Mission-Aware Spatio-Temporal Deep Learning Model for UAS Instantaneous Density Prediction

Ziyi Zhao[1], Zhao Jin[1], Wentian Bai[1], Wentan Bai[1], Carlos Caicedo[2], M. Cenk Gursoy[1], Qinru Qiu[1]
[1]Department of Engineering and Computer Science, Syracuse University, Syracuse, New York, USA
[2]School of Information Studies, Syracuse University, Syracuse, New York, USA
Email: [1]{zzhao37, zjin04, wbai02, wbai03, mcgursoy, qiqiu}@syr.edu,[2]ccaicedo@syr.edu

***Abstract*—The number of daily sUAS operations in uncontrolled low altitude airspace is expected to reach into the millions in a few years. Therefore, UAS density prediction has become an emerging and challenging problem. In this paper, a deep learning-based UAS instantaneous density prediction model is presented. The model takes two types of data as input: 1) the historical density generated from the historical data, and 2) the future sUAS mission information. The architecture of our model contains four components: Historical Density Formulation module, UAS Mission Translation module, Mission Feature Extraction module, and Density Map Projection module. The training and testing data are generated by a python based simulator which is inspired by the multi-agent air traffic resource usage simulator (MATRUS) framework. The quality of prediction is measured by the correlation score and the Area Under the Receiver Operating Characteristics (AUROC) between the predicted value and simulated value. The experimental results demonstrate outstanding performance of the deep learning-based UAS density predictor. Compared to the baseline models, for simplified traffic scenario where no-fly zones and safe distance among sUASs are not considered, our model improves the prediction accuracy by up to 15.2% and its correlation score reaches 0.947. In a more realistic scenario, where the no-fly zone avoidance and the safe distance among sUASs are maintained using A* routing algorithm, our model can still achieve 0.822 correlation score. Meanwhile, the AUROC can reach 0.951 for the hot spot prediction.**

***Index Terms*—instantaneous density prediction, UAS, mission aware, spatio-temporal model**

I. Introduction

Recently, many companies, such as DJI, Lockheed Martin and Amazon, devote themselves to develop small Unmanned Aircraft Systems (sUAS). Complicated and high density UAS traffic imposes significant burden on air traffic management, city planning and communication resource allocation. Under this environment, the following critical questions are usually asked: Given the list of scheduled launches in an area, do we know in advance whether a feasible route in terms of air space safety and energy efficiency can be found for a specific mission at a specific time? Do we need to delay the launch of some sUAS in advance to accommodate a mission with higher priority scheduled at a specific time? Answering such questions and being able to predict the traffic distribution ahead of time will provide an opportunity for more efficient planning and control.

UAS density prediction is a critical and challenging problem in the Unmanned Aircraft System Traffic Management (UTM) system. Most existing studies focus on simulation-based approaches. Although accurate, they usually take a long time to deliver results. Neural networks have been used to predict the traffic density. However, most such studies require the sampling of the traffic density from the past data and predict the future density using past density information. These models assume a static environment. For example, the source (i.e. the location where the sUAS enters the air space) and sink (i.e. the location where the sUAS leaves the air space) of the traffic flow are assumed to remain the same, and air space constraints, such as no-fly zones, are fixed. Based on these assumptions, the traffic in the future will exhibit similar pattern as the traffic in the past, and can be predicted from the historical data. A constant environment may be reasonable for road traffic, however, the operational environment of sUAS features higher dynamics and flexibility. For instance, the no-fly zones may change due to construction or special activities/events, launching or landing zones may be added or removed. The model based on historical data will become obsolete as soon as the environment changes. New data must be collected and a new model needs to be trained, which can take days or weeks. Furthermore, most of the existing models consider traffic distribution as a stationary process, and focus on predicting the steady states. For resource provisioning or safety assurance, we need to know not only the steady state traffic but also the worst case traffic. Hence the ability to predict the transient behavior of air traffic distribution is highly desirable. There are a few works that utilize the long short-term memory (LSTM) model to predict future traffic based on recent traffic activities, however, their prediction horizon is very limited. Accurate prediction cannot be made beyond 4 or 5 timestamps.

In this paper, a deep learning-based prediction model is presented for semi-transient traffic density distribution prediction. The model takes the air space environment and the pre-scheduled launch list in the next T time units as the inputs, and predict the average traffic density of traffic distribution in this air space during time [T-δ, T]. The parameter T controls the prediction horizon and by reducing the value of parameter δ , the focus of the model changes from the long-term average behavior to transient behavior of the traffic. By taking the flight environment and detailed launch information as part of the inputs, the model is specific only to the type of trajectory planning algorithms. It can be generalized to different air space environment as long the trajectory of each UAS is routed using the same algorithm. It will have no "down time" after the map or the launching/landing zone has changed.

This work is partially supported by the NSF IUCRC ASIC and collaborated with our industry partner Thales Group.

