長期的な状態依存性を考慮した移動軌跡からの目的地予測

遠藤 結城[1]　西田 京介[1]　戸田 浩之[1]　澤田 宏[1]

概要：本稿では，ユーザの出発地から現在地までの移動軌跡をもとに目的地を予測するデータ駆動型の手法を提案する．移動軌跡は遷移の系列が長く多様なパターンを持つことが多いため，目的地は過去の長期的な状態に左右されやすい．しかし，単純に多次元のマルコフ過程を利用するのは，訓練データのスパースネスが問題となるため効果的ではない．そこで本研究では，これらの問題を緩和できる recurrent neural network (RNN) によって，グリッド空間上における離散的な状態遷移をモデリングする．この RNN モデルを用いてタイムステップごとの遷移確率を計算し，確率的なサンプリングに基づいて各目的地候補への訪問確率を効率的に計算する．タクシーと個人の移動軌跡の実データを用いて提案手法を評価し，直近のタクシー移動の目的地予測コンペティション優勝手法を含む state-of-the-art 手法よりも高精度に目的地を予測できることを示す．

1. はじめに

GPS や Wi-Fi を搭載した携帯電話などの普及に伴い，ユーザの位置情報を**移動軌跡**として容易に収集できるようになった．移動軌跡データは，場所のコンテキストに基づいた店舗情報や広告配信サービスなど，様々な location-based service (LBS) において重要な役割を果たす．しかしながら，単に移動軌跡を利用するのみでは，適切なナビゲーションを行うことが難しいケースが存在する．例えば，駅に向かおうとしているユーザに対しては，現在地周辺のレストランを提示しても煩わしいだけであり，電車の運行状況や時刻表を提示する方が有益である．このような人々の行動を先読みしたサービスを実現するためには，ユーザが訪問する目的地を予測する必要がある．

移動軌跡を扱う従来研究は，位置の変化を捉えやすくするために，移動軌跡をグリッド空間で表現することが多い [15], [28], [29], [30], [37]．特に目的地予測を対象とした従来手法の多くは，移動軌跡履歴における短い遷移を数え上げる低次元のマルコフ過程に基づいて，ユーザの移動をモデリングしていた．しかしながら，様々な状況に応じて移動する人々の移動軌跡は，長く多様なパターンを持つことが多いため，**長期的な状態の依存性**を考慮して目的地を予測することが重要となる．図 1 に例を示す．我々の問題設定は，過去の移動履歴 T^1 や T^2 を訓練データとして利用することで，T_q^1 や T_q^2 などの部分的な問合せ移動軌跡に対するユーザの目的地を予測することである．このため

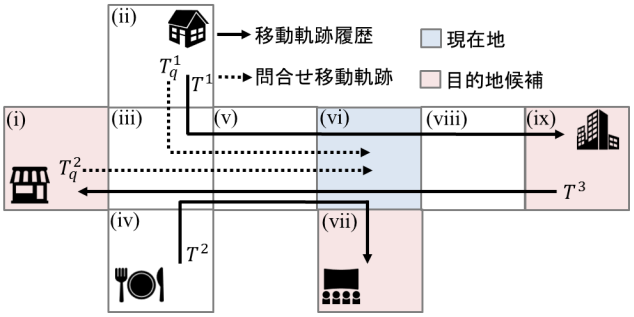


図 1 本研究で取り組む目的地予測の問題設定. 問合せ移動軌跡が所与のもと, 我々の手法は上位 k 個の目的地を, 移動軌跡履歴に基づいて予測する. 例えば問合せ移動軌跡 Tq^1 に対し, グリッド (vi) から 1 ステップごとの予測を繰り返し, 複数の遷移を経て目的地候補であるグリッド (ix) などを予測する.

に，まずは各々のグリッド間の遷移確率を算出する必要がある．従来手法と同様に 1 次マルコフ過程を考えた場合，問合せ移動軌跡 T_q^1 に対する次ステップでの遷移確率は $P(\mathrm{v}|T_q^1) = P(\mathrm{vii}|T_q^1) = P(\mathrm{viii}|T_q^1) = 0.33$ となる．これに対し，2 次マルコフ過程を用いると，ユーザがどの方向から来たかを考慮できるため，$P(\mathrm{vii}|T_q^1) = P(\mathrm{viii}|T_q^1) = 0.5$ と絞り込むことができる．さらに 4 次マルコフ過程を用いると，どこを出発したかを考慮できるため，$P(\mathrm{viii}|T_q^1) = 1$ と移動先を特定できる．しかしながら，高次元のマルコフ過程を用いるのは，**訓練データのスパースネス**が問題となる．例えば T_q^2 のように過去の履歴とは異なる地点を出発した場合，4 次マルコフ過程では該当する遷移が存在しないため，$P(\mathrm{vii}|T_q^2)$ を計算できない．これらの問題は，個

[1] NTT サービスエボリューション研究所

人の移動軌跡に限らず，タクシーのような不特定多数の移動軌跡を対象とした場合においても存在する．

そこで本稿では，長期的な状態の依存性を考慮し，かつデータスパースネスの影響を受けにくい目的地予測手法を提案する．これを実現するために，我々は deep neural network (DNN) の一種である recurrent neural network (RNN) [22] を採用する．RNN は中間状態の特徴ベクトルを循環させながら，長期にわたる系列情報を記憶できるネットワークであり，近年文書生成などのタスクに多く応用されている [11], [18], [25]．RNN を移動軌跡に適用する場合，移動軌跡をどのように学習可能な特徴形式で表現するかが問題になる．提案手法は移動軌跡をグリッド空間上で扱い，グリッド系列をバイナリ特徴系列として表現して RNN に学習させることで，ステップごとの遷移確率を予測するモデルを構築する．この際 RNN は，系列の情報を連続的な空間において表現された統計的な重みとして記憶するため，マルコフ過程のような遷移を数え上げる方法とは異なり，データスパースネスの問題を緩和できる．すなわち T_q^2 のような，入力系列の遷移と完全に一致する遷移を持つ移動軌跡が履歴に存在しなくとも，予測が可能となる．

RNN によってユーザの動きをモデリングした後は，RNN エンコーダデコーダ [11], [24] の枠組みを用いて目的地を予測する．この際，現在地から目的地への移動は複数ステップの遷移を経由する必要がある一方で，RNN は 1 ステップごとの遷移を予測するものであることが問題となる．これをふまえた上で目的地を予測する直接的な方法は，各目的地候補に対して全ての経路を経由する確率を RNN から算出・周辺化し，各々への訪問確率を算出することであるが，計算量が膨大になり非現実的である．我々はこの問題に対して，RNN を用いた文書生成から発想を得ることで，効率的に目的地を予測するアルゴリズムを開発した．具体的には，RNN から得られる遷移確率に基づく確率的なサンプリングによってユーザの移動をシミュレーションする．これを複数回行い，各試行において訪問された目的地候補に投票することで，各々への訪問確率を算出する．

本研究の貢献をまとめると以下のようになる：

- 長期的な状態の依存性の考慮およびデータスパースネス問題を緩和可能な RNN を用いた移動軌跡のモデリング
- 投票ベースのサンプリングシミュレーションを用いた RNN エンコーダデコーダの枠組みに基づく目的地予測アルゴリズムの提案
- 不特定多数および個人の実データを利用した評価を通じた提案手法の有効性の確認

2. 関連研究

移動軌跡から目的地を予測する手法として，移動軌跡に加えて外部情報を利用するものが提案されている．例えば，事前調査によって得られた車の移動時間の分布を利用した手法 [14], [15], [16] や，事故情報や道路状況を利用した手法 [37] が存在する．このような外部情報を利用する手法は効果的な一方で収集にコストがかかるため，本研究では対象外とする．また，個人の行動をモデリングすることで目的地を予測するアプローチとして，ベイジアン法や [17], [21] マルコフモデルに基づいたもの [1], [2] が存在する．これに対し，本研究では個人に限らず不特定多数の移動軌跡も扱えるより汎用的な手法を提案する．他にも距離基準を用いた類似軌跡 (nearest neighbor trajectory) の探索 [27] や，木構造に基づく軌跡のマッチング [3] など，マッチングに基づく手法も提案されているものの，データスパースネス問題への対処が十分でない．

上記のような外部情報を用いる手法に対し，近年 Xue ら [29], [30] は移動軌跡の履歴のみを利用し，かつ対象を個人のユーザに絞らない汎用的な目的地予測のタスクを定めた．本タスクにおいて，彼らは訓練データのスパースネスを問題を解決することに主眼を置いた *Sub-Trajectory Synthesis* (SubSyn) アルゴリズムを提案した．SubSyn アルゴリズムは，まずグリッド空間上で定義された移動軌跡を部分遷移系列とみなすことで，ユーザの移動を 1 次または 2 次マルコフ過程によりモデリングする．次に，このモデルを利用することで迂回路をふまえた各グリッドへの遷移確率を算出する．それから，算出した遷移確率を用いることで問合せ移動軌跡の出発地および現在地から各目的地候補への移動確率を計算する．最後にこれらの移動確率を用いることで，ベイズ推定に基づいて各目的地候補への訪問確率を推定する．彼らの手法は移動軌跡を部分遷移系列として捉えることで，データスパースネスに上手く対処することが可能である．一方で，低次元のマルコフ過程では長い遷移系列を持つ移動軌跡をモデリングするには不十分である．また，予測時において出発地と現在地の間の情報が使われていないことが，予測精度に影響を与えてしまう．

Brébisson ら [8] はニューラルネットワークを利用したアプローチを考案し，タクシーの目的地予測のコンペティション [*1] において最も高い性能を示した．彼らは予測される目的地の位置座標が，過去に多く訪問された目的地の位置座標の線形和で表せると仮定し，これを多層パーセプトロン (MLP) を用いた回帰問題として解いた．具体的には，まず，訓練データの目的地を密度ベース手法でクラスタリングする．次に，移動軌跡とタクシー ID などのメタデータを MLP の入力として与え，出力としてクラスタ重心の和が目的地となるようにソフトマックス層を用いてモデリングする．MLP に与える入力ベクトルの次元は固定する必要があるため，移動軌跡は最初と最後の 5 個の測位点の

[*1] http://www.geolink.pt/ecmlpkdd2015-challenge/

表 1　各手法の課題への対処有無を示す表．ここで, '+', '±' および '−' はどの程度課題に対処しているかを表す.

