**Report for exercise Final from group F**


| | |
|---|---|
| Tasks addressed: | 6 |
| Authors: | Jianzhe Liu (03751196)<br>Hao Chen (03764817)<br>Yang Cheng (03765398)<br>Pemba Sherpa (03760783) |
| Last compiled: | 2023–07–18 |
| Source code: | https://github.com/Chuck00027/MLCMS-GroupF/tree/main/Z |


The work on tasks was divided in the following way:

| | | |
|---|---|---|
| Jianzhe Liu (03751196) | Task 1 | 25% |
| | Task 2 | 25% |
| | Task 3 | 25% |
| | Task 4 | 25% |
| | Task 5 | 25% |
| Hao Chen (03764817) | Task 1 | 25% |
| | Task 2 | 25% |
| | Task 3 | 25% |
| | Task 4 | 25% |
| | Task 5 | 25% |
| Yang Cheng (03765398) | Task 1 | 25% |
| | Task 2 | 25% |
| | Task 3 | 25% |
| | Task 4 | 25% |
| | Task 5 | 25% |
| Pemba Sherpa (03760783) | Task 1 | 25% |
| | Task 2 | 25% |
| | Task 3 | 25% |
| | Task 4 | 25% |
| | Task 5 | 25% |

**Report on task TASK 1, Introduction**

In this chapter, we provide an overview of pedestrian movement analysis, highlighting its significance in urban planning, transportation management, and crowd control. Pedestrian movement analysis (PMA) is crucial in many fields, including urban planning, transit management, and crowd control. Understanding how pedestrians navigate urban environments is critical for creating efficient pedestrian amenities that enhance safety, accessibility, and comfort. City Planners can utilize this knowledge to improve the mobility and minimize congestion while also optimising the layout of sidewalks, crosswalks, and other pedestrian infrastructures. For example, Munich sees a huge number of out of city tourists during October Fest. This not only puts pressure on the public venue and transportation but also is a huge overcrowding and safety hazard. We could use PMA to determine venue design and additional transportation services in places of expected congestion. We can also extend this analysis for transportation management. By combining vehicle traffic data and pedestrian movement analysis, engineers can optimize traffic signal timing and reduce conflicts between pedestrians and automobiles by accurately forecasting pedestrian behaviour.

**Challenges in PMA:** While there are impressive benefits of PMA, the challenge comes in creating an accurate model and performing intelligent analysis. For this report, we will look into three main challenges:

- Diversity in Pedestrian Behaviors: Pedestrians exhibit a wide range of behaviors which can be influenced by factors such as age, gender, individual preferences, etc.
- Spatial Complexity: Environments will hevily influence pedestrian behavior. This is more apparent in complex spatial configurations, like intersections, corridors, and bottlenecks.
- Crowd Dynamics: The collective behavior of a crowd can result in emergent phenomena such as crowd turbulence or lane formation, which impact pedestrian flows and will require specialized analysis techniques.

**Fundamental Diagram:** Let's start with a simple method where we will try to understand the relationship between pedestrian density and velocity [4]. The research aims to test the ability of microscopic models to replicate the velocity-density relationship. The study compared the velocity-density relationship observed in single-file movement with the movement in a plane according to Weidmann [7]. Surprisingly, the findings reveal an agreement between the velocity-density relationship observed in single-file movement and existing literature data for movement in a plane 1. The study identifies a linear relationship between the required length of one pedestrian and velocity, providing valuable insights into pedestrian dynamics. This simple model can serve as a benchmark for evaluating more complex pedestrian movement models. Below is the formula 1 used for Weidmann Model where $s\bar{K}$ is the mean spacing, $v$ is the velocity, $\lambda$ is a paramenter, $T$ is the time gap.

$$v = FD(s\bar{K}, v_0, T, \lambda) = v_0 \left(1 - e^{-\lambda \frac{s\bar{K}}{v_0 T}}\right) \tag{1}$$

**Experiments:** We can also use actual experimental data as a baseline for training and analysis. Furthermore, such experiments can be restricted to focus on a few key aspects such as understanding the dynamics of pedestrian streams under various boundary conditions. A study [8] was conducted to analyze pedestrian behavior in various scenarios: straight and rounded corridors, a T-junction, and a bottleneck.The experiments were conducted in halls of the fairground in Düsseldorf, Germany, in 2009, with up to 400 participants 2. An additional similar experiment involving a group of soldiers was also carried out in Düsseldorf in 2006. By utilizing the Voronoi method for high-resolution analysis, the researchers compare pedestrian movement characteristics in different scenarios, including open, periodic boundary conditions, and restrained outflow. These findings revealed distinct characteristics of pedestrian movement based on the specific boundary conditions. Notably, at higher densities, pedestrian movement patterns vary significantly across different scenarios, providing valuable insights into the influence of boundary conditions on pedestrian flow dynamics. These experiments and results provide valuable insights into the dynamics of pedestrian movement and a valuable starting point to futher our understanding of crowd behavior dynamics and pedestrian movement analysis.