The model has high prediction accuracy. Compared with other existing methods, our model can achieve a correlation score of 0.947 and can improve the prediction accuracy by up to 15.2%. In a realistic traffic scenario, where no-fly zones avoidance and safe distance between sUASs are considered by planning the trajectory using A* routing algorithm [1], our model can still achieve a correlation score of 0.823. The following summarizes the major contributions of our work:

- A novel UAS traffic density prediction model is developed that captures the information from the historical data and the pre-scheduled sUAS launch list.
- A novel input representation of the future sUAS mission information is proposed. The pre-scheduled missions are categorized into 3 types according to their launching times. Our model is designed to extract features from all type of inputs simultaneously. The learnable parameters are introduced to adjust the degrees affected by different types of features.
- Compared to the baseline models, our model improves the prediction accuracy by up to 15.2%. When doing hot spot prediction, our model can achieve an AUROC score of 0.951.

II. Related Works

Over the past decade, the unmanned aerial vehicles have played an increasingly essential role in many areas [2] [3] [4] [5]. With the rise in the popularity of sUAS, many notable issues telated to UAS traffic management have been discovered. However, most of them explored the novel applications for the single sUAS or formulated the UAS management policies. The study of the UAS cluster behaviors such as forecasting of the UAS traffic density has generally not been addressed. In our investigation, the density forecasting approaches can be categorized into simulation based method and deep-learning based method. In this section, we will analyze the pros and cons of recent works in these two categories.

Many existing works study issues such as sUAS navigation, obstacle avoidance or UAS traffic management, by developing a corresponding simulator with fair time complexity. In [6], the authors presented an indoor algorithm to navigate single sUAS to avoid collisions. [7] proposed a solution to avoid collisions in a static environment by importing geometrical constraints. Other single sUAS classical approaches applied rapidly-exploring random trees [8] and Voronoi graphs [9] [10]. Multiple sUAS trajectory simulation has been studied as a multi-agent cooperative system and solved in a rolling horizon approach using dynamic programming [11] or mixed integer linear programming [12]. Other strategies [1] [13] involved real-time routing algorithms with communication and airspace safety considerations. Recently, a very strict and rigid airspace structure to handle dense operation in the urban low altitude environment was suggested by the work on Unmanned Air craft System (UAS) Traffic Management (UTM) at NASA in [14]. The authors explored UAS operations in non-segregated air space and managed the risk of mid-air collision to a level deemed acceptable to regulators. In the paper, the airspace was divided into multiple layers, and the layers were further divided into orthogonal sky lanes. There are no current works that solve the traffic prediction problem from a big picture perspective within a small running time.

In this work, instead of developing a simulator, we utilize deep learning for UAS traffic density prediction. The deep learning based approach has shown an outstanding success in many areas [15] [16] [17] . Similar multi-agent works are addressed in other fields such as pedestrian density prediction [18] and autonomous driving [19]. In [20], the author proposed a LSTM based scene-aware model to predict trajectories for autonomous driving. However, the prediction errors grew exponentially as the time horizon increased. Another work addressed pedestrian traffic flow prediction by fusing historical information, but the prediction is limited by historical data regardless of upcoming event information. Existing single agent trajectory prediction works concentrated on the behavior of a single sUAS and the impact of environment conditions, without any sUAS cluster consideration. For example, [21] proposed a LSTM-based flight trajectory model with weather considerations taken into account. [22][23] aimed to solve environment navigation problems, and developed a reinforcement learning model to plan energy efficient waypoint in a static environment.

Compared to the existing work, our model has the ability to learn and extract the information from the historical data and the pre-scheduled sUAS mission launch list. And the error is restrained strictly via dynamic feature extraction. By adopting a novel channel segmentation approach, our mission feature extraction module can learn the density features accurately.

III. Methods

Our density prediction model is an end-to-end model where each module is fully differentiable. The mean square error (MSE) loss is calculated by measuring the difference between the predicted density map and the labeled density map. The architecture of our model contains four components: Historical Density Formulation module, UAS Mission Translation module, Mission Feature Extraction module, and Density Map Projection module. The model structure is depicted in Figure 1. In this section, each component of the model will be elaborated.

A. Historical Density Formulation

The historical density describes the pre-existing air space environment. The size of the historical density map is $100*100$ grid units which is the same as the simulation environment. The value at each pixel is between 0 and 1 and represents the average density in the past m simulation cycles. The value of m is set to be 10 in this paper. The historical density will also be called "initial density" in this paper. Given the historical density, we employ a convolutional neural network (CNN) to extract the relevant features. The model is composed of 3 convolution layers, 2 pooling layers and the ReLU activation layers. Finally, the $C * 32 * 32$ feature maps are obtained from the feature extractor, where C is the number of feature channels. The feature extracted from the historical density is denoted as X_h.

B. UAS Mission Translation

This module is responsible for translating the UAS missions to the image representation. First, the UAS future missions

