京都目黒区大岡山 2–12–1
受付 20.7.25　採択 20.11.5

2. データの特性と課題

本節では，データの特性に関して述べ，モデル選択および作成の際に留意すべき点を説明する．その後に，提案手法の全体像を述べる．

2.1 データの特性

本研究では，経営科学系研究部会連合協議会主催，令和元年度データ解析コンペティションで提供されたデータを使用する．提供されたデータは，都内タクシーのプローブデータであり，GPS 時系列データやタクシーが乗車可能であるかなどの情報である．GPS 時系列データは平均で 40 秒に 1 回程度の間隔で取得されており，low-sample なデータである．さらに，対象ドライバー数は 2 万人を超え，2 年間に渡る大規模なデータとなっている．

2.2 解決するべき課題

本研究では，流し営業において効率的な運転ルートを推薦することを目指す．基礎分析を行ったところ，長距離の顧客を見つけるよりも空車時間を短くする方が実車走行距離を長くできると判明した．そのため，本研究では乗車可能状態になってから次の顧客獲得までの時間（顧客獲得時間）を短くするようなルートを推薦することを目標とする．運転ルートの推薦にあたっては，実際のドライバーが行っているであろう「おおよそどの道をどの方向に向かってタクシーを流すか」という粗い粒度での判断に近い形で顧客獲得システムを構築する方向も考えられる．しかし，本論文ではすべての道の移動を考慮することで，タクシー運転手が車を運転中に推薦されたルートを確認し直進などの判断を下せるナビのような形でのルート推薦を行う．そのため，オンライン性が高く瞬時に推論できるモデルが必要となる．ナビのような形でのルート推薦を実現することで，まったくの新人ドライバーや顧客を送り届ける中で土地勘のない場所を走行するドライバーに対して，具体的に細かく道を示すことが可能となる．また，実際のドライバーよりも細かい粒

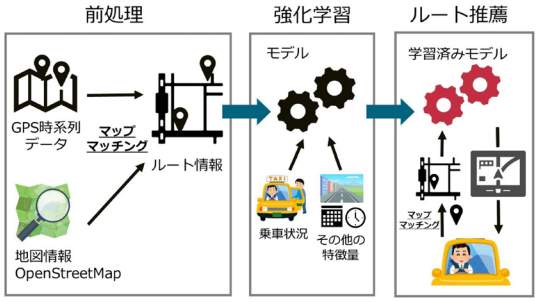

図 1　提案手法の全体像

度での分析を行うことで，現状行っている流し営業の成績を超える顧客獲得システムを構築できる可能性がある．

2.3　全体像

提案手法の全体像は図 1 のように表される．

まず，提供された GPS 時系列データをルート情報に変換するために，前処理としてマップマッチングを行った．ここで地図情報として，OpenStreetMap[1]を使用した．そして，ルート情報と乗車状況，その他の特徴量を用いて学習することで，顧客獲得時間を最小とするルートを推薦するモデルを作成した．この前処理は，計算機サーバーなどで実行することを想定している．実際にルートを推薦する際には，個々のタクシーに搭載されたコンピュータで独立に計算を行う．具体的には，ドライバーが移動する度に再度マップマッチングを行う．そのうえで，学習済みのモデルを用いてルートを適宜推薦する．このようにルート推薦を行うことで，新人ドライバーや土地勘のない場所を運転するドライバーが顧客に出会えるようにタクシーを流すことが可能となる．

このように，マップマッチングは，前処理時とルート推薦時に必要となる．前処理時に使われるマップマッチングに求められる条件は大規模なデータにも対応できることであり，ルート推薦時に使われるマップマッチングに求められる条件は推論が迅速であることである．

3.　関連研究

3.1　マップマッチング

本研究の関連研究として，マップマッチングについて説明する．マップマッチングとは GPS などで得られた現在地の情報と今までに走行してきた道路を比較し，最も適切な道路上に現在地を補正させるシステムである．日本においてはカーナビゲーションシステムなどに導入されているが，カーナビゲーションで実際に使用されるアルゴリズムは公開されていない．

マップマッチングは，オンラインマップマッチングとオフラインマップマッチングという 2 種類がある．オンラインマップマッチングは，リアルタイムで取得される GPS データに対して行われる．オフラインマップマッチングは，すでに取得済みの GPS データに対して行われる．オンラインマップマッチングにおいては隠れマルコフ (Hidden Markov Model, HMM) モデルをベースとした手法が多く提案されている．HMM をベースにした手法では，文献 [5] のように推論の正確性を保つために，収束点を検出した際にルートを確定させていた．そのため，常に推論の遅れが発生してしまうという問題点があった．また，誤った推論をした際にそれを後から正す仕組みがなかった．そこで，Luo et al. [6] はそれまでのルートを正しいと仮定するという日和見主義的な考え方を採用することで，推論が遅れる問題を解決した．ここでは，日和見主義的な形で推論を行い，そのうえで異常な推論に対しては，ロールバックメカニズムで間違えを正すという方針 (incremental route inference algorithm with rollback, INC-RB) を導入している．その結果，正確性を保ちながらも即座に推論を行うことが可能となった．

