%%

%% Copyright 2007-2020 Elsevier Ltd

%%

%% This file is part of the 'Elsarticle Bundle'.

%% ---------------------------------------------

%%

%% It may be distributed under the conditions of the LaTeX Project Public

%% License, either version 1.2 of this license or (at your option) any

%% later version. The latest version of this license is in

%% http://www.latex-project.org/lppl.txt

%% and version 1.2 or later is part of all distributions of LaTeX

%% version 1999/12/01 or later.

%%

%% The list of all files belonging to the 'Elsarticle Bundle' is

%% given in the file `manifest.txt'.

%%

%% Template article for Elsevier's document class `elsarticle'

%% with numbered style bibliographic references

%% SP 2008/03/01

%%

%%

%%

%% $Id: elsarticle-template-num.tex 190 2020-11-23 11:12:32Z rishi $

%%

%%

%%\documentclass[preprint,12pt]{elsarticle}

%% Use the option review to obtain double line spacing

%% \documentclass[authoryear,preprint,review,12pt]{elsarticle}

%% Use the options 1p,twocolumn; 3p; 3p,twocolumn; 5p; or 5p,twocolumn

%% for a journal layout:

%% \documentclass[final,1p,times]{elsarticle}

%% \documentclass[final,1p,times,twocolumn]{elsarticle}

\documentclass[final,3p,times]{elsarticle}

%% \documentclass[final,3p,times,twocolumn]{elsarticle}

%% \documentclass[final,5p,times]{elsarticle}

%% \documentclass[final,5p,times,twocolumn]{elsarticle}

%% For including figures, graphicx.sty has been loaded in

%% elsarticle.cls. If you prefer to use the old commands

%% please give \usepackage{epsfig}

%% The amssymb package provides various useful mathematical symbols

\usepackage{amssymb}

%% The amsthm package provides extended theorem environments

%% \usepackage{amsthm}

%% The lineno packages adds line numbers. Start line numbering with

%% \begin{linenumbers}, end it with \end{linenumbers}. Or switch it on

%% for the whole article with \linenumbers.

%% \usepackage{lineno}

\journal{Nuclear Physics B}

\begin{document}

\begin{frontmatter}

%% Title, authors and addresses

%% use the tnoteref command within \title for footnotes;

%% use the tnotetext command for theassociated footnote;

%% use the fnref command within \author or \address for footnotes;

%% use the fntext command for theassociated footnote;

%% use the corref command within \author for corresponding author footnotes;

%% use the cortext command for theassociated footnote;

%% use the ead command for the email address,

%% and the form \ead[url] for the home page:

%% \title{Title\tnoteref{label1}}

%% \tnotetext[label1]{}

%% \author{Name\corref{cor1}\fnref{label2}}

%% \ead{email address}

%% \ead[url]{home page}

%% \fntext[label2]{}

%% \cortext[cor1]{}

%% \affiliation{organization={},

%% addressline={},

%% city={},

%% postcode={},

%% state={},

%% country={}}

%% \fntext[label3]{}

\title{Title of Your Manuscript}

%% use optional labels to link authors explicitly to addresses:

%% \author[label1,label2]{}

%% \affiliation[label1]{organization={},

%% addressline={},

%% city={},

%% postcode={},

%% state={},

%% country={}}

%%

%% \affiliation[label2]{organization={},

%% addressline={},

%% city={},

%% postcode={},

%% state={},

%% country={}}

\author[inst1]{Author One}

\affiliation[inst1]{organization={Department One},%Department and Organization

addressline={Address One},

city={City One},

postcode={00000},

state={State One},

country={Country One}}

\author[inst2]{Author Two}

\author[inst1,inst2]{Author Three}

\affiliation[inst2]{organization={Department Two},%Department and Organization

addressline={Address Two},

city={City Two},

postcode={22222},

state={State Two},

country={Country Two}}

\begin{abstract}

%% Text of abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

\end{abstract}

\begin{keyword}

%% keywords here, in the form: keyword \sep keyword

keyword one \sep keyword two

%% PACS codes here, in the form: \PACS code \sep code

\PACS 0000 \sep 1111

%% MSC codes here, in the form: \MSC code \sep code

%% or \MSC[2008] code \sep code (2000 is the default)

\MSC 0000 \sep 1111

\end{keyword}

\end{frontmatter}

%% \linenumbers

%% main text

\section{Introduction}

\parskip5pt

The transportation industry is facing a rapidly increasing demand for artificial intelligence and machine learning applications for congestion modelling. These applications are often used to study traffic flow, transportation network design, and other transportation-related problems. Additionally, simulation models such as MATSim is widely used in transportation research and planning, and it is particularly useful for analyzing the impacts of land use and transportation policies on travel behavior and for evaluating the performance of different transport systems. They enable to simulate various scenarios, such as changes in transportation infrastructure, changes in land use, and changes in transportation policies, and it can also be used to evaluate the effectiveness of different transport planning and management strategies.