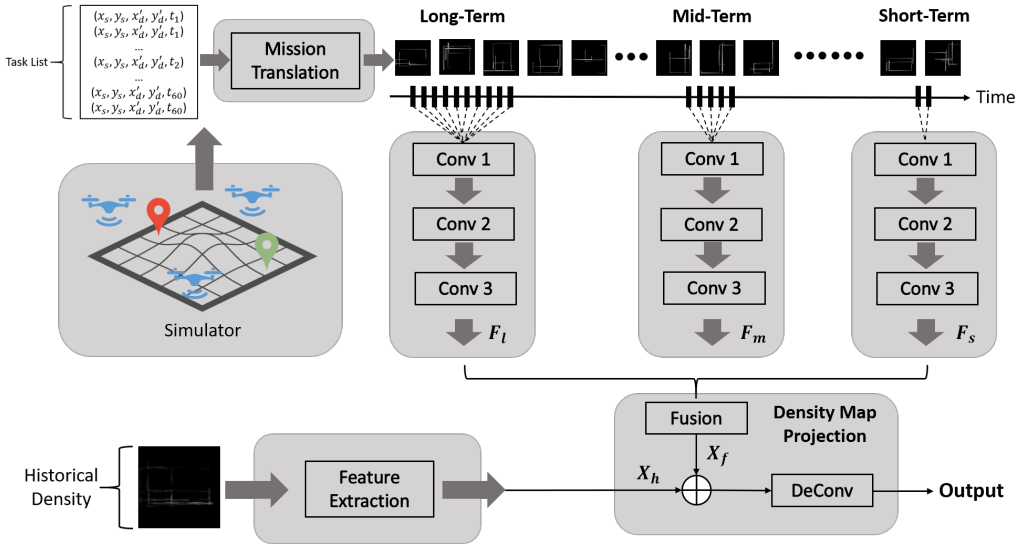


Fig. 1: Mission-Aware Spatio-Temporal Model Architecture

are summarized into a mission list. The dimension of the mission list is $n * 5$ where n is the number of missions in the future. Each mission is defined by a 5 dimensional vector: $\{X_s, Y_s, X'_d, Y'_d, T_n\}$. The $\{X_s, Y_s\}$ and the $\{X'_d, Y'_d\}$ represent the launching and landing locations of the mission. The launching time is indicated by T_n. Given the mission list, the model will first cluster the missions into different groups based on the mission launching time. The mission translation module follows in the same manner as the BFS algorithm to map the trajectory into a $2D$ map. From each Origin-Destination (O-D) pair, we draw a shortest path from the launching location to the landing location. For each individual mission, we assume that the horizontal direction movement has a higher priority than the vertical direction movement. The movement priority is the same as the MATRUS simulator [24]. After the UAS mission translation procedure as mentioned above, a K channel output can be obtained. Each channel lumps the trajectory information of the sUAS that will be launched at the same simulation cycle. K is set to be 60 in this paper.

Moreover, we introduce a novel sUAS trajectory representation approach, which we call as "Flow". The "Flow" input representation uses an ascending sequence to represent the sUAS movement from the launching location to the landing location. Therefore, in the $2D$ map, the waypoints near the landing location are brighter than the waypoints near the launching location. If one location is occupied by more than one sUAS, we use the mean of all the overlapped values to represent this location. By using this input representation, the model can distinguish launching and landing locations. In addition, the order of the sUAS movement is also specified. Figure 2 shows two mission translation examples.

C. Mission Feature Extraction

The translated UAS missions are fed into the mission feature extraction module. This module is responsible for learning the density features from the pre-scheduled missions. Inspired by [25], we develop a novel channel segmentation model. First, according to the mission launching time, the translated

(a) Translation 1 (b) Translation 2

Fig. 2: UAS Mission Translation Examples

missions are categorized into 3 groups: long-term, mid-term and short-term. In our case, the long-term group contains the launching missions from cycle 1 to cycle 30. The mid-term group involves the launching missions from cycle 31 to cycle 50. The rest of the launching missions, cycle 51 to cycle 60, belong to the short-term group. Then, three types of models with different number of input channels are employed to extract the features from the inputs. The number of input channels for long-term, mid-term and short-term models are 10, 5 and 2, respectively. Each individual model has the same structure but the weight will be updated independently. The intuition of the model architecture design is that the mission whose launching time is close to the end should have a larger impact on the final density. The type of convolution (2D or 3D) operation we applied in this module will be discussed in Section V-C. Then, three types of features (long-term feature, mid-term feature and short-term feature) can be obtained, which are denoted as $\{F_l, F_m, F_s\}$. In the fusion module, the learnable parameters are introduced to adjust the degrees affected by different features. Therefore, the mission from different times will contribute accordingly to the final density. The fusion equation is defined as follows:

$$X_f = \{W_1 * F_{s\_1} + ...W_k * F_{m\_1} + ... + W_n * F_{l\_n}\} \quad (1)$$

where W denotes the learnable parameters. The output density feature is denoted as X_f. And, the F_s, F_m and F_l are the

features extracted from short-term input, mid-term input and long-term input, respectively.

Consequently, a $C*32*32$ feature map is obtained from this module, where C stands for the number of feature channels.

D. Density Map Projection

Finally, two features $\{X_h, X_f\}$ are concatenated together to construct a fused density feature representation. Then, we apply a de-convolution module to project the density feature into a $2D$ density map that has the same width and height as the whole simulation environment. The de-convolution module is composed of 4 2D-transpose layers, batch normalization layers and the ReLU activation layers. The value at each location stands for the average density at the given prediction time T_1. In this paper, T_1 is set to be 10.

IV. Experiments

A. Data Generation

Inspired by the MATRUS framework [24], we implement a python based sUAS flight simulator. For each traffic scenario tested in this paper, we ran the simulator to generate 3000 samples. All the data sets are divided into two subsets: training and testing. The split ratio is $90 : 10$. For each sample, the simulator randomly generates 5 launching areas and 5 landing areas on a 100*100 grid environment. Each area has the size 3*3. Any grid in this area can be considered as the launching location. The minimum distance between any two areas is 5 cells. For each launching area, the simulator uses uniform distribution to randomly generate a float number as the launch probability. At every simulation cycle, the simulator randomly selects 15 launching locations from all launching areas. For each selected location, the simulator randomly decides whether a mission should be launched from current location at current cycle based on the launch probability.