3.2　強化学習

本研究のもう一つの関連研究として，強化学習 [7] について説明する．強化学習とは，与えられた環境の中で，将来得られる報酬和を最大化するよう，エージェントと呼ばれる行動主体を学習させる枠組みである．

強化学習では，ある時刻において，ある状態に遷移したときに，エージェントがある行動を選択すると，所与の状態と選択した行動に依存して報酬と新たな状態が得られる，というサイクルが成立する．これはマルコフ決定過程 (Markov Decision Process, MDP) として，$(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{S}', p_t)$ で定義される．なお，$\mathcal{S}$ は各時刻における状態の集合 $\{s^n_{t=0}\}$，$\mathcal{A}$ は行動の集合 $\{a^n_{t=0}\}$，$\mathcal{R}$ は状態と行動に依存して決まる報酬の集合 $\{r^n_{t=0}\}$，$\mathcal{S}'$ は次の状態の集合 $\{s'^n_{t=0}\}$ であり，p_t は状態の遷移確率 $(\mathcal{S} \times \mathcal{A} \times \mathcal{S}')$ である．

ここで，タイムステップ T において，即時に得られる報酬 r_T を最大化する行動を選択した場合，それは必ずしも将来得られる報酬和の最大化につながるわけではないことに注意する．このため，強化学習では以下の式 (1) で与えられる J を最大にするような行動を選択することを考えている．

[1] https://www.openstreetmap.org/

$$J = \sum_{t=T}^{n} \gamma^{t-T} r_t \tag{1}$$

ここで $\gamma \in (0, 1]$ は割引率である．すなわち，J は T 期以降に得られる報酬 r_t の現在割引和を意味する．つまり，強化学習では現在だけでなく将来の報酬も加味したうえで行動を選択する．特に，Q 学習と呼ばれる手法では，報酬の現在割引和 J に現在の行動が繋がるかどうかのポテンシャルを表す価値を導入する．より具体的には，状態と行動に付随する価値をこれまでの行動と獲得報酬の組である経験の蓄積を基に学習し，その価値を最大化することで，近似的に式 (1) で表される報酬和の最大化を行っている．

一方，強化学習はすべての学習環境に対し，状態に依存する行動の価値を容易に推定できるわけではない．特に，部分的にしか環境を観測できない場合やスパースな報酬の場合には，学習が困難となる．

まず，部分的にしか環境を観測できない場合について説明する．強化学習の環境においては，エージェントが状態全体を把握できず，一部の状態のみが観測可能である場合が考えられる．このとき，先述の MDP は，部分観測マルコフ決定過程 (Partially Observable Markov Decision Process, POMDP) として捉え直すことができる．POMDP は $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{O}, \mathcal{S}', p_o, p_t)$ で定義される．ここで，$\mathcal{S}$, $\mathcal{A}$, $\mathcal{R}$, $\mathcal{S}'$, p_t は MDP の場合と同様である．加えて $\mathcal{O}$ は観測された状態の集合 $\{o_{t=0}^n\}$, p_o はある状態 s_t において観測 o_t が得られる確率 $(\mathcal{O} \times \mathcal{A} \times \mathcal{S})$ として新たに導入される．このとき，POMDP では現在の状態を知るための手がかりとして，エージェントが部分的に観測可能な状態に限られてしまう．そのため，現在の状態を特定するための情報量が少なく，状態に依存した行動価値を求めることが難しくなるという課題がある．この POMDP の課題に対して，Kapturowski et al. [8] の提案した R2D2 では，一連の時系列に対し，連続したデータひとまとまりをエピソードとして価値関数の学習に使用することで，現在の状態をより正確に推測できるようになった．これによって，各時間ステップを独立で学習するよりも，状態に依存した行動価値の推定をより正確に行いやすくなり，POMDP における学習の困難性に対処した．