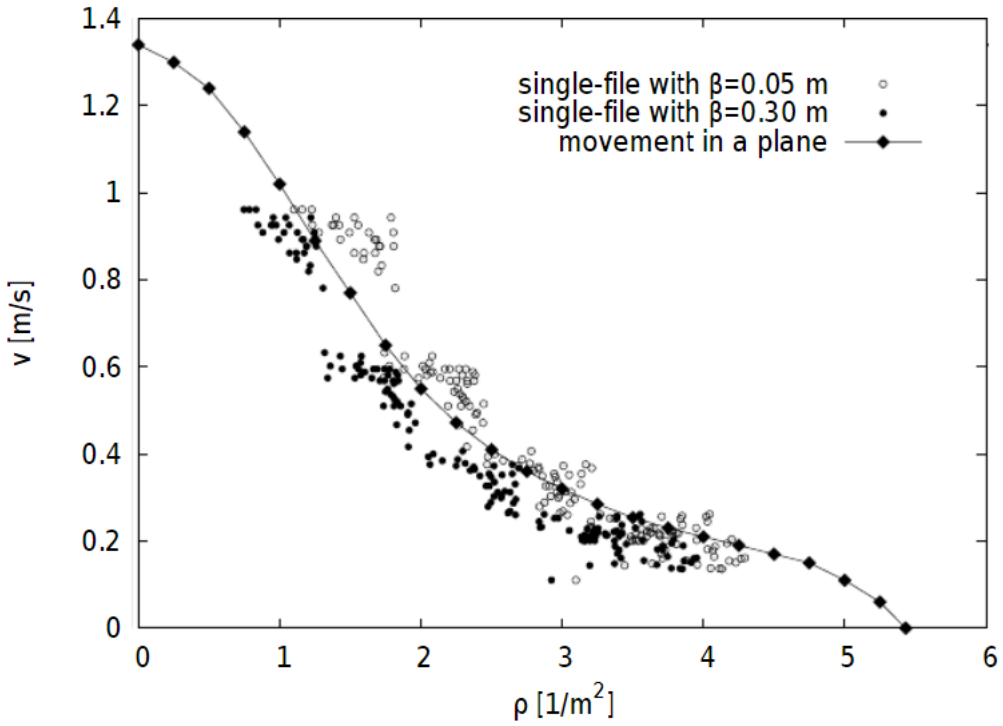



Figure 1: Comparison of the velocity-density relation for the single-file movement with the movement in a plane according to Weidmann. [4]

**ANN and Future Works:** Artificial neural networks (ANN) show promise in enhancing the accuracy of pedestrian movement predictions. Their ability to identify various patterns and utilize multiple parameters provides an alternative approach for forecasting pedestrian behaviors. Integrating ANN-based models in pedestrian movement analysis can improve the understanding of complex geometries and aid in designing efficient pedestrian facilities. Traditional models [7] with limited parameters face challenges in accurately predicting pedestrian movements in complex geometries. This research [5] presents initial steps in testing this approach by comparing estimations of pedestrian speed using a classical model and an ANN in corridor and bottleneck experiments. Below shows the formula 2 used for the ANN where $H$ is the hidden layers, $\bar{s}_K$ is the mean spacing and $K$ is the relative positions .

$$v = NN(H, \bar{s}_K, (x_i - x, y_i - y, 1 <= i <= K)) \tag{2}$$

In this study, a feed-forward neural network is used and is compared to a fundamental diagram-based model. They trained the neural network using data obtained from experiments, like [8], conducted in bottleneck and corridor scenarios with closed boundary conditions. The study also shows that when compared the performance of the neural network to an aggregated model based on fundamental diagrams, the speed prediction for mixed data is significant increased. Furthermore, the neural network also outperforms the aggregated model by achieving up to a 20 percent enhancement in prediction accuracy. Even though these are only initial steps, this performance and accuracy increase demonstrates the promise of using artificial neural networks for predicting pedestrian behavior in complex geometries.

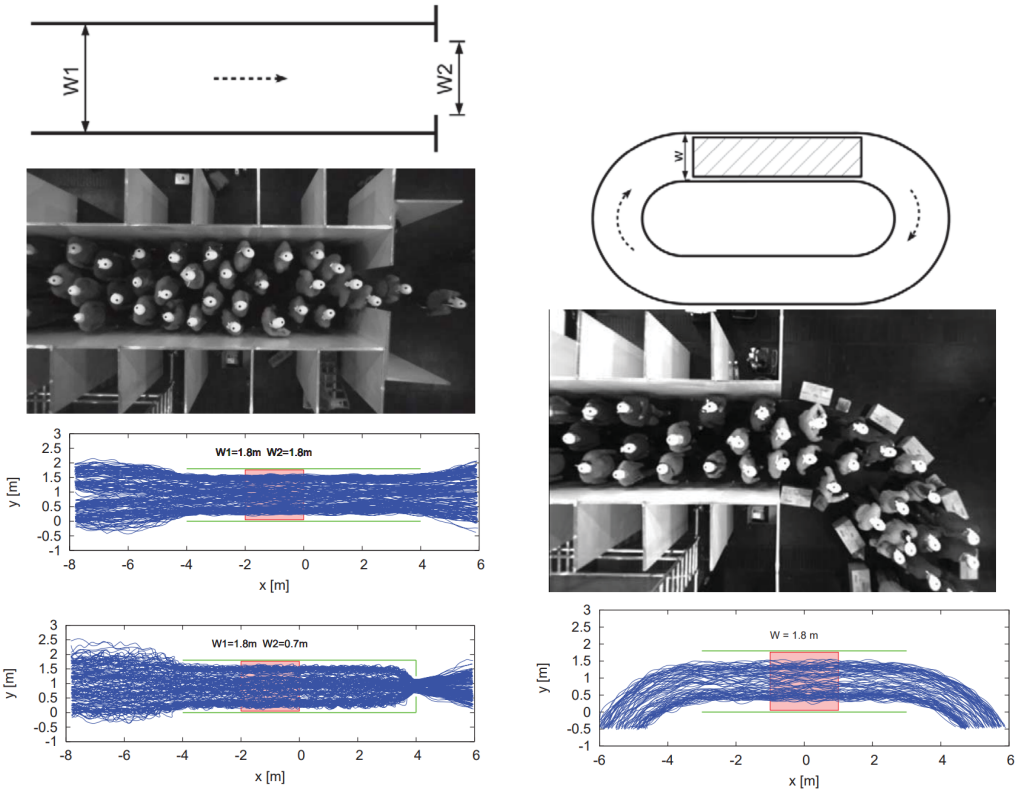



Figure 2: Sketch, snapshpot and pedestrian trajectories of the experiment: (Left) in straight corridors, (Right) rounded corridors with closed boundary condition. [8]