Data-driven methods and simulations relies on accurate and detailed input data to create a realistic representation of the population and their travel behavior. One of the major challenges of these applications is the availability of high-quality, diverse and representative data. Particularly, the agent-based simulation models require a large amount of micro-level population data to create the agents in the simulation environment. However, real-world data is often limited and expensive to collect, making it difficult to obtain the required amount of data for training and testing these models. Published datasets are generally given aggregated and anonymized due to privacy reasons. For example, the conditional distribution of variables (e.g., age distribution given sex) for the entire population of the census are deemed confidential and not published. If the input data is incomplete, inaccurate, or not representative of the population, the simulation may not produce realistic results. Also, if the input data is not recent or not updated, it might not reflect the current travel behavior of the population, which can lead to unrealistic simulation results. In this context, synthetic data plays a crucial role~\cite{badu2022composite,horl2021synthetic,farooq2013simulation,lederrey2022datgan}.

Despite the limitations of available data, the Census and Household Travel Survey datasets are a valuable resource for researchers, policymakers, and government officials, as it provides a wealth of information on the population and housing characteristics of different countries, as well as on travel patterns and trends\cite{bindschaedler2017plausible,stopher1996methods,horl2021synthetic}. They are commonly used in various fields like transportation, urban and regional planning, economic development and demographic analysis to understand and model travel habits and preferences of the households. Besides, they can be used by businesses to identify market opportunities and make decisions about advertising, marketing and distribution. These datasets are conducted and published every 5 or 10 years in most countries. They provide rich content about the demographic information of households and geospatial information on the daily trip routines these households have. The Census dataset includes socioeconomic characteristics of the population such as age, sex, and industry, while household travel surveys provide information on the travel patterns of individuals and households. This survey collects data on the number and types of trips taken, the mode of transportation used, and the purpose of the trip. It also provides information on the characteristics of the individuals and households making the trips, including age, income, and employment status. Although Census data provides various demographic features of the households. On the other hand, household travel surveys are more travel oriented and provides more details in relation to the travel features of the households.

Our methodology offers a novel way to generate a synthetic population along with corresponding trip sequences. Our framework is capable of generating a relational database including both socioeconomic features and sequential activity locations. Current state-of-the-art models~\cite{badu2022composite,lederrey2022datgan,horl2021synthetic,sun2015bayesian,farooq2013simulation,xu2019modeling} mostly focus on only socioeconomic features, while we offer a broader scope by incorporating sequential trip data into the synthetic population generation problem.

In summary our contributions can be listed as:

\begin{itemize}

\item Using a Conditional GAN to learn the joint distribution of the socioeconomic variables in the tabular population data

\item Using recurrent neural networks to create synthetic sequences of locations

\item Proposing a novel merging method to integrate population and trip data

\item Offering new evaluation methods for aggregated and disaggregated results

\end{itemize}

\section{Literature Review}

One of the main advantages of data augmentation is that it allows for the creation of a more complete and representative dataset. The utilization of data augmentation techniques in machine learning allows for expansion of the training dataset, resulting in increased representativeness and diversity. Furthermore, it helps to mitigate the risk of underfitting when dataset is limited. This can be especially useful when the available input data is incomplete or not representative of the population. By artificially increasing the size of the dataset, data augmentation can help to improve the performance and generalization of machine learning models. some examples are:

\begin{enumerate}

\item Image data augmentation: This technique is commonly used in computer vision tasks such as object recognition. Examples of image data augmentation techniques include rotation, flipping, zooming, and cropping.

\item Audio data augmentation: This technique is used in speech recognition and natural language processing tasks. Examples of audio data augmentation techniques include adding noise, changing the tempo and pitch, and time shifting.

\item Text data augmentation: This technique is used in natural language processing tasks such as sentiment analysis and text classification. Examples of text data augmentation techniques include synonym replacement, random insertion, deletion, and substitution.

\item Tabular data augmentation: Used in tasks involving numerical data such as regression and classification. Examples: random sampling, adding noise, feature scaling, feature engineering.

%% \item Generative models (overall generation): Generative models such as GANs and VAEs can be used to generate new samples that are similar to the existing data.