次に，スパースな報酬について説明する．強化学習の単純な問題設定では，ブロック崩しゲームのように，すべてのブロックが崩れなくても，ブロックを一部崩すたびに報酬が得られる．そのため，毎回の行動でほとんどの場合何かしらの報酬が得られていた [9]．一方，囲碁のようなゲームの場合，長い時間軸にわたり正しい行動をとり続けることでようやくゲームに勝ち，報酬を得ることができる [10]．このようなスパースな報酬の場合，各時間ステップにおいて正しい行動を取り続ける同時確率は非常に小さくなってしまう．これにより，途中までは報酬に繋がる行動をしていたが，ある時刻において正しくない行動をしてしまったため，報酬に繋がらなかった，という事例が起こりやすくなる．すると，途中まで行っていた正しい行動について，その行動は将来の報酬和最大化に有効であったにもかかわらず，きちんと評価されない．そのため，スパースな報酬の場合は学習過程において報酬に繋がる行動の探索が困難である．この問題に対し Gulcehre et al. [4] の提案した R2D3 では，学習データにデモンストレーションと呼ばれる人間の行動経験をごく少量混入させるという工夫を行った．この方針を取ることで，ある程度報酬にたどり着けている人間のデータをヒントとしてエージェントに学習させ，学習時の報酬に繋がる行動の探索が容易になる．また同時に，人間データの導入は初期状態の種類が多い状況において，初期状態の観測確率に依存して報酬へたどり着くための同時確率が低くなる状態においても有効である．なぜなら，同様に将来の報酬和を最大化するための報酬に繋がる行動の探索および行動価値の学習が行いやすくなっているからである．さらに，R2D3 は R2D2 を基に改良を加えたモデルであることから POMDP の困難性にも対処している．以上の理由により，R2D3 は

1. 報酬がスパースな場合
2. エージェントが環境を部分的にしか観測できない（POMDP である）場合
3. 初期状態の種類が多い場合

という三つの学習上の困難性に対処することができる．

4. 提案手法

本研究では，大規模な GPS 時系列データからルート情報を生成し，その情報を基に顧客獲得のためのルート推薦を行う強化学習モデルを構築した．以下では，4.1 節で高速なマップマッチング法について， 4.2 節で強化学習モデルについて説明を行う．

4.1 高速マップマッチング

2 節で述べたように，今回はオンライン性のあるマップマッチングを行う．そのため， 3.1 節で述べたように，データに対して正確性を保ちながらも即座に推論ができる INC-RB [6] を採用した．しかしながら，INC-

RB を大規模データに適用すると現実的な時間では前処理が終わらないという問題点があった．そのため，本節では，INC-RB の概要を説明したうえで，改善策の提案を行う．この改善により，INC-RB を大規模データに対しても高速に行うことが可能となる．

4.1.1 INC-RB の概要

INC-RB のマップマッチングのアルゴリズムについて説明する．INC-RB ではオフライン処理とオンライン推論が行われる．オフライン処理では，道路間での移動のしにくさを表す遷移コストを設定する．オンライン推論では，新しい GPS 座標に対して推論が行われる．この際に，オフライン処理で設定した遷移コストを使用する．

まず，オフライン処理について述べる．オフライン処理では，グラフとして表されている道路情報を基に，道路から道路への遷移コストを計算する．このとき文献 [11] で採用されている基準に従って計算を行った．このコストは，各道路の長さと道路同士の接続角度に依存しており，実用的な道を選択できるようになっている．

次に，オンライン推論について述べる．今，GPS 時系列データとして新たに $g_i(i > 1)$ という GPS 座標が与えられたとする．INC-RB では $i-1$ 番目までのルート L_{i-1} を正しいと仮定して，i 番目のルート L_i を求める．具体的な手順は以下のようになる．

INC-RB のルート推定の手順：

1. 位置 g_i に対して，候補となる近隣道路を複数列挙する
2. 候補とした道路に対し，g_i までの距離などを考慮して遷移コストを決定する
3. L_{i-1} の最後の道路を s_{i-1}^{lst} とする．s_{i-1}^{lst} から，候補となる道路へのルートの中で，ルート上の遷移コストの和が最小となるものを求める．また，遷移コストの和をそのルートのスコアとする
4. ソートを行い，最小スコアのルートを推論結果 L_i とする
5. 異常なルートが検出された場合，以前のルートの推論をやり直す

異常なルートが検出された場合に以前のルートの推論をやり直すことにより，多くの場合は日和見主義の推論方法で十分となる．なお，ルート L_i が確定した時点で，i 時点で位置している道路も特定されている．

INC-RB で使用する道路情報は OpenStreetMap から取得している．OpenStreetMap では，頂点は交差点，辺は道路に対応した有向グラフで表現されている．また，道路に対しては，その道路を覆う長方形の中で最小となる長方形がわかっており，その長方形の頂点は四つの緯度経度情報で表されている．

4.1.2 ルート推定の手順 1 と 3 の改善

まず最初に，INC-RB のルート推定の手順 1 の改善に関して述べる．候補となる道路を複数列挙する際には，GPS 座標 g_i を中心とした適当な大きさの長方形を考える．上で述べたように，道路情報は長方形として表される．この道路集合に対応した長方形集合の中で，最初の長方形と共通部分をもつものをすべて探し出すことにより，GPS 座標に近い道路を列挙している．文献 [6] では，この作業を R-Tree というデータ構造を用いて実現していたが，本研究では PR-Tree [12] を用いて実現した．これにより，道路数を N としたとき，対象となる道路を探索する際の最悪計算量が $O(N)$ から $O(\sqrt{N})$ に改善された．