手法	長期的な状態依存性		データ
	学習	予測	スパースネス
低次元マルコフ過程	−	−	+
多次元マルコフ過程	+	+	−
SubSyn [29], [30]	−	±*1	+
MLP [8]	±*1	±*1	−
提案手法	+	+	+

*1: 出発地と現在地に基づく予測は, 低次元のマルコフ過程よりも長期的な状態依存性の課題に対処しているものの, 全系列を使えているわけではない.

緯度経度を入力としている．MLP モデルを構築する際には，訓練データにおける真の目的地と推定されたクラスタの線形和との距離が最小になるように backpropagation でモデルパラメータを最適化する．しかし，彼らの手法は密度ベースのクラスタリングに基づいているため，訓練データが少ないと十分な性能を発揮できない．さらに，SubSyn アルゴリズムや提案手法とは異なり，彼らの手法は目的地への訪問確率値 (すなわち複数のランキングされた候補) を得られるわけではなく，単一の目的地を得るものである．実サービスにおいては，前者の方がアプリケーションの拡張性を期待できる．

また，彼らは MLP よりも高い性能を示すことはできなかったものの，RNN を用いた結果も示している．RNN を用いることで，移動軌跡におけるすべての測位点を利用できる利点がある．彼らは MLP の方法と同様に，RNN によって得られた中間状態を用いて，クラスタの線形和となる目的地を予測した．我々も RNN を用いるが，彼らの手法とは主に三つの違いがある．一つ目に，提案手法は離散的な空間におけるユーザの移動を，遷移として陽にモデリングする．これが長期的な状態依存性に対するモデルの表現力を向上させ，高精度な目的地予測を可能とする．二つ目に，MLP を含む彼らの手法は入力系列から直接一つの目的地を予測する sequence-to-point のモデリングであるのに対して，我々の手法は系列から系列を予測する sequence-to-sequence のモデリングである．これにより，単一の移動軌跡から一つの正解データしか得られない既存手法に対して，我々の手法は単一の移動軌跡から各測位点と対応する次の測位点として，より多くの正解データを利用していると考えることができる．三つ目に，我々の手法は多項分布として次の遷移先をモデリングすることで，確率値として目的地を算出し，ランキングされた複数の結果を得ることができる．

表 1 に一般的なマルコフモデル，最新手法，および提案手法が 1 節で述べた各課題に対処しているか否かをまとめる．

3. 準備

3.1 Recurrent neural network (RNN)

RNN は中間層における状態ベクトルを循環させて処理することで，長期的な系列情報をモデリングできる技術である．RNN によって入力の系列情報から中間状態における特徴表現を獲得することをエンコードと呼び，エンコードされた特徴表現は，デコードによって文書生成 [11], [18], [25] や自動翻訳 [6], [26]，および画像の説明文生成 [4] などに応用されている．一般的なニューラルネットワークに対しては back propagation (BP) によって誤差関数の勾配を深い層に伝播させてモデルパラメータを最適化するのに対し，RNN に対しては時間軸を逆行しながら BP と似た枠組み，すなわち back propagation through time (BPTT) [23] を利用する．BPTT による RNN モデルの学習は非常に深いネットワークの学習に相当するため，膨大な計算量が必要とされることが多い．この問題に対する解決法として，時間軸を遡るステップ数を固定して BP を行う truncated BPTT [38] が提案され，長い系列のデータに対してより効率的にモデルパラメータを最適化できるようになった．

Long short term memory (LSTM). しかしながら，通常の RNN はモデル学習時の誤差関数の勾配消失問題により，10 程度までの系列を記憶することしかできなかった．これに対して，循環する中間層を LSTM に置き換えることで，この問題を緩和できることが知られている [10], [13]．LSTM は cell と呼ばれるユニットを利用し，学習時に過去の誤差情報を記憶させておくことで，1000 以上の系列を記憶できる．この際，複数のゲートが cell に対する情報の流れを制御している．特に forget gate は不必要な記憶を cell に蓄積せずにリセットすることで，効率的な系列情報の記憶を実現している [10]．具体的には，モデルパラメータ θ_{LSTM} を持つ LSTM が，タイムステップ t におけるデータの特徴ベクトル $\mathbf{x}_t$ と，$t-1$ における状態ベクトル $\mathbf{h}_{t-1}, \mathbf{c}_{t-1}$ とを用いて，t における状態ベクトル $\mathbf{h}_t, \mathbf{c}_t$ を出力する:

$$\mathrm{LSTM}(\mathbf{x}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}; \theta_{LSTM}) \rightarrow \mathbf{h}_t, \mathbf{c}_t, \tag{1}$$

$$\begin{aligned}
\mathbf{i}_t &= \mathrm{sigm}(W_{ix}\mathbf{x}_t + W_{ih}\mathbf{h}_{t-1} + \mathbf{b}_i),\\
\mathbf{f}_t &= \mathrm{sigm}(W_{fx}\mathbf{x}_t + W_{fh}\mathbf{h}_{t-1} + \mathbf{b}_f),\\
\mathbf{c}_t &= \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tanh(W_{cx}\mathbf{x}_t + W_{ch}\mathbf{h}_{t-1} + \mathbf{b}_c),\\
\mathbf{o}_t &= \mathrm{sigm}(W_{ox}\mathbf{x}_t + W_{oh}\mathbf{h}_{t-1} + W_{oc}\mathbf{c}_{t-1} + \mathbf{b}_o),\\
\mathbf{h}_t &= \mathbf{o}_t \circ \tanh(\mathbf{c}_t).
\end{aligned}$$

ここで，添え字のついた W および $\mathbf{b}$ はそれぞれ重み行列およびバイアスベクトルであり，モデルパラメータ θ_{LSTM} のサブセットになる．$\circ$ はベクトルの要素ごとの積を計算する演算子であり，$\mathrm{sigm}(\cdot)$ および $\tanh(\cdot)$ はそれぞれベクトルの各要素にシグモイド関数およびハイパボリックタン

ジェント関数を適用する．最近では LSTM の代替として gated recurrent unit (GRU) [5] も提案され，LSTM と同様の有効性が確認されている [7]．本研究においては，予備実験の結果から良好な結果を確認した LSTM を利用する．

3.2 問題設定

本稿で扱う用語と問題の定義を述べる．主要な記号は表 2 にもまとめる．

定義 1 (移動軌跡履歴): 移動軌跡履歴 D は出発地点から目的地点までの生の移動軌跡 T の集合である．T は経度 x，緯度 y，そしてタイムスタンプ τ から構成される e_T 個の測位点 p の系列で表される:

$$T = [p_1, p_2, ..., p_{e_T}] = [< x_1, y_1, \tau_1 >, < x_2, y_2, \tau_2 >, ..., < x_{e_T}, y_{e_T}, \tau_{e_T} >].$$

ここで $\forall_{1 \leq t < e_T} \tau_t < \tau_{t+1}$ であり，移動軌跡 T における測位点 p はタイムスタンプの古い順に並んでいるものとする．D は後述する RNN モデルに対する訓練データとして用いられる．

本稿で用いるデータセットでは，タクシーの乗車から降車の移動軌跡，あるいは，パーソナルユーザが一定時間以上滞留した点から，次の滞留点への移動軌跡が該当する．

定義 2 (問合せ移動軌跡): 問合せ移動軌跡 T_q は出発地点のタイムスタンプ τ_1 から現在地点のタイムスタンプ τ_c までの測位点 p の系列で表される:

$$T_q = [p_1, p_2, ..., p_c] = [< x_1, y_1, \tau_1 >, < x_2, y_2, \tau_2 >, ..., < x_c, y_c, \tau_c >].$$

ここで $\forall_{1 \leq t < c} \tau_t < \tau_{t+1}$ である．T_q は後述する目的地予測における入力として用いられる．

定義 3 (グリッド空間): グリッド空間は離散化された地理範囲であり，$|G|$ 個のグリッドからなる．各々のグリッドはグリッド ID として $g \in G$ $(1 \leq g \leq |G|)$ が割り当てられている．

定義 4 (目的地): 目的地 d は特定の ID であり各グリッドの位置と対応する．

本稿で用いるデータセットでは，タクシーの降車位置あるいはパーソナルユーザが一定時間滞留した点のグリッド ID が該当する．

定義 5 (目的地候補群): 目的地候補群 C はユーザが訪問する可能性のある $|C|$ 種類の目的地 d の集合を表す．

定義 6 (目的地訪問確率): 目的地訪問確率 $\mathbf{P} \in [0,1]^{|C|}$ は各目的地候補 $d \in C$ に訪問する確率を表す．

問題定義: 本稿において扱うタスクは，入力として問合せ移動軌跡 T_q が与えられた際に，各目的地候補 $d \in C$ に訪問する確率値をベクトル $\mathbf{P}$ として出力するものとする．移動軌跡は $|G|$ 種類のグリッド ID が割り当てられたグリッド空間上で表現される．目的地予測を実現するために予め与えられるデータは，移動軌跡の履歴 D のみであり，D が個人のものか集団のものかは区別しない．

なお，出力を確率値とした理由は，最も訪問する可能性の高い一つの目的地のみを出力するよりも，確率値によって全ての目的地候補に順位付けできる方が応用の幅が広がるためである．

表 2 主要な記号の説明.