\end{enumerate}

The proposed algorithms for synthetic data generation problem depends on the type of the data and the task. Image data differs from text data and each type requires a specific method of augmentation. Image data often requires computer vision techniques in a generative model to create new synthetic data, while text data can be augmented using language models or other text generation techniques. The selection of the appropriate augmentation technique is crucial for ensuring the accuracy and usefulness of the synthetic population and trajectory data. Generative adversarial network models are generally preferred for image generation. Recurrent neural network models (RNN), on the other hand, are designed to process sequential data, such as text

and suited for text generation due to their ability to process sequential data\cite{shorten2021text}. GANs (Generative Adversarial Networks) are particularly fit for generating images because they can easily learn the high-dimensional probability distributions of image data. This is achieved through the use of convolutional layers in the generator network, which are able to capture connections between objects and their positions within an image in the data. Additionally, GANs are able to generate high-resolution images, which is difficult to achieve with other generative models. %%performance of the generation algorithm depends on the data and format: here image vs text, overall view

GAN architectures are commonly designed to handle uniform datasets such as generation of images~\cite{karras2020analyzing} rather than tabular datasets. Generating population tabular data with machine learning models can be more challenging for several reasons. First, population data often contains a mix of continuous and categorical variables, which can be difficult for a model to handle. Second, tabular data may contain missing values or outliers, which can also be difficult for a model to handle. Third, tabular data often has a large number of features, which can make it difficult to train a model. Finally, it is difficult to generate a synthetic dataset that has the same statistical properties as the original dataset, which is an important requirement for many applications.

The process of augmenting synthetic population and trajectory data is different and requires different types of data based on the nature of the information being augmented.

Our problem involves two types of data: population data in a tabular format and trajectory data in a sequential format. Synthesizing a population for transportation systems is a two-step process. First, the household population and its socioeconomic features should be created, and afterward, their trip activities need to be combined ~\cite{horl2021synthetic,badu2022composite}. Population data synthesis differs from trajectory data synthesis due to the nature of the datasets. Population features are nominal and kept in a tabular format, whereas trajectories are sequential and not necessarily tabular. While Census data is commonly used to generate population data, it is not suitable for trajectory prediction and generation. In addition to socioeconomic features, household travel surveys include individual activity locations, which are given as sequences of spatial zones and referred to as trip sequences or trajectories.

\subsection{Synthetic Population Data}

There is a wide range of methods that has been proposed to produce synthetic population data; Iterative Proportional Fitting~\cite{horl2021synthetic}, Markov Chain Monte Carlo~\cite{farooq2013simulation}, Bayesian Network~\cite{sun2015bayesian}, and deep learning models~\cite{badu2022composite,lederrey2022datgan}.

Iterative Proportional Fitting (IPF) is a method of generating a synthetic population based on the replication of sample data. It calculates marginal weights for specified columns of the dataset and generates a synthetic population using these weights. IPF based methods are prone to overfitting, because the interactions between columns are not taken into account. Sometimes these weights are available ~\cite{horl2021synthetic}, but it is rarely the case where the weights of all column combinations in this form are available for a given dataset and if the dataset does not contain such information IPF becomes computationally expensive. For example creating a synthetic travel demand based on open data is introduced to support replicable transport simulations and provide a comprehensive data basis for agent-based transport simulations and testing population demand. It is also discussed the impact of implicit correlation structures on the synthetic travel demand and the effects of using population samples for downstream simulations on the results. However, the evaluation of the model becomes too specific for the Paris and Île-de-France region due to the use of too many different datasets~\cite{horl2021synthetic}.

Monte Carlo Markov Chain (MCMC) method has been used when direct sampling is difficult and joint distributions $P(x,y)$ are not available. It learns from marginal distributions to generate data, and captures characteristics between variables. The principle of the MCMC approach is to use the Gibbs Sampler and combinations of marginal distributions for up-sampling the empirical data~\cite{farooq2013simulation}. The Gibbs sampler's input is created using marginals and partial conditionals from the dataset. The performance of IPF and combinatorial optimization techniques is discussed because of their limitations such as capturing heterogeneity, dependence on data accuracy, and poor scalability. The authors propose a Markov Chain Monte Carlo (MCMC) simulation-based approach to overcome these limitations. The method uses partial views of the joint distribution of agent's attributes to simulate draws from the original distribution. The performance of the simulation-based approach was compared to IPF using Swiss census data and showed improved accuracy. The method was also used to generate a synthetic population for Brussels, Belgium where data availability was limited \cite{farooq2013simulation}.