次に INC-RB のルート推定の手順 3 の効率的な計算法について述べる．遷移コストは二つの道路の間で定義されている．よって本研究では，交差点を頂点とするのではなく，道路を頂点としてもつ形でグラフを再構築し，辺に遷移コストを紐付けた．この工夫により，遷移コストの和が最小となるルートを探す問題は標準的な最短経路問題となる．これにより，既存の効率的なアルゴリズムを適用することが可能となった．

4.1.3 最短経路問題に対する改善

4.1.2 節の工夫を行うことで，最短経路問題へと帰着できた．ただし，一度のマップマッチングで多くの最短経路問題を解くことになる．また，2.1 節で述べたように，前処理では大量のマップマッチングを行う必要があるため，計算時間が問題となる．そこで，次の五つの高速化のための工夫を行った．

(a) 複数候補のある道路の探索の並列化
(b) A*法の導入
(c) 最短経路問題の解をキャッシュメモリに保持
(d) キャッシュメモリに保持されたデータを用いた，最短経路問題の効率的な枝刈り
(e) ドライバーの並列化

以下では，これらの工夫について順に説明する．

一つの GPS 座標に対して，複数の候補となる道路がある．そこで (a) では，それぞれの候補の道路に対する最短経路問題を並列で解いた．このとき各スレッドから各最小コストのルートを集める際に，排他制御が必要である．しかしながら，実装で使用した Python のライブラリである Numba では排他制御の一種である mutex が使用できなかった．そのため，Numba の

バックエンドである LLVM の Compare and Swap 命令を直接制御することで排他制御を実装した．

今回扱うデータが位置情報であることを考慮すると，2 次元座標の情報を活用することで効率性が増す可能性がある．そこで，(b) ではヒューリスティックな情報を用いた最良優先探索アルゴリズムである A*法 [13] を導入した．A*法は，適切な下界の設定により探索の効率性が増すと，実行速度が速くなるという特徴がある．今回は下界としてユークリッド距離を用いた．

タクシーには同じタクシー乗り場から出発するなどの理由により，部分的に同じ道を移動することが多いという特性がある．そのため，(c) ではある道路から別の道路への最適解およびその値を保存し，後で利用することは有効であると考え，キャッシュメモリを導入した．

A*において，上界を導入することによる枝刈りは効率化に繋がる．そこで，(d) では文献 [14] のアイデアをもとに，キャッシュメモリ上の情報と三角不等式によって上界を計算し，探索中に枝刈りを行う．例として，$A \to B$ の最短経路問題を解く途中で，ある $A \to X \to B$ を満たす X を探索することを考える．$X \to B$ の最適解がキャッシュメモリ上にあれば使用する．こうすることで一部の計算が必要なくなる．さらに，別の X' を経由するルートの総コストの下界はユークリッド距離から得られている．これがキャッシュメモリ上にある $X \to B$ の最適値と，$A \to X$ の和よりも大きければ，枝刈りが行えるため探索が削減できる．

(a) の並列化のみでは CPU 利用率に余裕があったため，(e) では，さらにドライバーの並列化も導入した．ここでは，複数のドライバーに対して同時にマップマッチングを実行した．この際には，キャッシュメモリ上のデータのみ共有した．これにより，より効率的な並列化が実現できた．

4.2 R2D3 を用いたルート推薦

次の道路で顧客を獲得できるかどうかだけでなく，その先に続く道路での顧客獲得可能性も考慮に入れてルート推薦を行った方が効果的である．たとえば，一つ先の道路を直進すると顧客獲得率が 0.3，左折すると 0.5 であるが，二つ先の道まで考えたときに，一つ目を直進してから左折した場合が 0.9，一つ目を左折してから直進した場合 0.1 のようなケースを考える．このようなケースでは，一つ目の道を直進させるようなルート推薦を行いたい．しかし，単純に教師あり学習を使うと，次の道路での顧客獲得しか考慮しない貪欲的な推薦になってしまう．これでは，長期的な視点で見たときの顧客獲得率を小さくしてしまう可能性がある．一方， 3.2 節で述べたように，強化学習を使用したモデルでは，式 (1) で示した値を最大化するよう行動を選択する．つまり，次の道路だけでなく，将来の顧客獲得可能性も見越したうえでルート推薦を行うことができる．そのため，本研究では強化学習を用いて学習および推論を行うことにした．