記号	説明
D	学習用の移動軌跡履歴
T	D に含まれる移動軌跡
T_q	問合せ移動軌跡
p	T や T_q に含まれる測位点
g	$\|G\|$ 個のグリッドからなるグリッド空間に割り当てられた ID
d	$\|C\|$ 個の目的地候補に対する各目的地に割り当てられた ID
$\mathbf{P}$	目的地訪問確率
$\mathbf{g}$	One-hot 表現
$\mathbf{h}$, $\mathbf{c}$	LSTM における隠れ状態およびセル状態
$\mathbf{p}$	RNN によって得られる遷移確率
n	RNN の各層のニューロン数
θ, θ_{LSTM}	RNN および LSTM のモデルパラメータ
N	目的地予測におけるシミュレーション回数
M	シミュレーションにおけるサンプリング回数

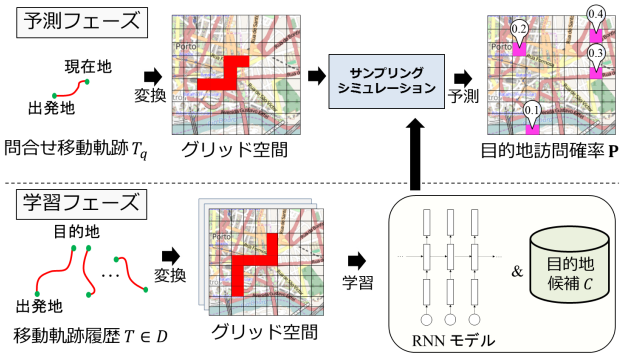


図 2 提案手法の処理の流れ.

4. 手法

図 2 に示すように，提案システムの処理は大きく分けて学習と予測の二つのフェーズから構成される．学習フェーズ (4.1 節) においては，ユーザの移動軌跡履歴 D を入力として，目的地予測を行うための RNN のモデルパラメータ θ を学習し，目的地候補群 C を作成する．予測フェーズ (4.2 節) においては，現在地までの問合せ移動軌跡 T_q を θ に基づく RNN モデルの入力として，各目的地候補 $d \in C$ に対する未来の訪問確率 $\mathbf{P}$ を出力する．

4.1 モデル

本節では，まず移動軌跡への適用をふまえた RNN の構造を簡潔に説明した後，RNN モデルに移動軌跡履歴を学習させる方法について述べる．

4.1.1 ネットワーク構造

図 3(a) に示すように，我々が用いる RNN は，入力と

してタイムステップ t におけるユーザのグリッド位置を表す $\mathbf{g}_t$，および $t-1$ における二つの状態ベクトル $\mathbf{h}_{t-1}$, $\mathbf{c}_{t-1}$(RNN の中間表現) を受け取る．結果，学習済みのモデルパラメータ $\theta = \{W_e, \theta_{LSTM}, W_s, \mathbf{b}_s\}$ が所与のもと，出力として $t+1$ における各グリッドへの移動確率 $\mathbf{p}_{t+1}$，および t における二つの状態ベクトル $\mathbf{h}_t$, $\mathbf{c}_t$ を出力する:

$$\mathrm{RNN}(\mathbf{g}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}; \theta) \rightarrow \mathbf{p}_{t+1}, \mathbf{h}_t, \mathbf{c}_t. \quad (2)$$

$$\mathbf{p}_t = P(g_t | g_{t-1}, g_{t-2}, ..., g_1). \quad (3)$$

以降では，移動軌跡を RNN に適用する際の個々の処理について述べる．

移動軌跡の one-hot 表現． 場所から場所への遷移を陽にモデリングするために，提案手法は離散化された空間において移動軌跡を扱う．すなわち，グリッドからグリッドへの遷移を RNN モデルに学習させる．代替のアプローチとして，緯度経度の二つのスカラー値の系列を直接 RNN モデルに学習させることも可能だが，データの分布に関する事前情報が不明な状態で，そのようなシンプルな方法で移動をモデリングするのは簡単ではない [8]．提案手法においては，移動軌跡の各測位点が特定のグリッドに存在するという情報を one-hot 表現として扱う．one-hot 表現とは，多次元の要素のうち一つだけが 1 で他が 0 であるバイナリベクトルであり，言語モデルにおいても良く用いられている [31]．提案手法においては，移動軌跡 $T \in D$ における各測位点の経度 $p.x$，緯度 $p.y$ とグリッドとの対応づけを行い，次式の通り one-hot 表現のベクトル $\mathbf{g} \in \{0,1\}^{|G|}$ に対して，グリッド ID の g と対応する要素には 1 を，それ以外の要素には 0 を割り当てる:

$$\mathbf{g}_t = \mathrm{G}_o(g_t) = [0, ..., 0, \underset{\text{at index } g_t}{1}, 0, ..., 0]. \quad (4)$$

特徴埋め込み． 前述の処理によって得られた one-hot 表現 $\mathbf{g}$ は，グリッド数からなる高次元のベクトルである．このため，one-hot 表現を直接 RNN の中間層への入力とする場合，モデル学習時の過学習や計算量の増加が問題となる．この問題を緩和するため，予め $\mathbf{g}$ を低次元のベクトル空間に埋め込み，より密なベクトル $\mathbf{x} \in \mathbb{R}^n$ に変換する:

$$\mathbf{x}_t = \mathrm{G}_e(\mathbf{g}_t) = W_e \mathbf{g}_t. \quad (5)$$

ここで，$W_e \in \mathbb{R}^{n \times |G|}$ はベクトルを埋め込むための重み行列である．また，n は埋め込み先のベクトル空間の次元数を表す．式 (5) においては，行列 W_e と $\mathbf{g}_t$ の積を直接計算する必要はない．なぜならば，$\mathbf{g}_t$ はバイナリベクトルかつ，一つの非 0 要素のみを持つため，非 0 要素と対応する W_e の列ベクトルを $\mathbf{x}_t$ として選択すれば良いためである．つまり，n や $|G|$ に対する特徴埋め込みの計算量は $O(1)$ となる．

隠れ層． 3.1 節において紹介した LSTM (式 (1)) を用いることで，埋め込まれた特徴ベクトル $\mathbf{x}_t$ と状態ベクトル $\mathbf{h}_{t-1} \in \mathbb{R}^n$, $\mathbf{c}_{t-1} \in \mathbb{R}^n$ から，タイムステップ t における状態ベクトル $\mathbf{h}_t$, $\mathbf{c}_t$ を算出する．LSTM を適用する際は，循環させる入力とは異なる入力 (すなわち $\mathbf{x}_t$) のみに dropout [12] を適用すると，効果的にモデルの汎化能力を向上させられると報告されている [32]．そのため，本稿でも同様の方法で dropout を適用する．

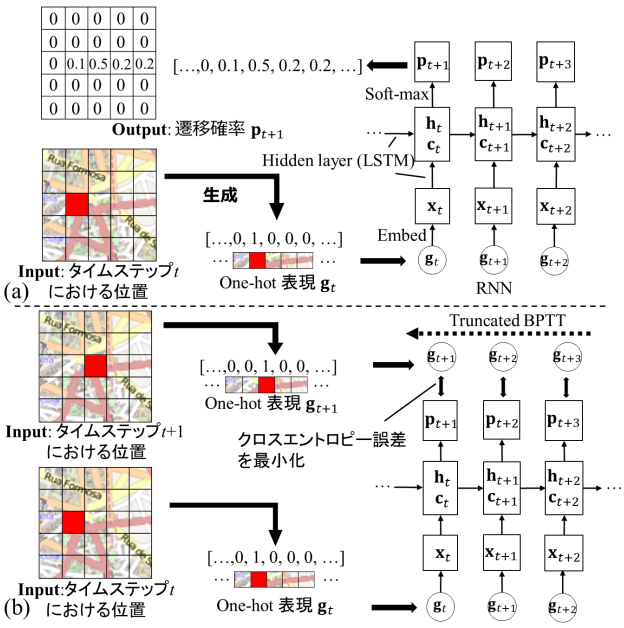


図 3 RNN を用いた移動軌跡における遷移のモデリング．(a) RNN は次のタイムステップにおける遷移確率を予測する．(b) RNN モデルを構築するために，次ステップにおける真の位置と予測される遷移確率とのクロスエントロピー誤差が最小になるように，モデルパラメータを最適化する．

Soft-max 層． LSTM によって得られた中間表現 $\mathbf{h}_t$ から，soft-max 層を経て次のステップにおける各グリッドへの移動確率 $\mathbf{p}_{t+1} \in \mathbb{R}^{|G|}$(図 3(a) 左上参照) を算出する:

$$\mathbf{p}_{t+1} = \mathrm{G}_s(\mathbf{h}_t) = \mathrm{softmax}(W_s \mathbf{h}_t + \mathbf{b}_s). \quad (6)$$