For each sample, the simulation time horizon is defined as T. In this paper, T is set to be 60 simulation cycles, and each cycle lumps sUAS launching information in 10 seconds. The time period that generates the density map will be described as T_1. The data generation procedure is depicted in Figure 3.

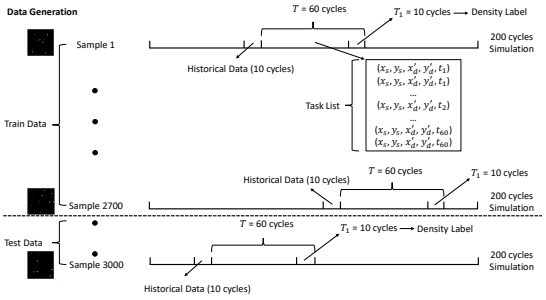


Fig. 3: Data Generation Procedure

B. Evaluation Metrics

Two metircs are used to measure the quality of the prediction.

(1) Correlation: Correlation is calculated between the simulated traffic density (Y), which is considered as the ground truth, and the predicted traffic density $(\hat{Y})$. In our experiments, it shows whether and how strongly the predicted and labeled variables are related. The equation of the Correlation is as follows:

$$\rho(X,Y) = \frac{Cov(X,Y)}{\sigma_X \sigma_Y} \tag{2}$$

where $\rho(X,Y)$ is Pearson's correlation coefficient of X and Y, σ_X and σ_Y are the standard deviation of X and Y, respectively. $Cov(X,Y)$ is the covariance of variables X and Y, which can be calculated by the following equation:

$$Cov(X,Y) = E[(X - E[X])(Y - E[Y])] \tag{3}$$

where $E[.]$ denotes the expected value.

(2) Area Under the Receiver Operating Characteristics (AUROC): The ROC curve is plotted by mapping the True Positive Rate (TPR) against the False Positive Rate (FPR) with different thresholds. Given a ROC curve, the AUROC evaluates the performance of the model by distinguishing between classes. The higher the AUROC score of a model, the better the performance is. For an uninformative model, the AUROC is close to 0.5. The maximum AUROC is 1.

C. Comparison Models

To the best of our knowledge, there is no prior work considers the exactly same application as this paper. For comparison, we selected some existing models that are potentially promising for traffic prediction, and re-trained them using our data set. We also compared with some modified version of our own model to show the effectiveness of certain design decisions of our model. The following five models are tested and compared.

- Vanilla CNN (VCNN): This is a typical CNN based encoder-decoder model. The model assumes that, the location and action probability of each launching/landing area is static and can be represented in a 2D map. It tries to learn the relation between traffic density and the 2D map, and makes prediction based on the static information.
- Vanilla LSTM (VLSTM): This is a typical LSTM based encoder-decoder model. It takes the T cycles scheduled launching information and predict the density at the $T+1$ cycle. Because the traffic density of cycle $(t+1)$ is determined by the density at cycle t and the current launching information, it was expected that such temporal dependency can be captured by an LSTM model.
- RouteNet [26]: The RouteNet model is encouraged by the Fully Convolutional Network (FCN) architecture that predicts the congestion in VLSI placement and route. And the FCN allows input to be any size and produces an output with exactly the same size as input, indicating the density (or hotspot) at any location.
- Segmented Channel: This is our model discussed in this paper. The inputs are categorized into 3 groups. Then, the designated models are assigned to each group for extracting the features. The model applies the 2D convolution with 2 * 2 and 4 * 4 kernel size. The max pooling is also used in the model.
- All Channel: This model has the similar structure as presented model except that there is no input channel segmentation. The model treats all the missions from different launching time as the same.

D. Experiment Setup

We run our experiments on a desktop server running Ubuntu 16.04 OS with 3.60GHz Intel Xeon W-2123 CPU, 256GB Memory and a NVIDIA 2080Ti GPU. During the training, the Adam optimizer is applied with a 0.005 learning rate. The weight decay is set to 0.0001. We use the ReLU to be the activation function. Batch normalization and dropout are also applied for preventing the overfitting. The training and testing framework are built in PyTorch.

V. Results

A. Predicted Density Accuracy Improvement

In the first experiment, we compare the accuracy of the density prediction between our model and other baseline models from Section IV-C. The model which extracts the trajectory features from all input channels is denoted as "$channel_{all}$". Our presented channel segmentation model is denoted as "$channel_{seg}$". Because the prediction model assumes a non-empty air space, we are interested to know how close the initial traffic density resembles the density at the target time of prediction. The column "$init$" gives the correlation between the initial density and the label density.