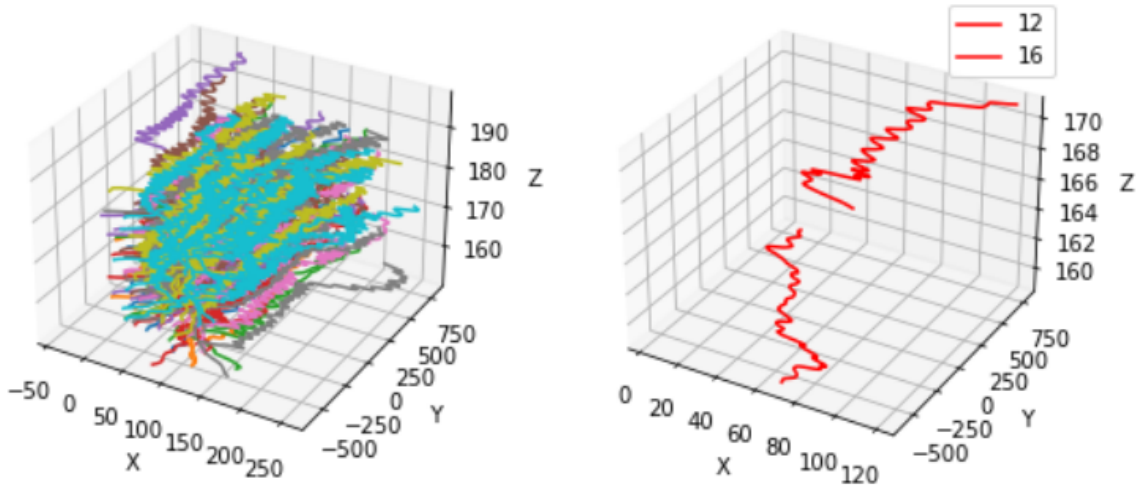
Starting here, we can explore other neural network models like Recurrent Neural Network (RNN) as was did in this study [1]. RNN models is great at sequence prediction tasks and we can create a Long Short-Term Memory (LSTM) model on top of that to predict pedestrian trajectory. They reimagine the problem of trajectory prediction as a sequence generation task, where the goal is to forecast the future trajectory of pedestrian based on their past positions. This paper explores the this idea and demonstrate that this LSTM model can learn underlying patterns in pedestrian movement and can be used to predict future pedestrian trajectories. This is not only a big step forward in understand pedestrian behavior but anticipating them as well. This concept can be vital in developing autonomous systems as well.

**Report on task TASK 2, Data processing and Weidmann model**

**Loading data**

As the request in the original paper, we load the dataset from 'https://doi.org/10.5281/zenodo.1054017'. There are two datasets named Corridor data and Bottleneck data, which store the pedestrians' trajectory in two different scenarios. There are five column in each dataset, in terms of ID FRAME X Y Z. ID refers to the name of the pedestrians. The frame rate is 1/16s .And the X Y Z refer to the position of pedestrians in 3D scenario.

In the first step, we load the dataset and visualize their trajectory . We could plot all the trajectory together to find out the characteristics of each scenario. And we could also select which pedestrians' trajectory to plot, so that we could zoom at the specific pedestrians.(As shown in FIgure 4 )




(a) 1 (b) 2

Figure 3: [1]All Trajectory in Bottleneck scenario [2]Two selected pedestrians' trajectory

Furthermore, we could plot the position of pedestrians at a specific frame such as Figure 4 . For better visualization, we ignore the Z axis (height). It's more intuitive to observe in a 2D figure than 3D figure. Moreover, the data processing method in the paper also transformed to 2D scenario.

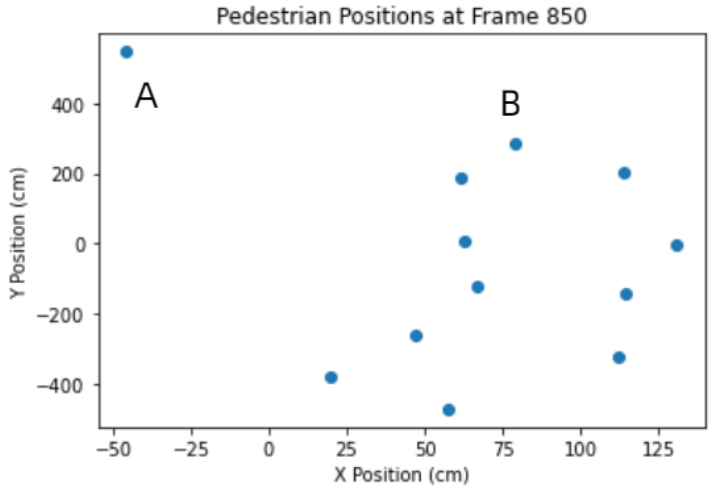



Figure 4: Pedestrian position at frame 850, from 'ug-180-030' dataset

**Data processing**

In this step we need to implement data processing as the request in the paper. The goal is that we transform the position information to mean spacing and corresponding speed of pedestrians. Further we generalize the dataset to fit the Weidmann model and ANN model. We will explain the methods in 3 different aspects.

**1. Data collection**

As the request in the paper , we need to measure the flow and density every 10 second for pseudo independence. In that way, we could ensure that the measurement at different frame will be approximately independent of each other. This assumption is made to simplify the analysis and calculations

**2.1 Data pruning**

As the request in the paper, we calculate the mean spacing based on the 10 nearest neighbors. In that way , we regard the 10 neighbors as a group. They are close to each other and therefore they might influence the behavior of each other. It is a straightforward method for data pruning. But it might be only reasonable for density scenario and we improve the data pruning method

We will explain with a specific scenario. As shown in the Figure 4, we could find out that pedestrian A is one of the 10 nearest neighbors of pedestrian B. But pedestrian A is far away from the pedestrian B. So that the pedestrian A should have little influence on pedestrian B, and pedestrian A should be regarded as noise and deleted from the 10 nearest neighbors. In other words, we should not only consider the nearest neighbors but also the distance.

So that we introduce the confidence interval theory to improve the data pruning method. A confidence interval is a range of values that provides an estimate of the true value of a population parameter. [3]Confidence Interval provides a range of plausible values for the population parameter based on the observed sample.(As shown in figure 5)we calculate a 95% confidence interval for the mean spacing, it means that we are 95% confident that the true mean spacing of the population lies within the calculated interval. Here we apply 95% confidence interval. Because if we could obtain 10 nearest neighbors, they usually stay close to each other. IN that way, we could delete the pedestrians, whose distance are outside of the 95% confidence interval. So that they are regarded as noise and deleted from the 10 nearest neighbors. We will use the remaining pedestrians to calculate the mean spacing.