Bayesian Network presented before as a new method for population synthesis in agent-based micro-simulation models. Bayesian network (BN) method uses Directed Acyclic Graph (DAG) and structure learning methods to find dependencies between variables. BN does not perform well with complex and large-scale datasets since it is time-consuming and computationally expensive. Additionally, prior knowledge is needed to describe the relation between variables and to build DAGs. The Bayesian network is a compact representation of the joint probability distribution of demographic/socioeconomic variables and allows for preservation of privacy and confidentiality. The method was tested on the Household Interview Travel Survey data for Singapore and showed good performance in characterizing the joint distribution while avoiding overfitting of data \cite{sun2015bayesian}.

Generative adversarial networks (GANs) are a deep learning architecture that has gained significant attention in recent years for its ability to generate synthetic data that closely resembles real-world data. GANs have been successfully applied in image synthesis domain particularly~\cite{karras2020analyzing,zhang2017stackgan,zhu2017unpaired,brock2018large,goodfellow2020generative}. Unlike image data, tabular data have explicit feature relationships, which need to be preserved in the synthetic data. Another challenge is the imbalance in the distribution of categorical variables, which can lead to the generation of synthetic samples that are not representative of the real data distribution.

However, Numerous algorithms have been proposed to address these challenges for generating synthetic population data.

Of particular interest to our study, conditional tabular GAN models~\cite{xu2019modeling,lederrey2022datgan} seem to address the limitations that existing algorithms face when generating tabular population data. Furthermore, Vanilla GANs are trained to generate data that is indistinguishable from real data, but do not have the ability to condition the generation process on specific constraints. In contrast, conditional models are designed to generate data that meets this criteria.

For example CTGAN~\cite{xu2019modeling} uses a conditional generator to address these challenges, outperforming other methods on most real datasets while existing statistical and deep neural network models fail. Morever, CTGAN using PacGAN framework to tackle mode collapse problems by modifying the discriminator to make decisions based on multiple samples from the same class, either real or generated. The goal is to penalize generators with mode collapse and favor those with less mode collapse during training, which is proven through a connection between packing and mode collapse~\cite{lin2018pacgan}. (MAYBE) The generator in CTGAN is trained to generate synthetic data that not only resembles real data in terms of its statistical properties but also satisfies the conditional relationships between columns. The two discriminators used in PacGAN are used to evaluate the quality of the generated synthetic data from different perspectives: one evaluates the overall distributional similarity with the real data, while the other evaluates the conditional relationships between columns. By combining the two discriminators, CTGAN is able to generate synthetic data that is both statistically similar to the real data and also satisfies the conditional relationships between columns, reducing the likelihood of mode collapse.

%% sophistcated gans SS

\section{Synthetic Sequential Data}

- SEQUANtial data can be anything: text, audio or trajectories.

- THERE are two variants of trajectories, short and long. short we refer ''location' generation

While GAN is a promising direction to produce synthetic population data, it has difficulties in directly generating sequences of discrete tokens, such as trajectories or texts. This is partly because taking the gradient of discrete values directly is not possible and using gradient eventually makes little sense~\cite{yu2017seqgan}. Secondly, loss functions used by GANs are not taken into account by partial sequences during generation and score output sequences. Sequence Generative Adversarial Nets with Policy Gradient (SEQGAN)~\cite{yu2017seqgan} adapts the reinforcement learning approach to overcome these problems where generator network is treated as a reinforcement learning agent. This allows the algorithm to generate sequential dataset by using generative networks.

By nature, trajectory generation problem is very similar to text generation problem, where predicting the next element in a sequence is the main task, therefore text generation algorithms can be implemented for trajectory generation purposes. For example, recurrent neural network (RNN) based text generation algorithms are used to create synthetic trajectory data ~\cite{berke2022generating}. Badu-Marfo and Farooq~\cite{badu2022composite} has developed a model named Composite Travel GAN, where the tabular population data is generated by Vanilla GAN and the trip data is generated by SEQGAN. To the best of our knowledge, this is the only study that combines the generation of synthetic population and trajectory problems. Nonetheless, Badu-Marfo and Farooq ~\cite{badu2022composite} does not provide a distinct matching algorithm that enables the merge of the two synthetic datasets, which is one of the contributions that our study makes.