TABLE I: The Correlation Score of the Density Prediction

Init	VCNN	VLSTM	RouteNet	$Channel_{all}$	$Channel_{seg}$
0.822	0.863	0.803	0.889	0.944	**0.947**

Table I shows the correlation score for all the models. The LSTM model has the worst performance among all the models. The correlation score of the "$VLSTM$" model is even lower than the "$init$" correlation score. One reason for this is that the 60 cycle prediction period is too long for the "$VLSTM$" model. The error will accumulate and propagate from the first cycle to the last cycle. The "$VCNN$" model improves the correlation score by 5.0%. However, ignoring the information of exactly when and where each sUAS is going to be launched and where it is heading from now to the end of prediction window makes the prediction much less specific. Therefore, the "$VCNN$" model cannot achieve a higher accuracy. In the "$RouteNet$" model, each scheduled mission will be marked by a bounding box between the Origin-Destination (O-D) pair. This approach gives the model a more forthright indication of each mission and the relation between the launching and the landing locations. Consequently, the "$RouteNet$" model improves the correlation score by 8.2%. Finally, our presented model, "$channel_{all}$" and "$channel_{seg}$", outperforms all other models. Compared to the initial traffic density, the predicted density of these models clearly resembles the actual density at the end of prediction window better, with 14.8% and 15.2% improvement of the correlation score, respectively.

Between the two model architectures that we proposed, the "$channel_{seg}$" model can achieve higher correlation score than the "$channel_{all}$" model. However, the difference is very marginal. In the next, we will show that using segmented channel is important under the scenario when the process of UAS launching is non-stationary.

B. The Impact of the Initial Density

In the second experiment, we study how the initial density affects the density prediction and evaluate the robustness of two model architecture designs. Two test scenarios have been designed:

- Without Training (w/o training): We follow the same training procedure in Section V-A. However, during the testing, the initial density is not provided. Instead an all-black image (i.e. an empty air space) is provided.
- With Training (w training): In the training phase, the initial density is also replaced by the all black image.

TABLE II: The Impact of the Initail State

	w/o training	w training
$init$	0.822	0.822
$channel_{all}$	0.885 (+7.7%)	0.894 (+8.8%)
$channel_{seg}$	0.913 (+11.1%)	0.924 (+12.4%)

The results in Table II show that the model performance is affected by the historical density. Nonetheless, compared to the "$channel_{all}$" model, the "$channel_{seg}$" model is more robust. In Section V-A, the correlation scores are 0.944 and 0.947 respectively for "$channel_{all}$" model and the "$channel_{seg}$" model. Without the training, the correlation score drops to 0.895 and 0.913 respectively for "$channel_{all}$" model and the "$channel_{seg}$" model. This proves that the "$channel_{seg}$" model can extract more meaningful features from the pre-scheduled missions. Even with the training, two models can only achieve correlation scores of 0.894 and 0.924. This proves that the historical density map can help the model improve the prediction accuracy. This phenomenon shows that, compared to the "$channel_{seg}$" model, the "$channel_{all}$" model relies more on the historical density.

C. The Model Sensitivity to Missions

In the third experiment, we study the model's sensitivity to missions. In our assumption, the most recent missions should have a larger contribution to the predicted density than those that took place earlier in time. And the model should be able to capture the features from the non-stationary missions. In order to further analyze this conjecture, we test different model architectures and design a specific experiment. In the model architecture design, we test both 2D convolution operation and 3D convolution operation to be the feature extractor backbone. For the experiment, we use the normal mission list as the input in the model training phase. The normal mission list means that all the 60 cycles have the launching missions. However, in the testing phase, we remove the missions from either the first 30 cycles or last 30 cycles. Therefore, the experiments are broken down into 2 scenarios:

- No Task Before 30 (NTB_30): No new launching mission from cycle 1 to cycle 30.
- No Task After 30 (NTA_30): No new launching mission from cycle 31 to cycle 60.

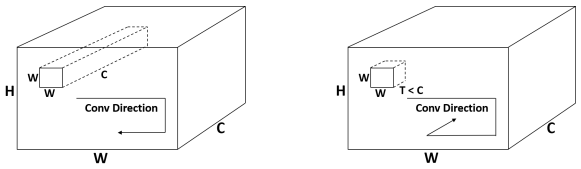
Table III shows the correlation score in both scenarios. The "$init$" stands for the correlation score between the initial density and the label. We take the "$channel_{all}$" and the "$channel_{seg}$" to be two comparison models. The testing data

TABLE III: Experiments comparing 2D and 3D convolution

Scenario (3D)	NTB_30	NTA_30
$init$	0.699	0.647
$channel_{all}$	0.554 (-21.3%)	0.532 (-17.8%)
$channel_{seg}$	0.712 (+1.9%)	0.678 (+4.8%)

Scenario (2D)	NTB_30	NTA_30
$init$	0.699	0.647
$channel_{all}$	0.765 (+9.4%)	0.751 (+16.1%)
$channel_{seg}$	**0.836** (+19.6%)	**0.858** (+32.6%)

is the same for both 3D convolution operation and 2D convolution operation. Therefore, the correlation scores between the initial density and the label are the same for both scenarios. Compared to the result in Section V-B, without the training, the prediction accuracy drops for both models. However, from the results, we can notice that the performance drop with 2D convolution operation is less severe in comparison. The correlation score of the "$channel_{all}$" model with 3D convolution operation is even lower than the "$init$". The reason is that the features from the most recent time and early time periods are not distinguishable as the 3D convolution uses the same cube filter for all cycles. However, for the 2D convolution, each filter has a spatial extent. The number of spatial extents is equal to the number of input channels. The spatial extent increases the representability of the model. The demonstrations of the 2D and 3D convolution operations are given in Figure 4. This suggests that using 2D convolution layers to extract features from each channel in our scenario is a more robust approach compared to using the 3D convolution. Consequently, we choose the 2D convolution operation to be the backbone of the feature extractor in our final model design.