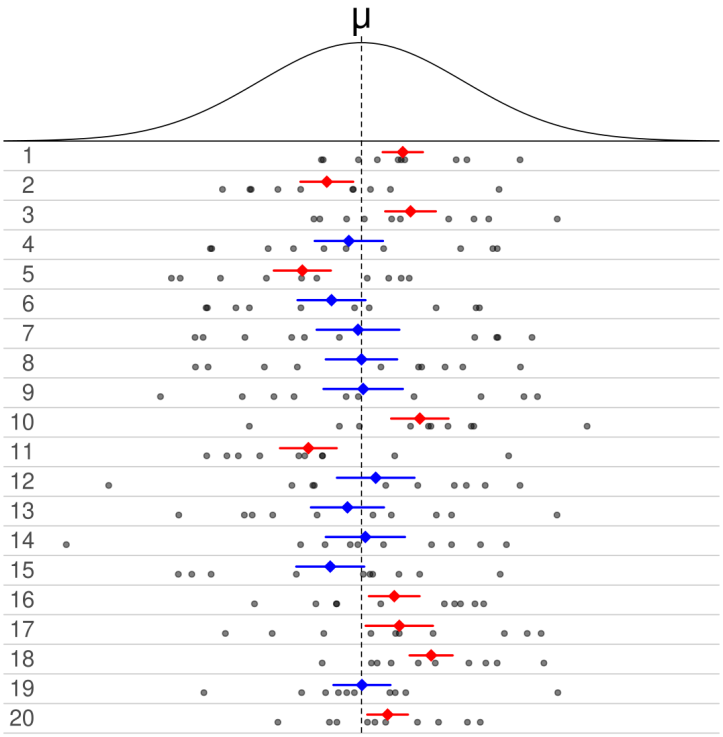



Figure 5: Confidence interval theory

**2.2 KDTree for 10 nearest neighbors**

Finding the 10 nearest neighbors could be hard.So that we apply KDTree, which enabling us to efficiently obtain the 10 nearest neighbors. [2]The KDTree is a multidimensional binary search tree that partitions the data space into regions for quick nearest neighbor searches.(As shown in Figure 6)

By constructing a KDTree from the pedestrian positions, we were able to query the tree to obtain the indices of the 10 nearest neighbors, including the pedestrian itself. This approach enabled us to calculate the mean spacing based on the distances to these neighbors.

Using the KDTree allowed us to optimize the search process and significantly improve the computational efficiency, especially when dealing with large datasets. It provided an effective solution for finding the nearest neighbors within the given dataset, facilitating the analysis of pedestrian dynamics and the study of their interactions.

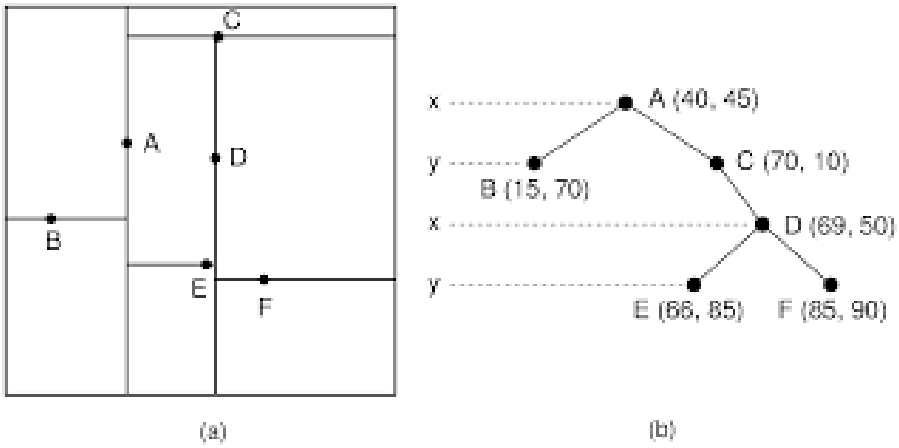
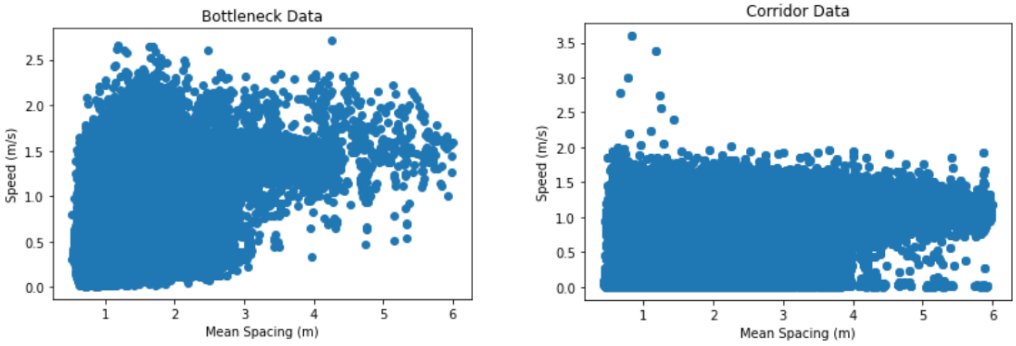



Figure 6: KDtree theory

**2.3 Claculation of speed** In order to calculate the speed, We just divide the position distance between two adjacent frames of the same pedestrian by the time. It is worth noting that each frame corresponds to 1/16 second. Until now, we could obtain the mean spacing and corresponding speed.