\section{Methodology}

As previously mentioned, most studies focus on either population generation or trajectory generation. However, these two pieces of information are often needed together (e.g., an agent with demographic features and activity locations in an agent-based simulation model). This paper proposes a comprehensive generation framework that handles both tabular population and non-tabular trajectory data. Our framework consists of three main components: $(i)$ Conditional GAN model to generate synthetic population data, $(ii)$ RNN-based generation algorithm for trip sequence synthesis, $(iii)$ combinatorial optimization algorithm to merge the output of these two algorithms effectively. Please see Fig. 1 for an overview of the proposed framework. We have two sets of features to be trained. The first set represents population features, and the second represents trip features. First set consists of elements of the population training data $ P\_i = \{P\_{i1}, P\_{i2}, P\_{i3}, \dots\ P\_{im}\} $ where each $P\_i$ represents a unique individual and $m$ is the total number of features. The second set is defined as $ L\_{i} = \{L\_{i1}, L\_{i2}, L\_{i3}, \dots\ L\_{in} \} $ where $i$ represents an individual and $n$ represents the maximum number of activity locations. Note that the number of activity locations might differ form one individual to another; we consider the individuals that have visited at most $n$ locations. Feature set $L$ is used to train the trajectory generation component of our algorithm.

\begin{table}[!ht]

\centering

\begin{minipage}[t]{0.48\linewidth}\centering

\caption{Population Features}

\begin{tabular}{ c c }

\toprule

Features & Samples \\

\midrule

Age & 25 \\

Sex & Male \\

Industry & Construction \\

Origin & (-27.468, 152.021) \\

Destination &(-27.270, 153.493) \\

\bottomrule

\end{tabular}

\end{minipage}\hfill%

\begin{minipage}[t]{0.48\linewidth}\centering

\caption{Sequential Features}

\label{tab:The parameters 2 }

\begin{tabular}{ c c }

\toprule

Features & Samples \\

\midrule

Location 1 & Kenmore \\

Location 2 & St. Lucia \\

Location 3 & South Brisbane \\

Location 4 & Kenmore \\

\bottomrule

\end{tabular}

\end{minipage}

\end{table}

Although the feature set $P$ is used to generate population samples, it includes origin destination values in addition to the socioeconomic features. These two features can in fact be matched with the activity locations in the feature set $L$, see Table 2. These common features will be used later within the bipartite assignment to match the observations in the population and trajectory data, see Figure 1. Nevertheless, origin and destination values in the population feature set P are given in coordinates, while the activity locations in the feature set $L$ are represented as zone names or zone ids. This is because it is computationally easier to train continuous variables in the CTGAN model; the excessive number of discrete zone variables in the network increases the computational complexity and causes memory issues within the CTGAN framework. Therefore, we use centroid coordinates instead of zone ids to train the CTGAN model. We keep the locations as categorical for the feature set $L$, because sequential models such as RNN and SEQGAN need discrete variables for training process.

\begin{figure}[htb]

\centering

\includegraphics[width=8.5cm]{flowchrt.pdf}

\caption{Schematic representation of the framework}

\label{fig:galaxy}

\end{figure}

\subsection{Tabular data generation}

\subsubsection{Generative Adversarial Networks}

Generative Adversarial Networks (GANs) have two neural network components as generator and discriminator, where they play a minimax game against each other. The generator tries to minimize the loss. The discriminator takes input from real and generated data for classification and returns values between 0 and 1 representing the probability of the given input is fake. Therefore, the discriminator wants to maximize the loss.

\begin{equation}

\min\_{G}\max\_{D}\mathbb{E}\_{x\sim p\_{\text{data}}(x)}[\log{D(x)}] + \mathbb{E}\_{z\sim p\_{\text{z}}(z)}[1 - \log{D(G(z))}]\label{eq}

\end{equation}

The generator produces desired output by using empty latent space and the discriminator behaves like a detective and tries to understand if the output from the generator comes from the real sample or not.

Latent space is nothing but a feature vector drawn from a Gaussian distribution with a mean of zero and a standard deviation of one. Through training, the generator updates the weights of this vector and learns the real distributions of the sample dataset. Vanilla GAN architecture commonly fails to handle imbalanced and mixed dataset, however recent studies propose significant changes in the GAN architecture to handle tabular data and learn joint distributions~\cite{xu2019modeling,zhao2021ctab,lederrey2022datgan}.

\subsubsection{Conditional GAN}

Conditional Tabular GAN (CTGAN) architecture differs from other GAN models considering its ability to generate tabular data and learn joint distributions. We make use of CTGAN model to synthesize the population because it allows us to generate discrete values conditionally. Besides, it ensures that correleations are kept. CTGAN explores all possible combinations between categorical columns during training. The model consists of fully connected networks to capture all possible relations between columns~\cite{xu2019modeling} and learns probability distributions over all categories. It uses mode-specific normalization and Gaussian mixture to shift the continuous values to an arbitrary range. One hot encoding and Gumble softmax methods are used for discrete values. By using Gumble Softmax trick, the model is able to sample from a categorical variables. The generator creates samples using batch normalization and ReLU (rectified linear unit) activation functions after two fully connected networks. Finally, the Gumble Softmax for discrete values and $tanh$ function is applied to generate synthetic population rows.