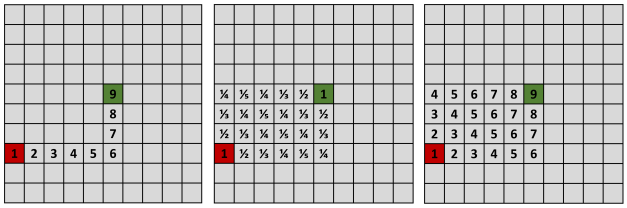
(a) 2D convolution (b) 3D convolution

Fig. 4: Convolution Operation Comparison

The second observation from the result is that, the "$channel_{seg}$" model always has a better prediction performance than the "$channel_{all}$" model. With $2D$ convolution operation, compared to the "$init$", the "$channel_{seg}$" model can achieve 19.6% and 32.6% correlation score improvement in NTB_30 and NTA_30, respectively. However, the "$channel_{all}$" model can only achieve 9.4% and 16.1% improvement. This result is consistent with our hypothesis at the beginning of the section. In our "$channel_{seg}$" model, the missions with different launching time can be distinguished. Our model is capable of learning the meaningful features from the non-stationary missions.

D. Density Prediction with No-Fly Zone Avoidance and Routing Algorithm

In the fourth experiment, we introduce no-fly zones into the simulation environment. In each batch of the simulation, the ratio of grids which are occupied by a no-fly zone is varying from 5% to 45%. By applying the routing algorithm, the simulated sUAS is capable of avoiding the no-fly zone and other sUASs. Hence, the sUAS trajectory is more heuristic and that leads to a more challenging density prediction task. In order to reach a high prediction accuracy, we investigate three potential input representations at the same time. The "Flow" input representation has been presented in Section III-B, as shown in Figure 5(a). In the second input representation, we draw a bounding box to incorporate the launching and landing grids of each sUAS. The launching/landing grids are located at the two corners of the bounding box. The value in each grid represents the needed steps to move from the launching location, as shown in Figure 5(c). Therefore, we call it "Ones" input representation. For the "Probability" input representation, we use the same bounding box to incorporate the launching and landing grids. However, the value in each grid stands for the probability that the sUAS moves from its previous adjacent grid to the current grid, as shown in Figure 5(b).



(a) Flow (b) Probability (c) Ones

Fig. 5: Different Input Representation

TABLE IV: Density Prediction with No-Fly Zone Avoidance and Routing Algorithm

Correlation	Flow	Probability	Ones
$init$	0.698	0.698	0.698
$channel_{all}$	0.795 (+13.9%)	0.818 (+17.2%)	**0.821** (+17.6%)
$channel_{seg}$	0.798 (+14.3%)	0.820 (+17.5%)	**0.822** (+17.8%)

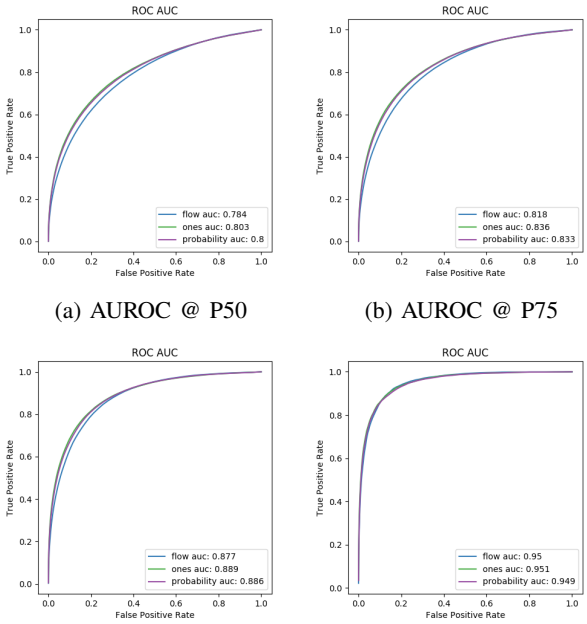
Table IV shows the correlation scores for the different representations. From the result, we can notice that both models can improve the correlation in all the input representation types. However, the performance of the "$channel_{seg}$" model is slightly better than all-channel model. Compared to the correlation score of the initial density, our presented "$channel_{seg}$" model can improve the correlation score up to 14.33% in "Flow" input representation, up to 17.34% in "Probability" input representation and up to 17.77% in the "Ones" input representation. The "Ones" input representation outperforms other two representations due to two reasons: 1) The routing algorithm is used in the simulation, therefore, the potential sUAS trajectories are more heuristic. Although the "flow" representation has the ability to indicate the sUAS movement, the flexibility of the model is also reduced by given only one possible path. 2) Compared to the "Probability" representation, the "Ones" representation does not only have all the possible trajectories, but also indicate the moving order of the sUAS.