表 1 ルート推薦における POMDP としての環境設定

$\mathcal{S}, \mathcal{S}'$	状態	GPS による位置情報を基にどの道をどの方向に走っているか（東京 23 区内道路ベクトル），そのときの時刻，これまでに各道路をエージェントが訪問した回数，OpenStreetMap に含まれる道路情報，道路のリンク構造から作成した PageRank [15] を状態として入力した．ただし，初期状態は過去データからランダムサンプリングした点とする．また，顧客が獲得された時点でエピソードは終了する．
$\mathcal{A}$	行動	次の信号で直進するか，右折するか，左折するかなど*．
$\mathcal{R}$	報酬	遷移した道路で前後 3 分以内に乗車が起こっていたら 1 の報酬がもらえるものとし，顧客獲得の状況を近似的に再現した．
$\mathcal{O}$	観測可能な状態	エージェントの現在位置からの近傍となる状態．
p_o	観測確率	$\mathcal{O} \times \mathcal{A} \times \mathcal{S}$，一つ前のタイムステップでの状態と行動に依存した観測確率を示す．
p_t	状態遷移確率	$\mathcal{S} \times \mathcal{A} \times \mathcal{S}'$，一つ前のタイムステップでの状態と行動に依存した次の状態への遷移確率を示す．

*ある道路から隣接している道路に移る行為を指す

強化学習は環境とエージェントが相互に作用できる状況下でエージェントが学習していく枠組みである．しかし，今回は実際にタクシーを営業させながら強化学習を行うことはできず，過去データを基にシミュレーションを行う必要があった．そのため，過去データのみを使って学習できるよう，環境および報酬の設計を行った．なお，今回の問題設定は環境を部分的にしか観測できない POMDP であり，$(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{O}, \mathcal{S}', p_o, p_t)$ に関する設定の詳細を表 1 に示す．

図 2 は，表 1 に記述した強化学習の設定を，エージェントと環境における相互関係として表している．図 2 において，1 タイムステップの中で，まず環境がエージェントに対し状態を示す．次に，エージェントは示された状態の内，一部のみを部分的に観測することができる．そのうえで，観測可能な状態を基に将来得られる報酬を最大化するような行動を選択する．その結果，環境から報酬が得られる．ここで， 3.2 節でも述べたように，将来得られる報酬和の最大化は，行動に付随させた価値の最大化として近似されることに注意が必要である．

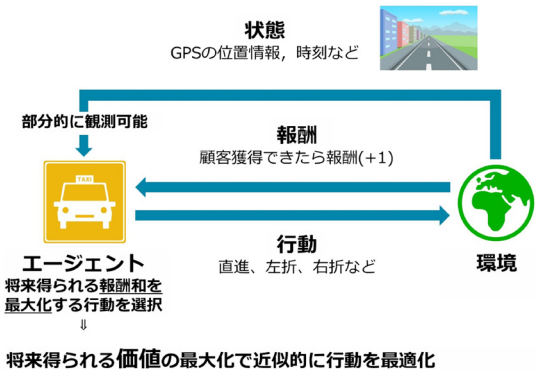

図 2　ルート推薦のための強化学習の概要

今回のルート推薦においては，強化学習の中でも，経験データをサンプリングして行動に付随する価値の学習が行える，方策オフ型の Q 学習を用いた．特に，使用したモデルである R2D3 は 3.2 節で述べた理由により，空車になった地点から顧客が乗車するまでの一連のタクシーのログをまとめたうえで，一つのエピソードとして学習させている．また， 3.2 節で述べた R2D3 のデモンストレーションとしては，過去のドライバーの実際の行動記録から，エージェントのシミュレーションデータに対し 1%の割合で経験データに混入させている．これにより，実際のタクシードライバーの運転をヒントに学習が行われるため，ある程度人間のドライバーにとって走りやすい実用的なルートを学習し推薦することができる．さらに，タクシーのルート推薦における以下の三つの課題を， 3.2 節で述べた理由により解決できている．

1. 乗車が起こるまでタクシーエージェントは報酬がもらえないため，正しい行動をとり続けなければ得られないスパースな報酬になっている
2. タクシーの顧客獲得可能性については，過去に通過した道路には既に顧客がいなかったことが判明している一方，未通過の道路に関しては情報が得られておらず，過去の行動すべてに顧客獲得可能性が依存することから，部分観測性をもつといえる
3. タクシーが空車に変わる地点，すなわちエージェントとの初期位置は，前の顧客を送り届けた地点に依存するため，初期地点の場所が非常に分散する

タクシーの顧客獲得を目的としたルート推薦に R2D3 を応用したこと，並びに，需要予測を必要とせずエンドツーエンドで顧客獲得のための実用的なルート推薦の最適化を行ったことは，本研究が初の試みである．

表 2　マップマッチングの実行時間

実験環境 [a]	計算時間（秒）
工夫なし	24118
＋ (a) 複数候補ある道路の探索の並列化	14782
＋ (b) A*法の導入	5787
＋ (c) キャッシュメモリの利用	963
＋ (d) キャッシュメモリを用いた枝刈り	737
＋ (e) ドライバーの並列化	196