**3 show the combined result** In order to analyse the pedestrians' behavior,we should combine the result of those data set which belong to a same scenario.Although their parameter are a little bit different, but we show the samiliar behaviour. So we should combine them together to analyse.The combined resluts are shown in Figure 7

**Weidmann model** The weidmann model is a classical models to predict the speed based on the mean spacing. [6] The equation is defined as : $v = FD(\overline{s}_k, v_0, T, l) = v_0 \left(1 - e^{\frac{l-\overline{s}_k}{v_0 T}}\right)$




(a) 1 (b) 2

Figure 7: [1]All Trajectory in Bottleneck scenario [2]Two selected pedestrians' trajectory

The corresponding parameter time gap T, the pedestrian size l and the desired speed $v_0$ parameters are shown in the Table. We base on these parameter and fit to curves, the result is shown is Figure 8

| Experiment | R | B |
|---|---|---|
| l(m) | 0.64 | 0.61 |
| T(s) | 0.86 | 0.48 |
| $v_0$(m/s) | 1.60 | 1.58 |

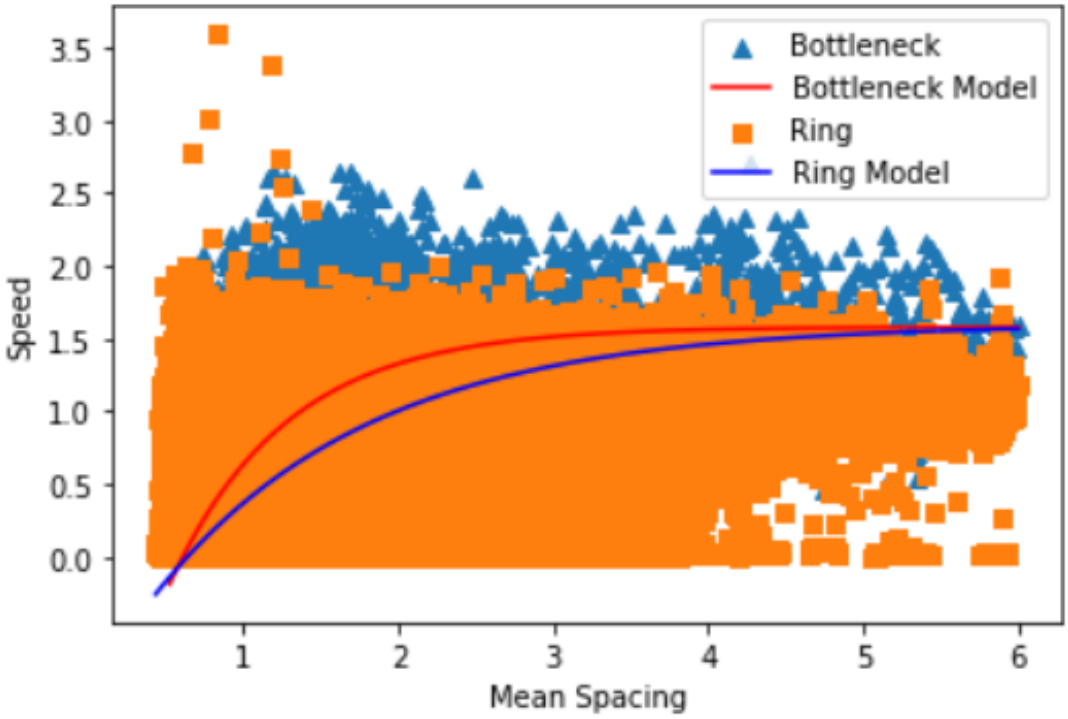



Figure 8: Curve fitting with weidmann model

The fitting curves are quite samiliar as the curves in the report.The two data sets represent distinct interaction behaviors, as shown in Figure 8.In congested conditions, the speed at a particular mean spacing is generally higher in the bottleneck than on the ring.As a result, the estimation of the time gap varies significantly depending on the geometry.

**Report on task TASK 3, Neural Network Training**

As shown in section 2, after having obtained the mean-spacing parameter, we can compute the result of weidmann model on different combinations of the two datasets. In order to make the comparison, our next step is to train a NN model to approach the velocity in a different way. In the original Paper,it describes a pretty simple network, which has a feed-forward mechanism, several hidden layers, a *ReLu* activation function and a one-node linear output layer. Here is to mention that during the training,we have to use *bootstrap* over 50 subsamples and *K-fold-cross validation* as our method to validate.

Basically, our implementing process can be described as followings:

- first, as requested, we load and split the data into *training* and *testing* parts, following a 50/50 percentage.
- then we take sk and other 20 x,y coordinates as input, measured v as target(they are given after the data processing part).
- then we start the train.By using the bootstrap, we train 50 samples each batch, and train them for 100 epochs. Here we use lr=0.01 and Adam optimizer. At the end of each 10 epoch it will print the train loss and by the end of the whole training we also plot both the train loss and test loss in one figure.
- besides, by validation, since we choose to use K-fold-cross validation, and we have 50 samples each batch, so we choose k = 5 so that it's perfectly divided into 5 subset with 10 samples each, it will help ease the computation.
- the code segment below also shows some more details about the training process.

```
# create the hidden layer model
class SpeedPredictionModel(nn.Module):
    def __init__(self, hidden_layers):
        super(SpeedPredictionModel, self).__init__()
        self.hidden_layers = hidden_layers
        self.fc_in = nn.Linear(3, 32)  #
        self.hidden = nn.ModuleList()
        for _ in range(hidden_layers - 1):
            self.hidden.append(nn.Linear(32, 32))  # hidden_nodes = 32
        self.fc_out = nn.Linear(32, 1)  # output_dim = 1

    def forward(self, x):
        x = torch.relu(self.fc_in(x))
        for hidden_layer in self.hidden:
            x = torch.relu(hidden_layer(x))
        x = self.fc_out(x)
        return x
```

But by actual implementation, it's not that easy that we just take everything described in the original paper. We have to build the model on our own and see, if it fits the given results in the paper. More specifically, we need to make sure that "a model with 3 hidden layers performs the best". What we did are actually 2 parts:

**Finding the best hidden layers value** We first set different values of hidden layer in our model, with a fixed number of hidden nodes in it. Here we take from 1-5 into consideration and then compare the result(as shown in Figure 9)