ここで，$W_s \in \mathbb{R}^{|G| \times n}$ および $\mathbf{b}_s \in \mathbb{R}^{|G|}$ はそれぞれ重み行列とバイアスを表し，softmax(·) はベクトルの各要素にソフトマックス関数を適用する．

4.1.2 学習

移動軌跡履歴 D を用いて目的地候補群 C および RNN のモデルパラメータ θ を学習する．目的地候補 C は各移動軌跡 $T \in D$ における最後のタイムステップ τ_{e_T} の測位点にあたるグリッドに存在するものとし，グリッドと対応する目的地候補 ID である d を抽出する．

RNN のモデルパラメータ θ を推定するため，移動軌跡 T のタイムステップ t における測位点の存在するグリッド $\mathbf{g}_t$ の入力に対する正解データとして，$t+1$ における測位点の存在するグリッド $\mathbf{g}_{t+1}$ を与える (図 3(b) 参照)．正解データが所与のもと，θ は次式の条件付き尤度を最大化することで推定される:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{T \in D} \prod_{t=1}^{e_T - 1} P(g_{t+1} | g_t, g_{t-1}, ..., g_1; \theta). \quad (7)$$

これは，$t = 2, 3, ..., e_T$ において予測確率 $\mathbf{p}_t$ と one-hot 表現 $\mathbf{g}_t$ とのクロスエントロピー誤差を最小化することと等価であり，truncated BPTT [38] によってこれを解く．最適化の前には，θ を一様分布に従う乱数で初期化しておく．一様分布の範囲は経験的に $[-0.08, 0.08]$ とした．最適化手法には収束速度の速さから mini-batch AdaDelta [33] アルゴリズムを利用する．我々の実験においては，truncated BPTT により過去の系列を遡って勾配を伝播させる系列数を 20 とし，mini-batch のサイズを 100 に固定した．また，勾配消失および爆発問題を緩和するため，mini-batch のサイズで正規化された勾配のノルムが 5 より大きくなった場合，勾配をクリッピングする [20]．

Mini-batch 単位でパラメータを更新する場合，移動軌跡 $T \in D$ における測位点の数が T によって異なることが最適化の効率性に対する問題となる．つまり，T の系列長がそれぞれ異なることから，T ごとにモデルの学習を行なわなければならず，これがパラメータの更新時間を増加させてしまう．我々は計算効率を重視するために，訓練データにおける全ての移動軌跡 T を一つの系列として繋ぎ合わせた T' を生成し，T' を均等に分割することで mini-batch 学習を行う．すなわち，次式の通りモデルパラメータを最適化することになる:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \prod_{t=1}^{e_{T'} - 1} P(g_{t+1} | g_t, g_{t-1}, ..., g_1; \theta). \quad (8)$$

この際，各々の T の終端において本来存在しない次の遷移を学習することになるものの，次節で述べるサンプリングシミュレーションにより目的地までの遷移しか考慮しないため，これは大きな問題にはならない．

4.2 予測

訓練データから推定したパラメータ θ を持つ RNN モデルを用いて，問合せ移動軌跡 T_q から各目的地候補 C への訪問可能性を算出する．このために，RNN エンコーダデコーダの枠組みを利用する．まずはエンコーダを用いて出発地 p_1 から現在地 p_c まで移動した際の状態ベクトル $\mathbf{h}_c$ および $\mathbf{c}_c$ を求める．具体的には，T_q の測位点群から古い順にグリッド ID および one-hot 表現を計算し，前節で学習した RNN モデルに one-hot 表現と前ステップの状態ベクトルを与え，状態ベクトルを繰り返し更新していくことで $\mathbf{h}_c$ および $\mathbf{c}_c$ を計算できる:

$$\mathbf{h}_t, \mathbf{c}_t \leftarrow \mathrm{LSTM}(\mathrm{G}_e(\mathbf{g}_t), \mathbf{h}_{t-1}, \mathbf{c}_{t-1}; \theta_{LSTM}), \quad t = 1, ..., c.$$

ここで，$\mathbf{h}_0$ および $\mathbf{c}_0$ は初期の状態ベクトルであるためゼロベクトルで初期化する．

次にデコーダを利用して，状態ベクトル $\mathbf{h}_c$ および $\mathbf{c}_c$ から，各目的地候補 C への訪問確率 $\mathbf{P}$ を算出する．1 章において述べた通り，グリッド系列から学習した RNN モデルを単純に用いた場合，入力の問合せ移動軌跡 T_q に対して，一つ先のタイムステップにおける移動先のグリッド，すなわち $\mathrm{G_s}(\mathbf{h}_c) = P(g_{c+1}|T_q)$ を予測することになる．一方で，目的地候補は現在地から複数のグリッドを経由して到達するものがほとんどであるため，この使い方では目的地訪問確率 $\mathbf{P}$ を計算できない．

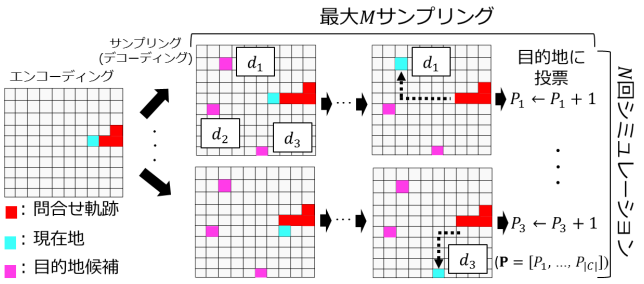


図 4　目的地予測のための RNN エンコーダデコーダを利用したサンプリングシミュレーションの流れ．

これを解決する最も直接的な方法としては，目的地へ到達するまでに考えられうる全ての遷移に対する確率を RNN モデルによって算出し，周辺化することである:

$$\sum_{\forall g_{c+m-1}} ... \sum_{\forall g_{c+1}} \prod_{t=c}^{c+m-1} P(g_{t+1} | g_t, g_{t-1}, ..., g_{c+1}, T_q; \theta).$$

従来の 1 次マルコフや 2 次マルコフを考慮した手法 [29], [30] であれば，短期的な過去の状態遷移しか考慮していないため，全ての遷移確率を実時間で計算することも難しくない．しかし，RNN は出発時点までの過去全ての遷移を遡って考慮することで次のグリッドへの遷移確率を算出するため，目的地に M ステップでたどり着くとすると，$|G|^{M-1}$ 通りの遷移確率を RNN で計算する必要がある．例えば，グリッド数 $|G|$ を数百，ステップ数 M を数十とした場合，現実的な計算時間で処理することが難しくなる．特にステップ数 M は迂回路も考慮する必要があるため，現在地から目的地へ到達するまでの最短ステップより大きな値を設定する必要がある [30]．あるいは，仮に一度の遷移で縦・横・斜めの隣接グリッドにしか移動しないという制約を設けたとしても，計算量は $O(8^{M-1})$ となり M が数十になると膨大な処理時間を要してしまう．

上記の問題をふまえて，我々は RNN を用いた確率的なサンプリングに基づく文書生成 [11] から発想を得ることで，目的地候補への訪問確率を効率的に推定するアルゴリズムを提案する．図 4 に示すように，提案手法は RNN から得られる遷移確率に基づいたサンプリングによって，問合せ移動軌跡の現在地から未来の目的地までの移動をシミュレーションする．具体的には，最初に $P(g_{c+1}|T_q)$ に従ってサンプリングし，次に $P(g_{c+2}|g_{c+1}, T_q)$ に従ってサンプリングする．この過程を繰り返し，目的地候補に到達

したら，目的地訪問確率ベクトル $\mathbf{P}$ の対応する目的地の要素に投票，すなわち定数を加算する．この際，シミュレーションにおけるサンプリング回数は，ユーザが目的地へ到達するまでに必要とするステップ数 M を上限とし，もし M 回以内のサンプリングでどの目的地候補にも到達しない場合は，$\mathbf{P}$ に何も加算せず試行を終了する．このシミュレーション試行を複数 (N) 回繰り返すことで，全ての遷移の可能性を考える必要なく，各目的地候補に対する訪問可能性を近似的に求めることができる．最後に，シミュレーション処理によって計算された $\mathbf{P}$ を正規化することで，これを確率ベクトルとして表現できる．以上のシミュレーションにおける各試行は簡単に並列化することができる．

サンプリングを行うために現在地から目的地への移動に要する最大ステップ数 M を決定する必要がある．従来研究においては，M に最短経路の遷移数の 1.2 倍の値を設定すれば，迂回路を適切に考慮できることが報告されている [30]．しかしながら我々の手法においては，シミュレーション試行が終了するまでどの目的地候補に到達するか明らかにはならない．そのため，提案手法では移動軌跡履歴 D に存在する移動軌跡のうち，最も長いグリッド長を M とする．実際は，サンプルが目的地候補に到達した時点で処理が終了するため，M 回のサンプリングが行われることは少ない．シミュレーション回数 N に関しては，特に明記のない限り我々の実験では 100 を用いた．