[a] ＋ は一つずつ条件が追加されることに対応

5.　実データによる検証

5.1　データと前処理

令和元年度データ解析コンペティションで提供されたみずほ情報総研株式会社提供の都内タクシーのプローブデータを用いて検証を行った．検証では，2017 年 4 月 1 日〜4 月 30 日のデータを対象とした．対象ドライバー数は，23,621 人で，検証範囲は東京 23 区内である．

前述したように道路情報としてオープンデータである OpenStreetMap を利用している．今回対象となった東京 23 区の道路データは約 55 万の道路から構成されている．また，Python のライブラリである OSMnx[2] から東京のマップデータを取得した．前処理で使用した OS は Ubuntu 20.04 であり，実装には Python 3.7, および Numba 0.48 を用いた．

5.2　高速マップマッチングの効果

ここでは，4.1.3 節で提案した高速マップマッチングに対する工夫の効果を検証する．対象データは無作為に抽出した 16 ドライバーの 2017 年 4 月の 1 ヶ月分のデータである．また，今回の実験は，Ryzen 3950x 16 コア，32 スレッド，メモリ 128 GB で行った．表 2 に段階的に工夫を加えていった状況での計算時間を示した．

この結果より，工夫一つ一つに効果が認められることが確認できる．なおすべての工夫を取り入れたときにキャッシュされた解の総数は，3,547,161 件であった．最終的にすべての工夫を用いることによって約 120 倍の高速化に成功している．また，データや計算環境は違うものの，既存プロジェクト [16] と比較しても非常に高速である（単純に比較して 100 倍以上）．

ドライバー数が増えるにつれ，同じ最短経路問題がすでにキャッシュメモリにある確率が上がるため，より並列化による高速化の効果が期待できる．実際，今回対象とした 2017 年 4 月の都内の全ドライバー 23,621 人に関するデータ点数は約 3 億 3,400 万点であるが，約

[2] https://github.com/gboeing/osmnx

図 3　教師あり学習

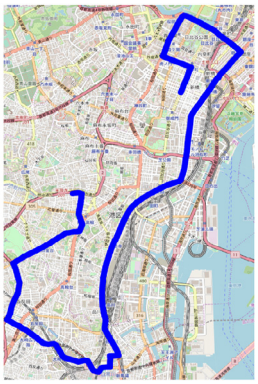

図 4　強化学習

3 日で計算を終えることができた．

5.3　教師あり学習との比較

ここでは，提案手法である強化学習と教師あり学習の比較を行う．

強化学習の学習には，2017 年 4 月 1 日～2017 年 4 月 14 日全ドライバーのデータを使用した．設定に関しては，4.2 節で述べたとおりである．また，この実験は Xeon E5-1620 3.50 GHz，8 コア，16 スレッド，メモリ 64 GB で行った．学習には約 4 日かかった．

比較手法として，LightGBM [17] を用いた教師あり学習を行った．LightGBM とは勾配ブースティングに基づく機械学習手法であり，高い予測精度が出やすいことから，近年教師あり学習でよく用いられる手法である．学習期間は，2017 年 4 月 1 日～2017 年 4 月 14 日までのデータである．まず，各道路における顧客獲得までの平均探索時間を計算した．データはスパースであったため，各曜日 2 時間ごとのデータを集計し利用した．そして，それぞれの道路において，隣接する道路の中で平均探索時間が最小となる道路を予測し，その方向に進む貪欲的な推薦モデルを作成した．過去のデータがなく学習できていない道路に関してはランダムで予測した．その結果，予測モデルの正解率は 75%を超えた．これをもって，比較手法として用いた教師あり学習の推薦モデルとしては，学習が十分であるとした．

図 3, 4 に，同時刻同地点[3]における教師あり学習での推薦と強化学習による推薦の様子を示した．教師あり学習では同じエリアで循環してしまい，不自然な動きをしていることがわかる．これは教師あり学習が次の道路しか考慮できないために発生する現象であり現実的ではない．一方，強化学習では循環が発生しないだけでなく，線路沿いの大通りを選択するなど人間に近いルート選択ができており実用性が高いまた，LightGBM においては二つの道路の行き来を続けるという動きが多く発生してしまうことを確認した．そのため，このような単純な動きを抑制することのできる強化学習の方が今回は有効であると判断した．

[3] 実際のドライバーのデータを取得

表 3　ドライバーの遷移距離

ドライバー	遷移距離 (km)
実際	24.0
強化学習	14.5

5.4　実際のドライバーとの比較

5.3 節では，提案手法が現実的なルート選択ができていることを確認したが，次にその顧客獲得の効果を測定する．そのため，実際のドライバーと強化学習の数値的な比較を行った．ただし，提案手法の評価は既に取得済みのデータを用いたシミュレーションによって行う以外になく，さまざまな仮定を置いたうえでの数値となる．そのため，完全な比較とはなっていないことに注意する．