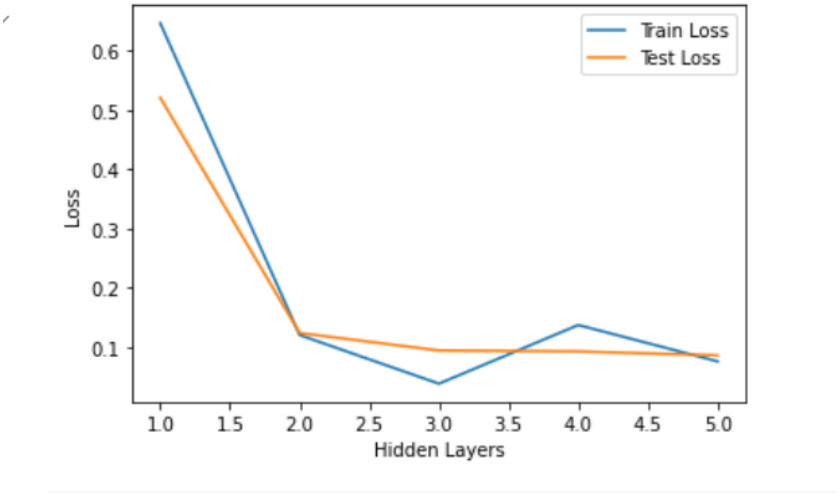



Figure 9: Loss on different hidden Layers

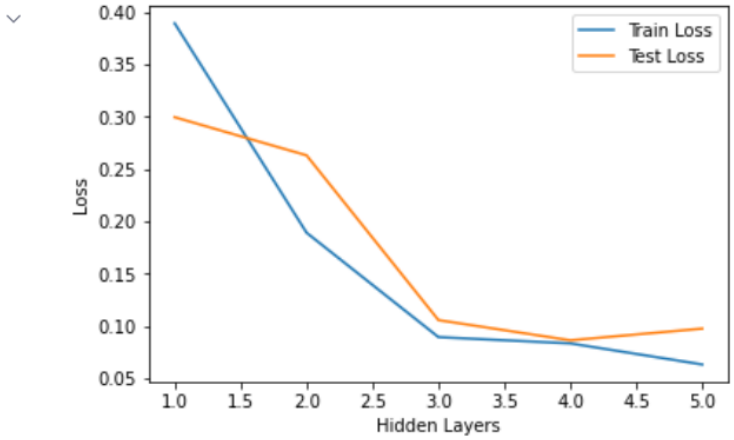



Figure 10: Loss on different hidden Layers

The criterion we take is that not only the train value has to be low, the test value also needs to be low, without showing sign of overfitting. Here in the figure above we can tell that actually when the number of the hidden layer is between 2-4, there is no big difference, but the value of 3 does sometimes show the best result(because of the random sample chosen mechanism), so we finally decided to use Hidden layer = 3.

**Finding the best hidden nodes value** Our next move is to fix the number of hidden layers, and try several different values of hidden nodes, and see if our choice is reasonable. Here we roughly take 3 sets of values, namely [5,10,20],[32,64,128] and [42,84,168], which represents respectively fewer nodes, typical choice of nodes and double amount of inputs. The result can be seen in Figure 11)

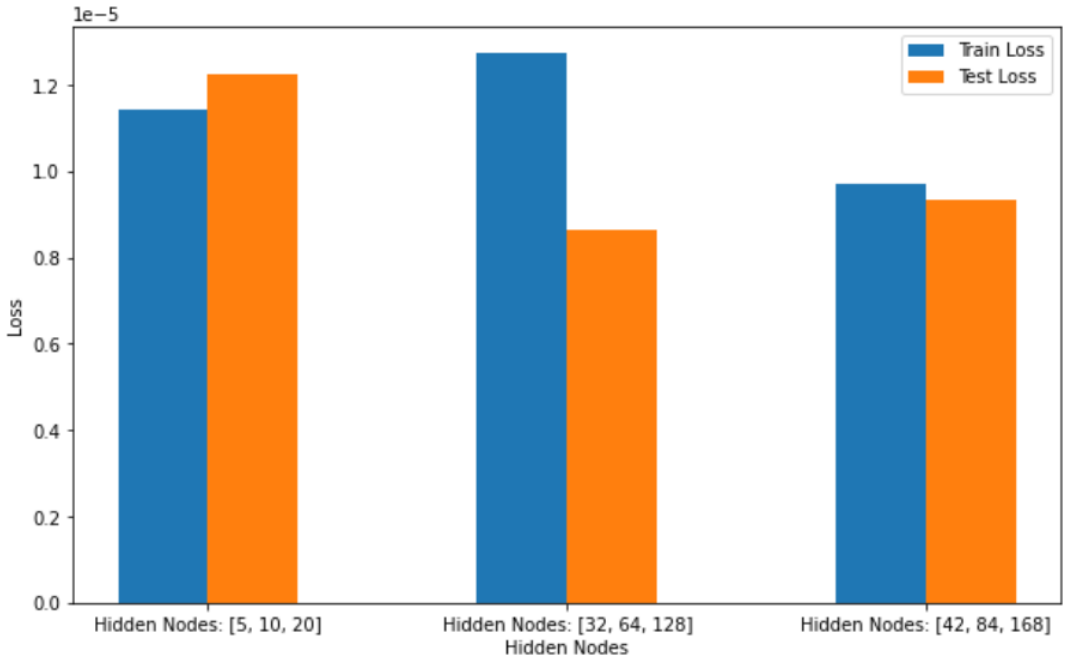



Figure 11: Loss on different hidden nodes

From the figure, we can say that there is no obvious difference between these number groups. In sake of the convenience and generosity, we choose to have [32,64,128] to be our values of nodes for those 3 hidden layers.

Here is to mention that the whole model tuning process is based on Bottleneck dataset, which is also adviced by the original paper. We have also tested the R-dataset-based model, and the results are indeed similar.