Besides, it includes PacGAN framework ~\cite{lin2018pacgan} to prevent mode collapse, which is the common failure for vanilla GANs ~\cite{badu2022composite}. CTGAN performs well in synthetic population generation, however, it has difficulties generating realistic trip distributions because the model does not support hierarchical table format and sequential features. Moreover, one hot encoding method can cause memory issues due to high dimensionality of locations. The composite models are proposed to address this issue ~\cite{liu2016coupled,badu2022composite}.

\subsection{Trajectory generation}

Synthetic data generation for music, text, and trajectory can be formulated similarly considering the sequential nature of the underlying data. To create input sequences, raw input data is converted into vectors of real numbers. Each discrete value is represented with an unique number in these vectors. This process is called tokenization in natural language processing. The number of tokens is determined by the number of the unique sequence values, which is named vocabulary size in text generation algorithms. RNNs show good performance in predicting and generating conditional elements in a sequence ~\cite{berke2022generating,caccia2018language}. In this study, we use \emph{textgenrnn} library for producing trajectory sequences. The model is inspired by char-rnn~\cite{karpathy2015unreasonable} and takes trajectory tokens as the input layer. Afterwards, it generates equal size of sequence lists by padding. The sequences are given to the embedding layer as input and create weighted vectors for each token. The embedding layer is followed by two hidden LSTM layers. The attention layer is skip connected to the LSTM and embedding layers. These layers are followed lastly by a dense and output layer.

\subsection{Bipartite assignment}

We sample synthetic population and trajectory datasets separately after training CTGAN and RNN models. Because our purpose is to generate individuals with trajectory sequences, we produce the same number of samples for population and trajectory components. We then use the Hungarian algorithm~\cite{munkres1957algorithms} to merge the generated samples.

The Hungarian method takes a non-negative n by n matrix as input, where n represents the number of sampled synthetic individuals. All row and column values are assigned a cost value in the matrix. This cost is defined as the distance between the origin coordinates by CTGAN and RNN components. Since the generated trajectory sequences are discrete, we use the centroid coordinates of the given locations. The cost can be formulated as $d = \sqrt {\left( {x\_{ctgan} - x\_{rnn} } \right)^2 + \left( {y\_{ctgan} - y\_{rnn} } \right)^2 }$, where $x\_{ctgan}$ and $y\_{ctgan}$ are the coordinates of the origin in the synthetic population data, and $x\_{rnn}$ and $y\_{rnn}$ are the coordinates of the origin zone centroid in the synthetic trajectory data. With the distance values, we build a complete bipartite graph $G=(P, T)$ with $n$ population vertices $(P$) and $n$ trajectory vertices $(T)$ where the edges are distances between the vertices. Hungarian algorithm finds the optimal matching by minimizing total distance. We use linear assignment without replacement to keep generated trajectories as they are and ensure the trip distributions do not change in the merging process. The algorithm minimizes the total cost (i.e., total distance between the pairs) via the following three steps. Firstly it subtracts the smallest element in each row from all the other elements. Then the smallest entry is equal to 0. Secondly, it subtracts the smallest element in each column. This makes the smallest entry 0 in the column. The steps continue iteratively until the optimal number of zeroes is reached. The optimal number of zeroes is determined by the minimum number of lines required to draw through the row and columns that have the 0 entries. One of the drawbacks of the Hungarian assignment method is the high computation cost with $O(n3)$ time complexity.

\section{Experimental Results}

We test our algorithm using the Queensland Household Travel Survey that includes travel behavior data from randomly selected households. The population and trip activity routines of the households are published as separate tables. Each observation is assigned to a unique person identifier code. We therefore connect population and trip activity tables by using these identifiers.

The population features contain socioeconomic characteristics such as age, sex, and industry (see Table 1), while the trajectory data includes the sequences of activity locations. The trip locations are categorical and defined as Statistical Area Level 1 (SA1) built from whole Mesh Blocks by Australian Bureau of Statistics. We eliminated observations which contains missing column values and limited the data considering a 40 km radius around Brisbane CBD. We have 5356 unique individuals with their activity chains in our dataset. The maximum number of locations in the trajectory feature set is 4 as 84\% of our observations in training data consist of short trips. In order to build a consistent setup, we train the population and trajectory components with the same dataset.