4.2.1 空間的近接性の考慮

実際の移動軌跡の特徴を考えると，突然遠くの場所に遷移する可能性は低いことから，サンプリングの際に，グリッド間の空間的な近さを考慮することが重要となる．もしこれを考慮しない場合，サンプルが物理的に起こり得ない距離のグリッドにジャンプしてしまう可能性がある．なぜならば RNN モデルから得られる遷移確率は，モデルの学習度合や訓練データのノイズの影響を受けるためである．そこで距離の情報を考慮するために，遷移確率をグリッドの空間的近接性に基づいて修正する．具体的には，サンプルが遠く離れたグリッドへ 1 ステップで移動する確率を下げるため，RNN デコーダによって得られる遷移確率 $\mathbf{p}_t$ を更新し，新たな遷移確率 $\mathbf{p}'$ を得る:

$$\mathbf{p}'_t = \frac{\mathbf{p}_t \circ \mathbf{s}_{g_{t-1}}}{|\mathbf{p}_t \circ \mathbf{s}_{g_{t-1}}|}, \tag{9}$$

$$\mathbf{s}_g = [\exp(-\frac{\mathrm{Dist}(g,1)}{\sigma}), ..., \exp(-\frac{\mathrm{Dist}(g,|G|)}{\sigma})].$$

ここで，$\mathrm{Dist}(\cdot,\cdot)$ は二つのグリッド間の距離 (メートル) を表し，σ はグリッド間の距離がサンプルのジャンプに寄与する度合を決定する分散パラメータである．σ が小さいほど，サンプルは遠くのグリッドにジャンプしにくくなる．これにより，予測する目的地を大きく外すのを避けることができる．σ の値は本実験において利用するグリッドのサイズ $(150m \times 150m)$ に基づいて，経験的に $200m$ とした．

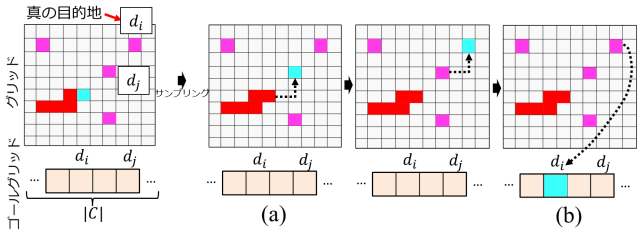


図 5 仮想的なゴールグリッドを用いた移動状態と訪問状態の区別．(a) 通常のグリッド空間においては，サンプリングが真の目的地への移動中に他の目的地候補で停止してしまうことが問題になる．(b) 目的地候補と対応する仮想的なゴールグリッドを用意し，サンプルを移動中のグリッドから空間的近接性に基づいて，近くのゴールグリッドへジャンプさせることで，この問題を回避する．

4.2.2 移動状態と訪問状態の区別

目的地候補への到達までの遷移をシミュレーションによって計算する場合，その経由地に別の目的地候補が存在することが問題となる．つまり，図 5(a) に示すように，サンプリング処理は目的地候補のどれかに到達したら停止するため，現在地から遠い目的地ほど到達しにくくなってしまう．

この問題に対しては，図 5(b) に示すように，仮想的なゴールとして目的地候補のグリッドを別に用意することで対処する．これにより，移動状態と訪問状態を区別できるようになるため，サンプリングの過程で目的地が存在する経由地に到達しても，処理が中断されなくなる．最終的に，RNN モデルが学習した適切な目的地のノードへ，目的地周辺のグリッドノードからサンプルがジャンプすることでシミュレーションが終了する．ゴールノードの追加に伴い，目的地候補群の生成や RNN モデルの学習を修正する必要がある．このために，まずグリッド ID については，通常のグリッド ID に加えて，新たな目的地候補と対応するゴールグリッドの ID が必要となる．すなわち，訓練データにおいて目的地候補が $|C|$ 個存在する場合，目的値の ID は $|G|+1 \leq d \leq |G|+|C|$ の範囲で定義され，抽出される one-hot 表現の次元も $|G|+|C|$ となる．加えて，RNN の出力である確率ベクトル，soft-max layer における重み行列およびバイアスはそれぞれ $\mathbf{p} \in [0,1]^{|G|+|C|}$, $W_s \in \mathbb{R}^{(|G|+|C|)\times n}$ および $\mathbf{b}_s \in \mathbb{R}^{|G|+|C|}$ として定義される．

最後に Algorithm 1 に RNN エンコーダデコーダを利用した目的地予測アルゴリズムをまとめる．

5. 実験

本節では評価実験を通じて，1 節で列挙した課題について提案手法の有効性を検証する．評価実験においては，2 節で述べた低次元のマルコフ過程に基づく最新手法である SubSyn アルゴリズム [30]，ECML PKDD のコンペ

表 3　利用データセットの地理範囲と統計量.

データセット	緯度範囲	経度範囲	移動軌跡数	平均距離 (m)	平均グリッド長
TST	[41.131571, 41.162477]	[-8.613876, -8.565437]	154,616	2127.4 ± 1130.0	14.8 ± 7.0
GL	[39.68, 40.19]	[116.05, 116.72]	8,390 (in 46 users)	5258.1 ± 8172.7	13.3 ± 13.8

*グリッド長は移動軌跡が存在するグリッド数を表す.

Algorithm 1 DestinationPrediction

Input: Query trajectory $T_q = [p_1, p_2, ..., p_c]$,
destination candidates C, and learned model parameter θ
Output: Destination visiting probability $\mathbf{P}$

$\mathbf{P} \leftarrow \mathbf{0}$, $\mathbf{h}_0 \leftarrow \mathbf{0}$, $\mathbf{c}_0 \leftarrow \mathbf{0}$;
for $t = 1$ **to** c **do**
  $g_t \leftarrow \text{ComputeGridID}(p_t)$;
  $\mathbf{h}_t, \mathbf{c}_t \leftarrow \text{LSTM}(\text{G}_e(\text{G}_o(g_t)), \mathbf{h}_{t-1}, \mathbf{c}_{t-1}; \theta_{LSTM})$;
endfor
$\mathbf{p}_{c+1} = \text{G}_\text{s}(\mathbf{h}_c)$;
for $n = 1$ to N **do**
  for $t = 1$ to M **do**
    $\mathbf{p}'_{c+t} \leftarrow \frac{\mathbf{p}_{c+t} \circ \mathbf{s}_{g_{c+t-1}}}{|\mathbf{p}_{c+t} \circ \mathbf{s}_{g_{c+t-1}}|}$;
    $g_{c+t} \leftarrow \text{RandomSampling}(\mathbf{p}'_{c+t})$;
    if $g_{c+t} \in C$
      $\mathbf{P} \leftarrow \text{Vote}(g_{c+t}, \mathbf{P})$;
      break;
    endif
    $\mathbf{p}_{c+t+1}, \mathbf{h}_{c+t}, \mathbf{c}_{c+t} \leftarrow \text{RNN}(\text{G}_o(g_{c+t}), \mathbf{h}_{c+t-1}, \mathbf{c}_{c+t-1}; \theta)$;
  endfor
endfor
$\mathbf{P} \leftarrow \mathbf{P} / \sum_{i=1}^{|C|} P_i$;

ティションにおいて最高精度を達成した MLP に基づく手法 [8][*2]，および提案手法とを比較した．以降では，まず実験に用いた実装の詳細を述べ，次に本実験で利用するデータセットを紹介する．それから実験設定を述べ，最後に実験結果を示し議論する．

5.1　実装の詳細

我々は本実験を Intel Xeon X5650 2.66GHz CPU, 48GB RAM が搭載された PC 環境で行った．手法の実装には Python を用い，特に RNN の実装には Chainer ライブラリを利用した [*3]．実装においては，グリッド空間の定義および緯度経度とグリッドとの対応付けに，ジオハッシュ [*4] を利用した．ジオハッシュは 32 種類の文字を使って最大 12 文字の長さでグリッドを表現したハッシュである．ハッシュ長を変えることでグリッドの大きさを調節することができ，ハッシュは緯度経度から高速に計算される．例えば，北緯東経 “30, 140” の緯度経度に対して，“30, 140” を含むグリッドを表す”xj7d9v2fsmq4“ のジオハッシュが得られる．このハッシュにおける前半の 6 文字を用いることで $1200 \times 600m$ のグリッド，7 文字で $150 \times 150m$ のグリッドを表現できる．

[*2] 著者によって公開されたコードおよびパラメータを利用した. (https://github.com/adbrebs/taxi).
[*3] http://chainer.org/
[*4] https://en.wikipedia.org/wiki/Geohash

現実のシナリオにおいては，問合せ移動軌跡の地理範囲は明らかにならない．すなわち，過去にデータの存在しないグリッドに問合せ移動軌跡が存在する場合，グリッド ID が定義されておらず問合せ移動軌跡の各測位点にグリッド ID を割り当てられない問題が生じる．そのため，グリッドベースの提案手法や SubSyn アルゴリズムでは，単純にそのようなデータを扱うことが難しい．本実験においては，問合せ移動軌跡が既知のもと予めグリッド ID を定義する非現実的なシナリオは仮定せず，移動軌跡履歴から定義されるグリッド ID のみを利用して目的地を予測する．そこで，問合せ移動軌跡の測位点と対応する定義済みのグリッド ID が得られない場合は，その測位点に最も近い定義済みのグリッド ID を割り当てる．この処理により，前述の非現実的なシナリオと比べても遜色ない結果が得られることを，予備実験において確認した．なお，もう一つの既存手法である MLP は，連続値を扱うためこの処理を必要としない．