**Combining datasets and training** After having the model settled, its about time to train different combinations of datasets and record the results. The process of combination can be shown in the code segment below.

```
# size tuning
df2 = df2.sample(len(df1), replace=False)
df1_half = df1.sample(frac=0.5, replace=False)
df2_half = df2.sample(frac=0.5, replace=False)

# get the loss
train_losses = []
test_losses = []

# dataset combination
train_inputs_1, train_targets_1 = create_train_data(df1[:-1])
test_inputs_1, test_targets_1 = create_train_data(df1[:-1])

train_inputs_2, train_targets_2 = create_train_data(df2[:-1])
test_inputs_2, test_targets_2 = create_train_data(df2[:-1])

train_inputs_combined, train_targets_combined =
create_train_data(pd.concat([df1_half, df2_half]))
test_inputs_combined, test_targets_combined =
create_train_data(pd.concat([df1_half, df2_half]))

train_datasets = [(train_inputs_1, train_targets_1),
(train_inputs_1, train_targets_2),
```

```
(train_inputs_2, train_targets_1),
(train_inputs_2, train_targets_2),
(train_inputs_combined, train_targets_1),
(train_inputs_combined, train_targets_2),
(train_inputs_combined, train_targets_combined)]

test_datasets = [(test_inputs_1, test_targets_1),
(test_inputs_2, test_targets_2),
(test_inputs_1, test_targets_1),
(test_inputs_2, test_targets_2),
(test_inputs_1, test_targets_1),
(test_inputs_2, test_targets_2),
(test_inputs_combined, test_targets_combined)]
```

With some explanation here, in order to make sure that the tensors can match their scales while being calculated, we didn't use all the data in those datasets. We cut some of them to satisfy the matching requirments. (The result of those different combinations will be given in the next section: Model Comparison, alone with the results of the Weidmann model).

**Report on task TASK 4, Model Comparison**

**Why do we need to compare the models?** In one sentence, we want to determine whether ANNs could be suitable tools for predicting pedestrian dynamics in complex geometries. More specifically, we want to compare the performance of Weidmann model and artificial neural network (ANN) model in different scenario settings, under some criteria of our choice, to determine the extent to which ANN models are suitable for predicting pedestrian dynamics in complex geometries.

**What aspects of the two models can we compare?** For the comparison of models, we could focus on the following aspects:

Prediction Accuracy: This is the primary criterion for measuring the quality of the model. We can use it to determine how far are the model predictions to the actual values.

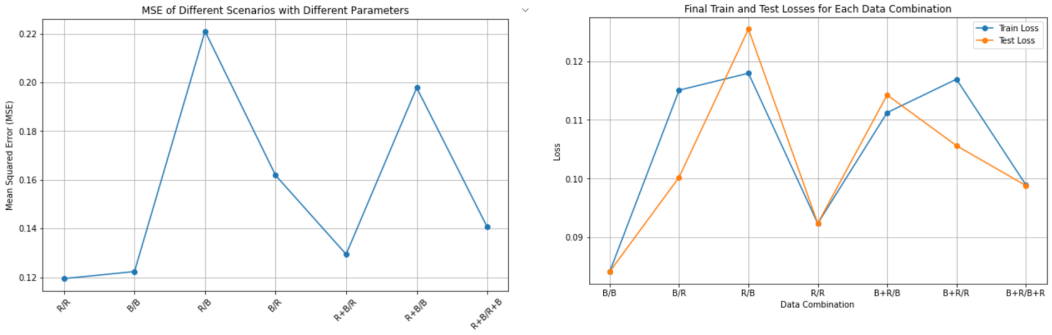
Time/Space Complexity of the Model: This affects the viability of the model in practical applications. If a model, while accurate, requires large computing resources or long computation times, it may not be suitable for real-time or online applications.

Since the data set of this project is not too large, the time and memory space required to train and test the two models are not too much, so in this project we do not compare the performance of the two models in terms of time/space complexity as a standard to measure the quality of the model, but the accuracy of the prediction is used to measure the quality of the model.

**How to measure the accuracy of prediction? What is the experiment setting regarding to this part?** In this project, we use the test MSE(Mean Square Error) as a measure of prediction accuracy. Specifically, we compare the test MSE of Weidmann model and ANN model under different training set and test set settings. The specific settings are as follows:

- R/R and B/B: Train and test the model in Ring (abbreviated as R later) and Bottleneck (abbreviated as B later) scenarios respectively.
- R/B and B/R: Train the model on one scenario and test the model on the other.
- R+B/R, R+B/B: Train the model under the combination of R and B scenarios, and test the model under the R and B scenarios respectively.
- R+B/R+B: Train and test the model under the combination of R and B scenarios.

**Experimental method and operation steps.** We first let the two models be trained on different training sets. During training, we used an appropriate optimization algorithm to minimize the loss function of the model. Once the model is trained, we test it on the corresponding test set and calculate the test MSE. For the ANN model, we used Python's deep learning library Keras for modeling and training. For the Weidmann model, we directly used the formula stated in the paper to model the data.




(a) 1 (b) 2

Figure 12: [1]Test MSEs of Weidmann model under different settings [2]Test MSEs of ANN model under different settings

| **Scenario** | **ANN test MSE** | **Weidmann test MSE** |
|---|---|---|
| R/R | 0.092 | 0.119 |
| B/B | 0.084 | 0.122 |
| R/B | 0.125 | 0.221 |
| B/R | 0.100 | 0.162 |
| R+B/R | 0.106 | 0.129 |
| R+B/B | 0.114 | 0.198 |
| R+B/R+B | 0.099 | 0.140 |

Table 1: Comparison of test MSEs for ANN and Weidmann model under different scenarios

**Experimental Results and Discussion.**

From figure 12 and table 1 we find that the test MSEs of the ANN model is lower than those of the Weidmann model under all settings. This shows that the ANN model has higher accuracy on the task of predicting pedestrian dynamics in complex geometries. As for the reason for why ANN outperforms the Weidmann model in last three settings, I think it is due to its ability of differentiate different patterns behind the two scenarios.

**Report on task TASK 5, Discussion of the approach and architecture**

**Background:** In this study, we compare the two modeling approaches used in Pedestrian Movement Analysis: the fundamental diagram (FD)-based model and a feed-forward artificial neural network (ANN). We evaluate these two approaches according to their ability to accurately describe pedestrian behavior in complex spatial structures and come to an well-informed conclusion about the entire problem space.