\subsection{Parameter Configuration}

We trained CTGAN with 300 epochs with batch size is 10. The layer sizes for the generator and discriminator are (512,512,512). The GAN models are fragile and hard to configure, thus we tested the hyperparameter

with different values. We applied the same number of epochs and batch sizes to our RNN model for training. We picked the temperature value as 0.7. The temperature takes values between 0 and 1. It allows the

algorithm to generate more divergent results. If the temperature is set as 0, it will make the model deterministic and the algorithm will always generate the same outputs.

\subsection{Results}

We will examine the results in three sections. First, we focus on our results for population generation from the CTGAN algorithm. Second, we discuss the trip sequence distributions resulting from the RNN algorithm.

Last, we focus on the final dataset that results from the merging of the synthetic population and trajectory datasets and analyze the conditional distributions between population and trip specific features. We use a wide range of test metrics considering the most prevalent statistical tests in the transportation literature~\cite{lederrey2022datgan,badu2022composite}.

\subsubsection{Synthetic Population Data}

\begin{figure}[htbp]

\centerline{\includegraphics[width=7cm]{vanillaganindustries.png}}

\caption{Comparison of marginals for industry attribute}

\label{fig}

\end{figure}

One of the prevalent methods to evaluate synthetic population is comparing population attributes. As discussed in literature review, GAN models are the most successful algorithms to generate synthetic population to our knowledge. We picked Vanilla GAN model for comparison and industry as the feature since amongst other features it has maximum number of classes and therefore hardest to get correct distributions, especially for Vanilla models. As observed in Figure 2, CTGAN outperforms Vanilla GAN in parallel with literature results. Vanilla GAN model cannot learn imbalanced distributions in the tabular data and struggles to reproduce the distribution of categorical industry features.

\subsubsection{Synthetic Trip Sequences}

Comparing trip sequences one at a time only provides little insight into the trip sequence distributions and cannot be used to determine whether the realistic route patterns are formed. Therefore, we consider the trip length to measure the similarity between real and synthetic trip sequences~\cite{badu2022composite,berke2022generating}. We calculate the total distance between all activity locations for each trajectory in the real and synthetic dataset and compare the trip distance distributions that result from this analysis. We also compare the resulting distributions with the synthetic trajectory dataset from SEQGAN that been applied earlier by ~\cite{badu2022composite} for the same purpose.

Fig 3 clearly shows that RNN closely replicates the trip length distribution in the real dataset, while SEQGAN largely fails to capture the trend in the trip distance distribution. SEQGAN under predicts the distances less than 5km and over predicts the trip lengths greater than 10km, which has been observed in other studies~\cite{badu2022composite} as well. This comparison clearly shows that RNN outperforms SEQGAN and produces realistic trip distance distributions.

\begin{figure}[htbp]

\centerline{\includegraphics[width=8.5cm]{comprr.png}}

\caption{Comparison of trip length distributions}

\label{fig}

\end{figure}

\subsubsection{Synthetic Composite Data}

The marginal distribution of trip distances are solely not enough to show if the correlations are kept between population and trajectories. One needs to consider joint or conditional distributions between population and trip specific features to further examine this aspect. Figure 4 shows conditional trip distances on a given population feature, industry in this case. In particular, Figure 4a presents the distribution of trip distance for those individuals who are employed in the education sector. Note that this conditional distribution can be observed only after the two datasets are merged using the Hungarian algorithm considering the distance between the pairs. Therefore, these results are labeled as 'CTGAN+RNN' in Figure 4a. The comparison between real dataset and the synthetic dataset that results from our algorithm reveals again a very strong similarity. The framework that we propose captures the joint distribution between variables after matching the two datasets.

Figure 4b depicts the distribution of trip distances for all sectors on the coordinate plane to see if the results in 4a are biased only for certain industries. We have 19 industry variables and use 30 bins to obtain trip length intervals. Thus, figure 4b has 570 points in total. Each point on the plane refers to conditional trip distance percentages of the synthetic and real samples. The percentage values on the X and Y axes represent the frequency of the observations in a specific bin interval. Therefore, for each industry, the sum of 19 separate points is equal to 1. Fig. 4b also includes a proportionality line and a regression line to illustrate the overall agreement between synthetic and ground trip distributions.

\begin{figure}[!ht]

\centering

\begin{subfigure}[b]{0.4\textwidth}

\centering

\includegraphics[width=\textwidth]{condisteducinkm.png}\llap{

\parbox[b]{3in}{(A)\\\rule{0ex}{1.4in}

}}

\label{fig:1a}

\end{subfigure}

~ %add desired spacing between images, e. g. ~, \quad, \qquad, \hfill etc.