5.2　データセット

Taxi Service Trajectory (TST). TST データセットはポルトにおける数百台のタクシーの移動軌跡とタクシー ID などのメタデータを含む [19]．本実験においては，汎用的な目的地予測に焦点を当てているため，移動軌跡のデータのみを利用する．移動軌跡はタクシーの乗車地から降車地までのおよそ 15 秒間隔で測位された緯度経度の系列を含んでいる．提案手法と SubSyn アルゴリズムに対してはグリッド空間の範囲とグリッドの大きさを定義する必要がある．本実験においては，実験負荷を減らし様々な条件で実験するために，全データセットの一部を利用した．特に，表 3 に示す対象の地理範囲のデータを全て利用し，およそ $4000m \times 4000m$ の範囲において $150m \times 150m$ のグリッドを利用した．

GeoLife(GL). 個人のデータを対象に評価するために，北京における個人の移動軌跡データ [34], [35], [36] を利用した．データには約 2 年分 69 ユーザの移動手段がアノテーションされた軌跡が含まれている．本実験では，移動手段の変化点が滞在地であると仮定し，同一移動手段がアノテーションされた区間を一つの移動軌跡として扱う．移動軌跡は基本的に数秒単位で測位されており，緯度経度および時刻情報を含んでいる．過剰な情報を減らすため，移動軌跡の測位点を 60 秒おきにサブサンプリングした．また，各ユーザにおいて，移動軌跡の数が 10 より少ないユーザ

を除外し，結果として 46 ユーザの軌跡が得られた．移動軌跡は表 3 に示す北京市街のものを利用し，グリッドは TST と同様に $150m \times 150m$ の大きさで利用した．

5.3 実験設定

評価指標. 目的地予測の評価には二つの指標を使う．一つは $Accuracy@k$ であり，各問合せ移動軌跡 T_q に対して正しく推定された目的地の割合を表す．もう一つは $Distance@k$ であり，各 T_q に対して推定された目的地と真の目的地との距離を表す．この際，目的地訪問確率 $\mathbf{P}$ に基づく上位 k 個の目的地候補を対象として，これらの評価値を計算する．具体的にはテストデータにおける全ての T_q の数を $\#all\_T_q$，T_q に対して上位 k 個に予測された目的地を $\hat{d}_{T_q} \in \hat{C}^k_{T_q}$，そして真の目的地を d_{T_q} とすると:

$$Accuracy@k = \frac{\sum_{T_q}\{[\hat{d}_{T_q} = d_{T_q}] | \forall \hat{d}_{T_q} \in \hat{C}^k_{T_q}\}}{\#all\_T_q},$$

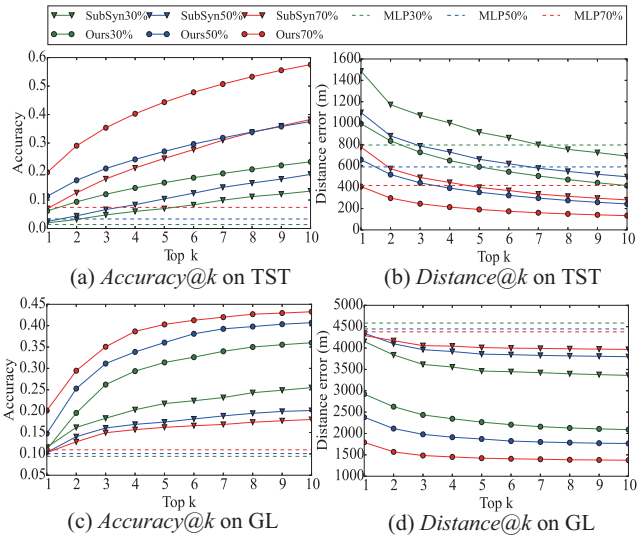
$$Distance@k = \frac{\sum_{T_q} \min_{\hat{d}_{T_q} \in \hat{C}^k_{T_q}} (\mathrm{Dist}(\hat{d}_{T_q}, d_{T_q}))}{\#all\_T_q},$$

と定義される．ここで $[\hat{d}_{T_q} = d_{T_q}]$ は真の目的地と予測目的地が同一のグリッドに存在するときに 1 をとり，そうでないときは 0 をとる．$\mathrm{Dist}(\cdot,\cdot)$ は二つの目的地間の距離を表し，具体的には提案手法と SubSyn アルゴリズムにおいては真の目的地と予測されたグリッドの中心との距離，MLP においては真の目的地と予測された目的地との距離を表す．$Accuracy@k$ は大きいほど，$Distance@k$ は小さいほど良い結果であることを意味する．

実験方法. TST データセットと GL データセットを用いて，前述の評価指標に基づいて提案手法の有効性を評価する．各データセットにおける移動軌跡をタイムスタンプの昇順にソートし，前の 70% を訓練データ，後の 30% をテストデータとする．訓練データを用いたモデルの学習については，データセットによって方法を変える．TST データセットにおいては，不特定多数のデータに対する手法の有効性を検証するために，訓練データの全ての移動軌跡に対して一つのモデルを構築する．一方で，GL データセットにおいては，個人を対象としているため，ユーザごとの訓練データに対して別のモデルを構築する．テストデータを用いた予測においては，各移動軌跡の最後の測位点に該当するグリッドを真の目的地とし，移動軌跡に含まれる測位点のうち出発地から α% を切り出した問合せ移動軌跡 T_q を入力とする．なお，GL データセットに対しては，各ユーザに対するテスト結果から得られる評価値の平均をとる．

5.4 結果と分析

まず，実験結果の分析を通して，次に列挙するリサーチクエスチョンを解決できているか否かを検証する．



(a) *Accuracy@k* on TST (b) *Distance@k* on TST

(c) *Accuracy@k* on GL (d) *Distance@k* on GL

図 6 各データセットにおける各手法による目的地予測の性能．手法名の後の数値は問合せ移動軌跡の入力長の割合 α を表す．

RQ1 提案手法は長期的な状態の依存性を考慮することで目的地予測の性能を向上させられるか?

図 6 に各データセットに対する結果を示す．まず，TST に対する結果からわかるのは，同じ長さの問合せ移動軌跡を入力とした場合，*Accuracy* と *Distance* の両方において，提案手法 (Ours) が SubSyn を上回っていることである．特に問合せ移動軌跡の入力長の割合が大きいほど，提案手法による性能の向上が顕著なことがわかる．さらには，Ous50% はより長い入力を利用した SubSyn70% をも上回っている．

MLP は全ての訓練データセットにおいて，真の目的地と予測目的地との距離の誤差が最小になるようにモデルを構築している一方で，提案手法はグリッド単位で目的地を正確に予測するようにモデルを構築している．そのため実験結果においてもそのような傾向が見られる．例えば，*Accuracy*@1 においては提案手法が顕著に MLP を上回っている．それと反対に *Distance*@1 においては，データセット全体における距離を小さくする能力を持った MLP が提案手法を上回っている．しかし，入力長 $\alpha = 50\%$ のとき提案手法は MLP に匹敵し，$\alpha = 70\%$ のときは僅かに上回っている．さらに MLP は一つの予測結果しか得られないのに対して，提案手法や SubSyn は確率値としてランキングされた結果を得ることができ，グラフにおいて k の値を大きくすると精度が向上していることからも，その有効性を確認できる．

以上の結果は提案手法が長期的な状態の依存性を考慮できていることを示唆している．

表 4 は問合せ移動軌跡長を変えながら (不) 正確に予測された目的地の数を各手法に対して算出したクロス表である．提案手法のみで正確に予測できた目的地の数と，従来

表 4　正しく (T) または間違って (F) 予測された目的地の数を表すクロス集計表．TST データセットおよび GL データセットにおいて異なる入力長の割合 α の問合せ移動軌跡を用いた，各手法による結果を表している．提案手法と既存手法の全ての結果において統計的な有意差が見られた (マクネマー検定, $p < .001$)．

TST

$\alpha = 30\%$	SubSynT	SubSynF	MLPT	MLPF
OursT	236	2,627	210	2,653
OursF	681	42,841	436	43,086
$\alpha = 50\%$	SubSynT	SubSynF	MLPT	MLPF
OursT	631	4,638	401	4,868
OursF	522	40,594	1,147	39,969
$\alpha = 70\%$	SubSynT	SubSynF	MLPT	MLPF
OursT	2,174	6,981	1,459	7,696
OursF	1,070	36,160	1,977	35,253

GL

$\alpha = 30\%$	SubSynT	SubSynF	MLPT	MLPF
OursT	126	212	81	257
OursF	97	2,103	121	2,079
$\alpha = 50\%$	SubSynT	SubSynF	MLPT	MLPF
OursT	121	295	91	325
OursF	78	2,044	115	2,007
$\alpha = 70\%$	SubSynT	SubSynF	MLPT	MLPF
OursT	131	344	97	378
OursF	65	1,998	110	1,953