**Fundamental Diagram:** The standard technique is FD-based. It depends on three variables: the time gap, the size of the pedestrian, and the speed 1.This model has been frequently utilized in research of pedestrian dynamics because of its simplicity, which also serves as a suitable foundational depiction of pedestrian movement. However, this simplicity has a cost because it finds it difficult to depict the subtleties of behavior in intricate spatial structures and changes across various contexts.

**Artificial Neural Network:** The ANN model is an alternate strategy that makes use of neural networks' strength to identify intricate underlying patterns. The training data came from an experiment [8] in which numerous head tracker-wearing individuals were permitted in particular spatial structures like bottlenecks. One benefit of employing NN in this case was that it could manage a huge number of factors without needing specific physical interpretations, making them more adaptive and versatile to various conditions. They were also able to discover complicated patterns and non-linear interactions that standard models may have difficulty noticing. One of the limitations, that we came across when using NN, was the availability and quality of the training data. In this instance, the experiment data only required a little amount of processing, but if we wanted to represent a more complex spatial structure, such as a train with many bottlenecks, getting the right data would be a big challenge. Additionally, because our research includes tracking pedestrians, we must take into account German and European Union laws regarding public and private data. The neural network architecture's intricacy is another drawback. To prevent either overfitting or underfitting, they must be precisely adjusted. A important step that may result in unnoticed issues is determining the ideal number of hidden layers and nodes.

**Architecture of ANN:** The architecture for the ANN model used in this study is a feed-forward ANN with hidden layers 2. The neural network takes as input the mean spacing between pedestrians and the relative positions of the K closest neighbors. These inputs provide information about the local density and the spatial arrangement of pedestrians. The number of parameters in the network depends on the number of nodes in the hidden layer, which also determines the complexity and capacity of the network to learn and represent the underlying patterns in the data. Since the neural network is feed-forward, data moves through it in one way, from the input layer to the hidden layers, then to the output layer. Each layer is made up of nodes, commonly referred to as neurons, which use activation functions to compute on the input data. So, the network can learn from the input data and extract intricate features and relationships thanks to the hidden layers. During the training phase, the neural network adjusts its internal parameters, known as weights and biases, based on the provided input-output pairs. An optimization procedure, like back-propagation, is used in this process to iteratively adjust the parameters and reduce the discrepancy between the expected and actual outputs. The trained neural network is then used for prediction by feeding new input data into the network and obtaining the corresponding output. In this study, a traditional model built on the fundamental diagram (FD) is contrasted with a feed-forward NN model. The two models performances were then assessed using information from bottleneck and corridor experiments.

**Comparison:** The study's findings demonstrated that in our testing, the neural network performed better at speed prediction than the FD-based model. In comparison to the FD-based model, the neural network was better able to distinguish between various spatial geometries and increased speed estimation by up to 20 percent. The neural network is a promising tool for upcoming studies on pedestrian dynamics due to its capacity to capture complicated patterns and linkages. However, more investigation is required to examine the neural network's performance over whole trajectories and compare it to other existing models.

**Report on task Summary**

**Introduction** Urban planning, transit management, and crowd management all rely heavily on pedestrian movement analysis (PMA). It enhances safety, improves traffic flow, and enhances pedestrian facilities. However, due to a variety of pedestrian behaviors, a complicated spatial layout, and crowd dynamics, PMA encounters difficulties. Researchers have studied the relationship between pedestrian density and velocity using the fundamental diagram (FD) method. This relationship is described by the Weidmann model, based on mean spacing, velocity, time gap, and a parameter. The behavior of pedestrians in a variety of situations, including corridors, crossroads, and bottlenecks, has been studied experimentally. These studies provide insights into the characteristics of pedestrian movement by demonstrating the impact of boundary conditions on pedestrian flow dynamics. A promising method for predicting pedestrian behavior in complex geometries is provided by artificial neural networks (ANN). ANN models perform better than conventional models because they can use several parameters and can spot patterns. In bottleneck and corridor tests, these models demonstrated considerable gains in speed forecast accuracy, up to 20 percent better than the aggregated models. The application of recurrent neural networks (RNN) and long short-term memory (LSTM) models for trajectory prediction is one of the new areas of research that might be pursued in light of the success of ANN in pedestrian movement analysis. This study emphasizes how ANN might help us better understand pedestrian behavior and develop autonomous systems.

**Discussion** In this study, a feed-forward artificial neural network (ANN) and a fundamental diagram (FD)-based model for pedestrian movement analysis were evaluated. The FD-based model represents pedestrian behavior simply but struggles to capture complexities in intricate spatial layouts. The ANN model, on the other hand, employs neural networks to recognize complex patterns and take into account a greater number of variables without a clear physical explanation. In terms of speed prediction, the study discovered that the neural network performed better than the FD-based model, with gains of up to 20 percent. The neural network is a promising tool for future research on pedestrian dynamics because of its capacity to recognize various spatial geometries and capture intricate patterns. To assess its performance throughout whole trajectories and contrast it with other models already in use, more investigation is required.

# References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. pages 961–971, 2016.

[2] JL Beniley. Multidimensional binary seareh trees used for assoeiative searehing. *ACM Communications*, 18(9):509–517, 1975.

[3] Avijit Hazra. Using the confidence interval confidently. *Journal of thoracic disease*, 9(10):4125, 2017.

[4] Armin Seyfried, Bernhard Steffen, Wolfram Klingsch, and Maik Boltes. The fundamental diagram of pedestrian movement revisited. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(10):P10002, oct 2005.

[5] Antoine Tordeux, Mohcine Chraibi, Armin Seyfried, and Andreas Schadschneider. Prediction of pedestrian speed with artificial neural networks. 2018.

[6] Ulrich Weidmann. Transporttechnik der fußgänger: transporttechnische eigenschaften des fußgängerverkehrs, literaturauswertung. *IVT Schriftenreihe*, 90, 1993.

[7] Ulrich Weidmann. Transporttechnik der fussgänger. transporttechnische eigenschaften des fussgängerverkehrs, literaturauswertung. *IVT Schriftenreihe*, 90, 1993-03.

[8] Jun Zhang and Armin Seyfried. Experimental studies of pedestrian flows under different boundary conditions. pages 542–547, 2014.