Besides the correlation evaluation, we also apply the AUROC to evaluate the performance of the "$channel_{seg}$" model. In this experiment, the pixel whose value that is not zero in the label is considered to be the evaluation reference as we are more interested in the high dense area on the map. The P estimation is a popular method in financial risk assessment and internet congestion investigation. Hence, we employ the $P50$, $P75$, $P90$ and $P99$ to select the threshold. After the threshold is defined, the pixel in the label whose value is larger than the threshold is binarized to 1, and vice versa. Table V shows the selected threshold for different P values. For the prediction, the threshold value is sampled progressively from 0.0 to 1.0, with the 0.01 granularity.

TABLE V: Threshold Selection in Different P Value

	P50	P75	P90	P99
$Threshold$	0.2	0.3	0.5	0.8

Figure 6 shows the AUROC in different thresholds. As we can see from the figure, the "Ones" input representation still outperforms other methods. The $P50$ means that half of the UAS flight areas are considered as the hot spot. In this strict circumstance, the AUROC of the "Ones" representation can still achieve 0.803. However, in reality, the severity is often exaggerated by choosing the $P50$. For the $P75, P90, and\ P99$, the AUROC of the "Ones" representation are 0.836, 0.889 and 0.951, respectively. This result further proves that our model is capable of making an accurate hot spot prediction.



(a) AUROC @ P50 (b) AUROC @ P75

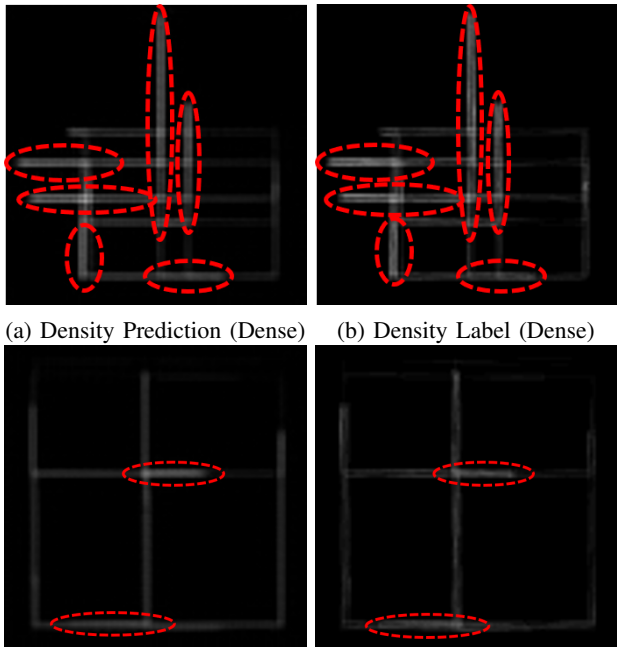
(c) AUROC @ P90 (d) AUROC @ P99

Fig. 6: AUROC at Different P Value

E. Density Prediction Visualization

In the last experiment, we visualize several UAS density predictions to give a qualitative demonstration of our model. We select two typical scenarios to test the performance: a) dense traffic, b) sparse traffic. Both scenarios are tested with/without the routing algorithm. Figure 7 shows the prediction results without the routing algorithm. The left side figures are the prediction and the right side figures are the label. The value of each pixel varies from 0 to 1, representing the average density in T_1. The brighter area means that there are more sUAS passing through this location. The density prediction with the routing algorithm is shown in Figure 8.

From Figure 7 we can notice that, our predicted densities are close to the label in both scenarios. In the dense traffic scenario, all 6 dense areas are predicted by our model, as shown in Figure 7(a) and Figure 7(b). Although some of the dense areas are close with each other, the model can still predict them clearly. In the sparse traffic scenario, there are only two horizontal dense areas. One is in the middle of the map, the other is at the bottom of the map. Both of them are predicted accurately by our model.



(a) Density Prediction (Dense) (b) Density Label (Dense)

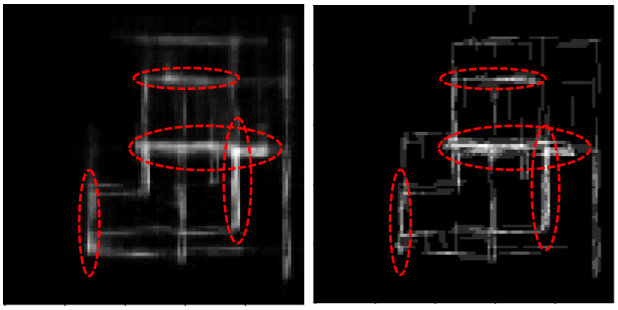
(c) Density Prediction (Sparse) (d) Density Label (Sparse)

Fig. 7: Density Prediction without Routing Visualization

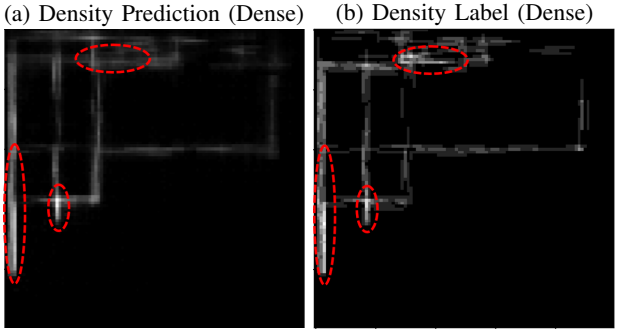
After the routing algorithm is introduced, the sUAS trajectory is heuristic which will lead to a more random density distribution. Both prediction and label become blurry in this situation. Under this circumstance, our presented model can still predict the most obvious dense areas. In Figure 8(b), there are four obvious dense areas which are marked by the red dash circle. Figure 8(a) shows that all of the dense areas are predicted successfully by our model. In the sparse traffic scenario, there are three obvious dense areas in Figure 8(d). Although the model fails to predict the dense area at the top of the map, two other dense areas at the left bottom have been predicted successfully, as shown in Figure 8(c).