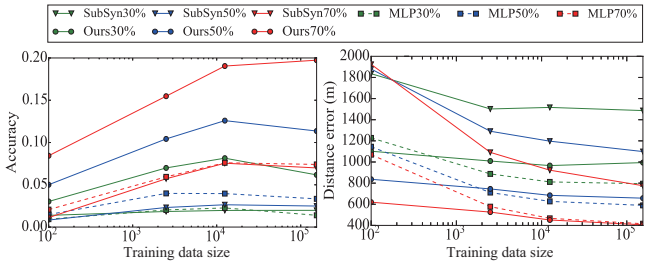


図 7　TST データセットにおける訓練データ量ごとの性能．

手法のみの数とでは統計的に有意なことを確認できる．これは入力長の割合 α が大きいほど顕著になる．

GL データセットにおいては，提案手法が従来手法と比べ α が大きいほどより良好な結果を得られていることがわかる．一つの理由として，提案手法は長い系列情報を上手く利用することで，個人のルーティンを上手く捉えられていることが想定される．一方で，SubSyn においては α が大きくても性能が向上していない結果が見られる．GL データセットは個人の移動軌跡であるため，自宅のような出発地が頻発し，α が大きいほど現在地は出発地から遠くなる．つまり，もともとデータ量の少ない GL データセットにおいては，α が大きいほどよりデータが過疎になることが原因で，問合せ軌跡の出発地と現在地からの遷移情報のみを利用した SubSyn アルゴリズムは上手く働かなかったと考えられる．

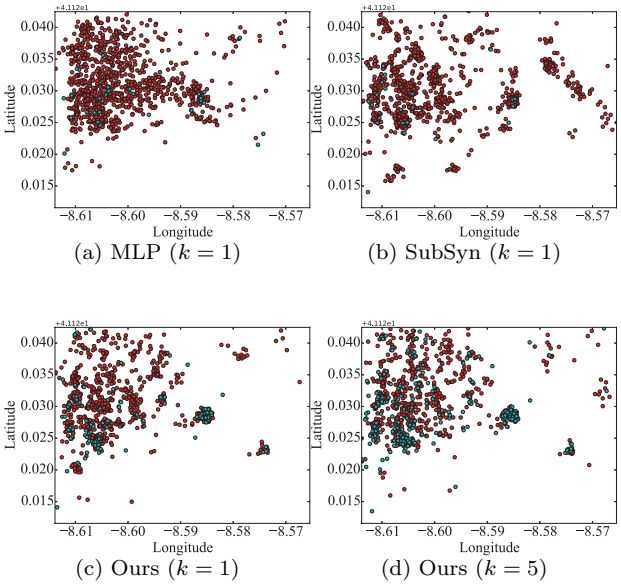
RQ2 提案手法はデータスパースネス問題を緩和できるか?

図 7 は TST データセットの訓練データ量を変えながら性能を比較している．長い系列をモデリングしているにも関わらず，提案手法は訓練データ量が少なくても，ほとんどのケースで従来手法を上回っていることがわかる．これは，マルコフ過程に基づいて遷移をカウントする方法とは異なり，RNN が可変長の問合せ移動軌跡の系列情報を中間層の連続的な特徴空間において上手く表現できているためである．加えて，表 3 と図 6 に示すように，GL データセットは TST データセットに比べデータ数が少ないものの，提案手法と既存手法との性能の差は TST よりも顕著である．このケースでは，MLP が最も低い性能を示しており，これは密度ベースのクラスタリングや sequence-to-point によるモデリングのアプローチが原因でデータスパースネスの影響を受けているためだと考えられる．SubSyn は MLP よりもわずかに良い性能を示しているが，前述の通りデータスパースネス問題に上手く対処している提案手法が最も高い性能を示している．

5.4.1　可視化

定性的に結果を評価するため，TST データセットにおける予測結果を図 8 に可視化した．正確に予測された目的地は緑の点，そうでないものは赤の点で示す．グリッドで重なって予測された結果を見やすくするために，予測結果に一定のホワイトノイズを加えた．SubSyn と MLP の結果においては，正確に予測された目的地が不均一に分布しており，データがたくさん存在するところに集まっている傾向にある．一方で，提案手法においては，多くの目的地をより均一かつ正確に予測できていることがわかる．

さらに図 9(a) に示す通り，特定の問合せ移動軌跡に対する予測結果の例を示す．傾向としては，SubSyn は，短い遷移の組合せで目的地への遷移確率を予測するため，現在地から近く目的地になりやすい地点を予測しているのに対して，提案手法は長期的な遷移を見ることで正確に予測できているケースが見られる．例えば左の結果を比較すると，多くのタクシーが市街地付近に訪問しているためか，SubSyn は市街地の目的地を予測している．これに対して提案手法は，市街地内から出発して市街地外に出るタクシーは駅に向かいやすいという意味を考慮して予測しているように見られる．なお，図 9(b) は提案手法の各サンプリングの過程において，何ステップ目に特定のグリッドを経由したかを表している．移動軌跡の測位間隔が一定であれば，おおよその訪問時間を予測していると考えることが



(a) MLP ($k = 1$) (b) SubSyn ($k = 1$)

(c) Ours ($k = 1$) (d) Ours ($k = 5$)

図 8 TST データセットにおいて予測された目的地をプロットした結果．50% の長さの問合せ移動軌跡を用いている．緑の点は正しく予測された目的地，赤の点は間違って予測された目的地を表す．

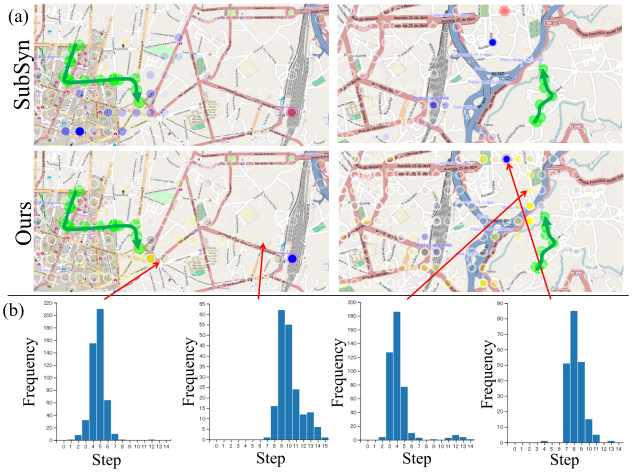
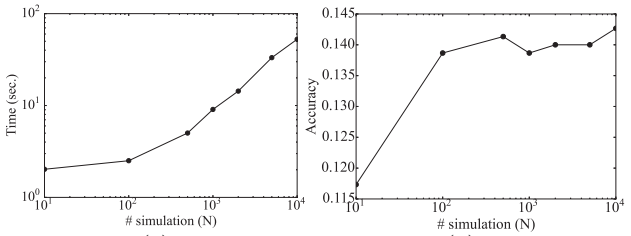


図 9 (a) SubSyn アルゴリズムと提案手法による目的地予測の可視化結果．緑が問合せ移動軌跡，赤が真の目的地，青が目的地予測結果，黄が移動経由地を表す．色が濃い地点ほど目的地である，または経由する可能性が高いこと表す．(b) サンプリングシミュレーションの各ステップにおいて，サンプルが矢印で示した地点に到達した回数．



(a) (b)

図 10 $\alpha = 50\%$ のときの異なる N に対する (a) 予測時間と (b) *Accuracy*@1 ．

できる．例えば (b) の左から 2 番目のグラフにおいては 9 ステップ目の頻度が最も高く，TST データセットは 15 秒間隔で測位されていることから，およそ $9 \times 15 = 135$ 秒後に矢印の地点に到達すると予測できる．

5.4.2 計算時間

図 10(a) は一つの問合せ移動軌跡に対して，予測に要する計算時間である．サンプリングシミュレーションの処理は CPU の 10 プロセスで並列化されている．グラフを見ると，オーバーヘッドコストが処理の多くを占めるシミュレーション回数 $N = 10$ の場合を除いて，N の増加に応じて計算時間が線形に増加しているのがわかる．特に N が数百であれば数秒で予測している結果を読み取れる．図 10(b) は異なる N による *Accuracy* を表す．ここでは実験効率化のため，TST データセットからランダムで選んだ 2500 の移動軌跡を用いて実験した．グラフから読み取れるように，N が大きいほど *Accuracy* が高い傾向にある一方で，$N = 100$ の時点で徐々に収束している．以上のことから，100 回程度のシミュレーション回数で十分な精度を得られ，実時間でも計算可能なことがわかる．

学習時間に関しては，TST データセットにおいては数日を要し，GL データセットにおいては一人のユーザにつき数十分を要した．学習時間は予測時間より長いものの，モデルはあらかじめ用意しておくことができるため，実アプリケーションサービスにおいては大きな問題にならないと思われる．

5.4.3 提案手法の妥当性

提案手法における個々のアプローチ (サンプリングシミュレーション，空間的近接性の考慮，およびゴールグリッドの利用) が妥当か否かを検証するために，他に考えられるアプローチとの比較実験を行った．

まず，サンプリングシミュレーションの代替として，RNN が出力する遷移確率のうち最も確率が高いものを目的地に到達するまで貪欲に選択するアプローチを実装し，TST データセットにおける 2500 の移動軌跡を用いて比較した．図 11 にこの方法 (Greedy) と提案手法 (Ours) による性能の比較を示す．Greedy は複数のルートを想定して目的地を予測しているわけではないため，Ours によるサンプリングシミュレーションの方が明らかに良好な結果を示している．また，Greedy は一つのパターンの遷移しか考えないため，目的地を確率として推定せず，ランキングされた複数の予測結果を得ることもできない．

図 12 はサンプリングの際に空間的近接性を考慮した遷移確率を用いた場合 (Ours) と用いなかった場合 (W/oSP) の性能を比較している．*Accuracy* は二つの手法でほぼ同じなのに対して，*Distance* は提案手法の方が良好な結果

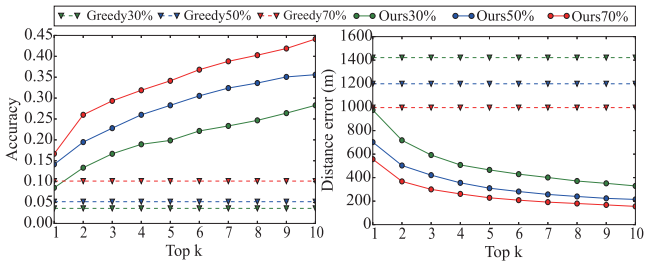


図 11 サンプリングシミュレーションを用いた提案アプローチ (Ours) と貪欲な探索によるアプローチ (Greedy) との性能の比較.