検証データとして全ドライバー 23,621 人を対象に，同時刻同時点からスタートして，顧客を獲得するまでの遷移距離を比較した．対象期間は，2017 年 4 月 15 日～2017 年 4 月 30 日である．各ドライバーが 1 回の顧客探索を行い，それぞれの探索の遷移距離を比較した．その際には，流しと見られる動きを抽出するため，遷移距離が 1km 以上のものを対象とした．強化学習においては，実際の乗車は観測できないため，ある乗車が観測された 3 分前から乗車の瞬間までの時間帯にその道路を通った場合に乗車とみなしている．比較した結果を表 3 に示す．強化学習のドライバーは実際のドライバーよりも早く顧客を見つけることができた．全体の結果を，図 5，6 にヒストグラムを示す．

特に 5 km 以下において，強化学習が実際のドライバーよりも短い距離で顧客を見つけられていることがわかる．

また，ドライバー 500 人のサンプリングを行ったうえで，同様の条件で LightGBM との比較を行った．サンプリングを行ったのは，LightGBM の計算時間の問題があったためである．LightGBM においても，実際の乗車は観測できないため，ある乗車が観測された 3 分前から乗車の瞬間までの時間帯にその道路を通った場合に乗車とみなしている．その結果を表 4 に示す．この結果からも，LightGBM よりも強化学習の方が有効であることが示される．

さらに，実際のタクシードライバーの流しの運転の

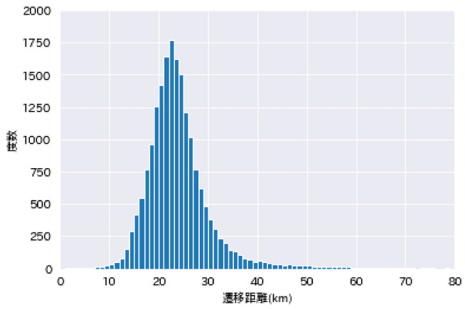

図 5　実際のドライバー

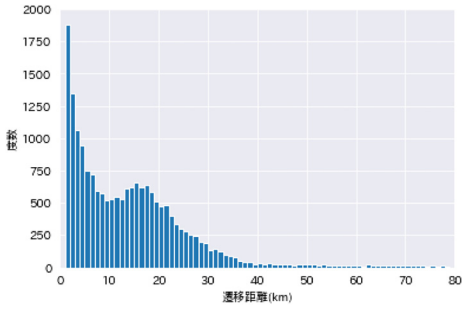

図 6　強化学習のドライバー

実績ルートとの比較を行う．比較のために，道路の共通率という指標を作成した．この指標は，強化学習のルートと実績ルートで共通する道路の数/実績ルートで遷移した道路の数という計算で求めた．そのため，実際のドライバーと全く同じルートを含むルートを通ると 1 となる．また，遷移距離の差=実際のドライバーの遷移距離−強化学習の遷移距離と定義した．この差が 0 より大きければ強化学習の方が良いという指標となる．道路の共通率と遷移距離の関係を図 7 に示す．

強化学習のルートと実績ルートはかなり異なること，および，実際のドライバーのルートと同じルートにいくかどうかはそれほど結果に影響しないことがわかる．

6.　結論と今後の課題

本研究では都内タクシープローブデータを使って学習を行い，流し営業において効率的な運転ルートをナビゲーションするための手法を提案した．提案手法では，オンラインでマップマッチングを可能にする INC-RB を利用している．その際，同手法を改善することにより，現実的な時間で膨大な数の GPS 時系列データをルート情報へと変換することができた．また，今回の状況設定に合った学習モデルとして強化学習の一つである R2D3 を選択した．そして，ルート情報と乗車情報を用いて適切に環境設定を行い学習することで，実用的で効率的なルートを提案することができた．