%(or a blank line to force the subfigure onto a new line)

\begin{subfigure}[b]{0.4\textwidth}

\centering

\includegraphics[width=\textwidth]{bonibonlines.png}\llap{

\parbox[b]{3in}{(B)\\\rule{0ex}{1.7in}

}}

\label{b}

\end{subfigure}

~

\caption{Comparison of trip length distributions for (a) Education (b) All industries}\label{fig:animals}

\end{figure}

The conditional and unconditional distributions we get from figure 3 and 4 are also tested with statistical metrics; Pearson Correlation, Mean-standardized root mean square error $ (SRMSE)$ and $ R2 $ Score (Coefficient of Determination). We created two frequency lists for the trip length distributions of Conditional GAN, Sequential GAN and RNN models. In Table 3, CTGAN simply indicates the results based on origin and destination values (in coordinates) that are generated as part of the population features. SEQGAN and RNN can be used solely to generate marginal trip distance distributions, while CTGAN+SEQGAN and CTGAN+RNN represents the results after merging of the corresponding population and trajectory datasets. We set the constant term as 0 when we fit a linear regression equation and calculate the $R2$ score. Overall, SEQGAN exhibits a poor performance on imitating real distribution because the generated conditional distributions follow Gaussian distribution rather than tracking the real distribution. Table III gives a summary of the synthetic trip sequence generation performance of three algorithms.

\begin{table}[htbp]

\def\arraystretch{1}%

\caption{Benchmarking algorithms on statistical tests}

\centering

\begin{adjustbox}{width=\columnwidth,center}

\begin{tabular}{|l|c|c|c|c|}

\hline

\textbf{Trip Distance} & \textbf{Algorithm} & \textbf{Pearson} & \textbf{R2 Score} & \textbf{SRMSE} \\

\hline

\hspace{0.1cm} & CTGAN & $0.495$ & $0.231$ & $0.0572$ \\\cline{2-5}

Unconditional & SEQGAN & $ -0.256$ & $ -0.558 $ & $0.116 $ \\\cline{2-5}

\hspace{0.1cm} & RNN & $0.967$ & $0.927$& $0.0053 $ \\

\hline

\hline

\hline

\hspace{0.1cm} & CTGAN & $0.302$ & $0.018$ & $0.0479$ \\\cline{2-5}

Conditional & CTGAN + SEQGAN & $ -0.161$ & $ -0.375 $ & $0.0789 $ \\\cline{2-5}

\hspace{0.1cm} & CTGAN + RNN & $0.867$ & $0.724$& $0.0134$ \\

\hline

\end{tabular}

\end{adjustbox}

\label{tab:template}

\end{table}

Fig 5 exhibits the spatial distribution of activity locations in both real and synthetic datasets. The most popular activity locations such as airport and city center are coloured with yellow, as most intense areas according to the real samples. As seen in Figure 5, the synthetic location distribution is similar to the ground truth. Even though some of the areas may exhibit a mismatch in terms of trip frequency, the percentage error does not deviate from the real amount significantly, Therefore we can conclude that our method is able to learn geospatial distributions contextually.

\begin{figure}[htbp]

\centering

\includegraphics[width=.45\textwidth]{brisbanerealsa2map.png}\hfill

\includegraphics[width=.45\textwidth]{brisbanesa2mapping.png}\hfill

\caption{Spatial distribution of real and synthetic datasets}

\label{fig:figure3}

\end{figure}

\section{Conclusion and Future Work}

Synthesis of a mobility database is challenging due to the complex nature of the data. We propose a hybrid framework for generating such complex dataset. We divided the synthesis problem into two sub-parts by generating population and sequential trajectories separately. We use up-to-date models such as CTGAN for tabular data generation and RNN for trajectory generation. We have also applied a combinatorial optimization algorithm to merge these two synthetic datasets. In our future work, we will extend our experiments and design a new framework considering other sources of data to help capture correlations between population and trajectories.

%% If you have bibdatabase file and want bibtex to generate the

%% bibitems, please use

%%

\bibliographystyle{elsarticle-num}

\bibliography{cas-refs}

%% else use the following coding to input the bibitems directly in the

%% TeX file.

% \begin{thebibliography}{00}

% %% \bibitem{label}

% %% Text of bibliographic item

% \bibitem{}

% \end{thebibliography}

\end{document}

\endinput

%%

%% End of file `elsarticle-template-num.tex'.