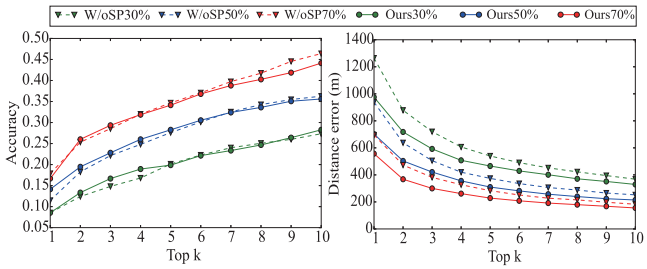


図 12 空間的近接性を考慮した場合 (Ours) としなかった場合 (W/oSP) との性能の比較.

を得られている．これは Ours がシミュレーションの際にサンプルが物理的に起こり得ないほど遠くへジャンプするのを防ぐことで，大きく距離を外して予測を間違えないような効果を生んでいるためである．

図 13 はゴールグリッドを使った場合と使わなかった場合 (W/oCG) の性能を比較している．Ours はほとんどの場合で W/oGG を上回っており，特に問合せ移動軌跡が短いほど，二つの手法による結果の差が大きくなっている．これは W/oGG の場合，真の目的地までの経由地に別の目的地が存在するとシミュレーションが止まってしまい，問合せ移動軌跡が短いほど（現在地が真の目的地から遠いほど），サンプリングで真の目的地に到達しにくくなってしまうためである．そのため，問合せ移動軌跡が長い 70% の場合のみ，わずかに *Distance*@1 において W/oGG が上回っている．

6. おわりに

本稿では，RNN を用いた移動軌跡からの目的地予測手法を提案した．移動軌跡をグリッド空間において one-hot 表現として扱い，系列的な遷移の情報を RNN モデルに学習させることで，(i) 長期的な系列の遷移を考慮し，(ii) データスパースネス問題を緩和可能な予測モデルを構築する．得られた RNN モデルをエンコーダデコーダの枠組みで利用し，ステップごとの遷移確率に基づいてユーザの目的地までの移動をサンプリングシミュレーションすることで，効率的な目的地への訪問確率の推定を可能とした．二つのデータセットを用いた評価実験を通して，提案手法は不特定多数や個人の移動軌跡データにおいて，従来手法よりも

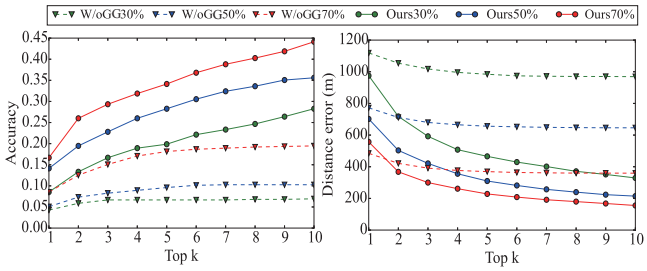


図 13 ゴールグリッドを利用した場合 (Ours) としなかった場合 (W/oGG) との性能の比較.

正確かつ少ない誤差で目的地を予測できることを示した．

制約と今後の課題. RNN モデルの学習においては，グリッド数 $|G|$ の増加に伴い計算時間が増加するため，今回の実験で対象とした地理範囲より広範囲かつ細かいグリッドを利用した場合，学習に多くの時間を要してしまう．本稿の実験においては，150×150 より細かいグリッドを利用していないにもかかわらず，提案手法は従来手法と比べ良好な結果を得ることができた．この問題を解決する一つのシンプルな方法としては，対象となる広い地域を分割して，分割された地域ごとに RNN モデルを構築することが考えられる．今後は細かいグリッドで広い地域をより効率的に扱える手法を検討したい．

もう一つの課題として，モデルに時間情報を組み込むことが考えられる．時間情報はユーザの行動を予測するのに有用な情報である (例えば，朝はオフィス，夜は居酒屋に向かうなど)．今後は異なる観点の複数の情報を上手く扱える Multi-view モデル [9] のような手法を応用することで，この課題に取り組みたいと考えている．

参考文献

[1] J. A. Alvarez-Garcia, J. A. Ortega, L. Gonzalez-Abril, and F. Velasco. Trip destination prediction based on past gps log using a hidden markov model. *Expert Syst. Appl.*, 37(12):8166–8171, Dec. 2010.

[2] D. Ashbrook and T. Starner. Using gps to learn significant locations and predict movement across multiple users. *Personal Ubiquitous Comput.*, 7(5):275–286, Oct. 2003.

[3] L. Chen, M. Lv, and G. Chen. A system for destination and future route prediction based on trajectory mining. *Pervasive Mob. Comput.*, 6(6):657–676, Dec. 2010.

[4] X. Chen and C. Lawrence Zitnick. Mind's eye: A recurrent visual representation for image caption generation. In *CVPR*, June 2015.

[5] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259, 2014.

[6] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, 1724–1734, 2014.

[7] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio. Em-

pirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[8] A. de Brébisson, É. Simon, A. Auvolat, P. Vincent, and Y. Bengio. Artificial neural networks applied to taxi destination prediction. *CoRR*, abs/1508.00021, 2015.

[9] A. M. Elkahky, Y. Song, and X. He. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *WWW*, 278–288, 2015.

[10] F. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: continual prediction with lstm. In *ICANN*(2), 850–855, 1999.

[11] A. Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

[12] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580, 2012.

[13] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997.

[14] E. Horvitz and J. Krumm. Some help on the way: Opportunistic routing under uncertainty. In *UbiComp*, 371–380, 2012.

[15] J. Krumm and E. Horvitz. Predestination: Inferring destinations from partial trajectories. In *UbiComp*, 243–260, 2006.

[16] J. Krumm and E. Horvitz. Predestination: Where do you want to go today? *Computer*, 40(4):105–107, April 2007.

[17] N. Marmasse and C. Schmandt. A user-centered location model. *Personal Ubiquitous Comput.*, 6(5-6):318–321, Jan. 2002.

[18] A. Mnih and Y. W. Teh. A fast and simple algorithm for training neural probabilistic language models. In *ICML*, 1751–1758, 2012.

[19] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas. Predicting taxi-passenger demand using streaming data. *Intelligent Transportation Systems, IEEE Transactions on*, 14(3):1393–1402, Sept 2013.

[20] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *ICML*, 1310–1318, 2013.

[21] D. J. Patterson, L. Liao, D. Fox, and H. Kautz. Inferring high-level behavior from low-level sensors. 73–89, 2003.

[22] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Neurocomputing: Foundations of research. chapter Learning Representations by Back-propagating Errors, 696–699. 1988.

[23] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning Internal Representations by Error Propagation, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, 318–362. 1986.

[24] A. Sordoni, Y. Bengio, H. Vahabi, C. Lioma, J. Grue Simonsen, and J.-Y. Nie. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *CIKM*, 553–562, 2015.

[25] I. Sutskever, J. Martens, and G. Hinton. Generating Text with Recurrent Neural Networks. In *ICML*, 1017–1024, 2011.

[26] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 3104–3112, 2014.

[27] D. Tiesyte and C. S. Jensen. Similarity-based prediction of travel times for vehicles traveling on known routes. In *GIS*, 14:1–14:10, 2008.

[28] L. Wei, Y. Zheng, and W. Peng. Constructing popular routes from uncertain trajectories. In *KDD*, 195–203, 2012.

[29] A. Y. Xue, J. Qi, X. Xie, R. Zhang, J. Huang, and Y. Li. Solving the data sparsity problem in destination prediction. *The VLDB Journal*, 24(2):219–243, Apr. 2015.

[30] A. Y. Xue, R. Zhang, Y. Zheng, X. Xie, J. Huang, and Z. Xu. Destination prediction by sub-trajectory synthesis and privacy protection against such prediction. In *ICDE*, 254–265, 2013.

[31] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu. Recurrent Neural Networks for Language Understanding. In *Interspeech*, 2013.

[32] W. Zaremba, I. Sutskever, and O. Vinyals. Recurrent neural network regularization. *CoRR*, abs/1409.2329, 2014.

[33] M. D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[34] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W. Ma. Understanding mobility based on GPS data. In *UbiComp 2008*, 312–321, 2008.

[35] Y. Zheng, X. Xie, and W. Ma. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010.

[36] Y. Zheng, L. Zhang, X. Xie, and W. Ma. Mining interesting locations and travel sequences from GPS trajectories. In *WWW*, 791–800, 2009.

[37] B. D. Ziebart, A. L. Maas, A. K. Dey, and J. A. Bagnell. Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior. In *UbiComp*, 322–331, 2008.

[38] D. Zipser. Advances in neural information processing systems 2. chapter Subgrouping Reduces Complexity and Speeds Up Learning in Recurrent Networks, 638–641. Morgan Kaufmann Publishers Inc., 1990.