表 4　ドライバーの遷移距離

ドライバー	遷移距離 (km)
LightGBM	18.5
強化学習	14.1

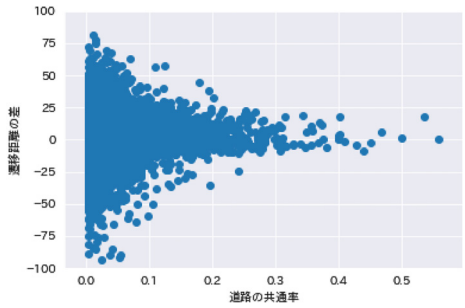

図 7　道路の共通率と遷移距離

しかしながら，提案手法の数値的な検証は十分に行えなかった．それは，新型コロナウイルスに起因する行動制限により計算環境への物理的なアクセスが制限されたため，大規模な実験を行うことができなかったためである．特に，新人ドライバーや土地勘のない場所を走っているドライバーに限定した提案手法の効果の測定や，強化学習が効果的であるエリアや時間帯の分析などは重要な今後の課題と考えている．さらに，導入した特徴量は道路情報や時間情報のみとなっている．さらなる精度向上のためには，曜日などの詳細な情報，さらに駅の場所や天気などの外部情報を使用する必要があると思われる．また，実際のドライバーとの比較においては，既に取得済みのデータを用いたシミュレーションによって提案手法の性能を計算する以外に方法がない．そのため，正確な評価を行うことは不可能であると考えられるという実験の限界が存在する．よって，正確な評価を行うためには，実証実験を繰り返し行う必要がある．また，今回は各タクシーが独立に最適ルートを選択することを想定している．そのため，提案手法をすべてのタクシーが導入すると需要の食い合いが発生する可能性がある．タクシー全体の利益の最大化も大事な研究テーマと思われるが，その場合マルチエージェント強化学習の枠組みが有効である可能性がある．

謝辞　データを提供してくださったみずほ情報総研株式会社様，およびデータ解析コンペティション関係者の方々に御礼申し上げます．また匿名の査読者から

は大変有用なコメントをいただき，論文を改善することができました．この場をお借りして御礼申し上げます．

参考文献

[1] 一般社団法人全国ハイヤー・タクシー連合会，「TAXI TODAY in Japan 2020」，http://www.taxi-japan.or.jp/pdf/Taxi_Today_2020.pdf（2020 年 7 月 5 日閲覧）

[2] 参議院，「議案情報，第 201 回国会（常会）」，https://www.sangiin.go.jp/japanese/joho1/kousei/gian/201/meisai/m201080201038.htm（2020 年 7 月 5 日閲覧）

[3] 交通政策審議会陸上交通分科会自動車部会，「タクシーサービスの将来ビジョン小委員会報告書」，2006, http://www.mlit.go.jp/singikai/koutusin/rikujou/jidosha/taxi/08/images/03.pdf（2020 年 7 月 5 日閲覧）

[4] C. Gulcehre, T. L. Paine, B. Shahriari, M. Denil, M. Hoffman, H. Soyer, R. Tanburn, S. Kapturowski, N. Rabinowitz, D. Williams, G. Barth-Maron, Z. Wang, N. De. Freitas and Worlds Team, "Making efficient use of demonstrations to solve hard exploration problems," In *ICLR 2020: Eighth International Conference on Learning Representations*, 2020.

[5] C. Y. Goh, J. Dauwels, N. Mitrovic, M. T. Asif, A. Oran and P. Jaillet, "Online map-matching based on hidden Markov model for real-time traffic sensing applications," In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pp. 776–781, 2012.

[6] L. Luo, X. Hou, W. Cai and B. Guo, "Incremental route inference from low-sampling GPS data: An opportunistic approach to online map matching," *Information Sciences*, **512**, pp. 1407–1423, 2020.

[7] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, Bradford Books, 1988.

[8] S. Kapturowski, G. Ostrovski, J. Quan, R. Munos and W. Dabney, "Recurrent experience replay in distributed reinforcement learning," In *ICLR 2019: 7th International Conference on Learning Representations*, 2019.

[9] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra and M. A. Riedmiller, "Playing Atari with deep reinforcement learning," *ArXiv Preprint*, ArXiv:1312.5602, 2013.

[10] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, **529**, pp. 484–489, 2016.

[11] Y. Yin, R. R. Shah and R. Zimmermann, "A general feature-based map matching framework with trajectory simplification," In *Proceedings of the 7th ACM SIGSPATIAL International Workshop on GeoStreaming*, pp. 1–10, 2016.

[12] L. Arge, M. D. Berg, H. Haverkort and K. Yi, "The priority R-tree: A practically efficient and worst-case optimal R-tree," *ACM Transactions on Algorithms (TALG)*, **4**, 1–30, 2008.

[13] P. E. Hart, N. J. Nilsson and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," In *IEEE Transactions on Systems Science and Cybernetics*, **4**, pp. 100–107, 1968.

[14] P. Michalis, B. Francesco, C. Carlos and G. Aristides, "Fast shortest path distance estimation in large networks," In *Proceedings of the 18th ACM conference on Information and knowledge management (CIKM '09)*, 867–876, 2009.

[15] F. Massimo,"PageRank: Standing on the shoulders of giants," *Communications of The ACM*, **54**, pp. 92–101, 2011.

[16] 東京大学空間情報科学研究センター，「人の流れプロジェクト」，https://pflow.csis.u-tokyo.ac.jp/（2020 年 7 月 5 日閲覧）

[17] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye and T. Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pp. 3146–3154, 2017.