VI. Conclusions

In this paper, we have proposed a novel mission-aware spatio-temporal model, which aims at predicting the UAS instantaneous density. The model has the ability to extract



(a) Density Prediction (Dense) (b) Density Label (Dense)



(c) Density Prediction (Sparse) (d) Density Label (Sparse)

Fig. 8: Density Prediction with Routing Visualization

meaningful features from the given historical density and learn the information from the pre-scheduled missions. In the experiment section, we use the correlation score and the AUROC to evaluate the prediction accuracy of our proposed model. Compared to the baseline models, for simplified traffic scenario where no-fly zones and safe distance among sUASs are not considered, our model improves the prediction accuracy by up to 15.2% and its correlation score reaches 0.947. The results in Section V-C show that our model is sensitive to the pre-scheduled missions and has the ability to predict the transient behavior of the traffic distribution. In a more realistic scenario, where the no-fly zone avoidance and the safe distance among sUASs are maintained using A* routing algorithm, our model can still achieve a correlation score of 0.822. Moreover, the AUROC results demonstrate that the hot spot predicted by our model is accurate. The qualitative results also show that the presented model can generate a detailed prediction.

REFERENCES

[1] Z. Zhao, Z. Jin, C. Luo, H. Fang, F. Basti, M. C. Gursoy, C. Caicedo, and Q. Qiu, "Temporal and spatial routing for large scale safe and connected uas traffic management in urban areas," in *2019 IEEE 25th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*. IEEE, 2019, pp. 1–6.

[2] A. Puri, "A survey of unmanned aerial vehicles (uav) for traffic surveillance," *Department of computer science and engineering, University of South Florida*, pp. 1–29, 2005.

[3] S. G. Gupta, M. M. Ghonge, and P. Jawandhiya, "Review of unmanned aircraft system (uas)," *International journal of advanced research in computer engineering & technology (IJARCET)*, vol. 2, no. 4, pp. 1646–1658, 2013.

[4] P. Liu, A. Y. Chen, Y.-N. Huang, J.-Y. Han, J.-S. Lai, S.-C. Kang, T.-H. Wu, M.-C. Wen, M.-H. Tsai *et al.*, "A review of rotorcraft unmanned aerial vehicle (uav) developments and applications in civil engineering," *Smart Struct. Syst*, vol. 13, no. 6, pp. 1065–1094, 2014.

[5] F. Nex and F. Remondino, "Uav for 3d mapping applications: a review," *Applied geomatics*, vol. 6, no. 1, pp. 1–15, 2014.

[6] M. Odelga, P. Stegagno, and H. H. Bülthoff, "Obstacle detection, tracking and avoidance for a teleoperated uav," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 2984–2990.

[7] H. L. N. N. Thanh and S. K. Hong, "Completion of collision avoidance control algorithm for multicopters based on geometrical constraints," *IEEE Access*, vol. 6, pp. 27 111–27 126, 2018.

[8] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[9] S. A. Bortoff, "Path planning for uavs," in *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No. 00CH36334)*, vol. 1, no. 6. IEEE, 2000, pp. 364–368.

[10] J. Tisdale, Z. Kim, and J. K. Hedrick, "Autonomous uav path planning and estimation," *IEEE Robotics & Automation Magazine*, vol. 16, no. 2, pp. 35–42, 2009.

[11] R. W. Beard and T. W. McLain, "Multiple uav cooperative search under collision avoidance and limited range communication constraints," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, vol. 1. IEEE, 2003, pp. 25–30.

[12] B. D. Song, J. Kim, and J. R. Morrison, "Rolling horizon path planning of an autonomous system of uavs for persistent cooperative service: Milp formulation and efficient heuristics," *Journal of Intelligent & Robotic Systems*, vol. 84, no. 1-4, pp. 241–258, 2016.

[13] Z. Jin, Z. Zhao, C. Luo, F. Basti, A. Solomon, M. C. Gursoy, C. Caicedo, and Q. Qiu, "Simulation of real-time routing for uas traffic management with communication and airspace safety considerations," *arXiv preprint arXiv:2002.11861*, 2020.

[14] D.-S. Jang, C. A. Ippolito, S. Sankararaman, and V. Stepanyan, "Concepts of airspace structures and system analysis for uas traffic flows for urban areas," in *AIAA Information Systems-AIAA Infotech@ Aerospace*, 2017, p. 0449.

[15] Z. Zhao, K. Pugdeethosapol, S. Lin, Z. Li, C. Ding, Y. Wang, and Q. Qiu, "Learning topics using semantic locality," in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 3710–3715.

[16] H. Fang, A. Shrestha, Z. Zhao, Y. Li, and Q. Qiu, "An event-driven neuromorphic system with biologically plausible temporal dynamics," in *38th IEEE/ACM International Conference on Computer-Aided Design, ICCAD 2019*. Institute of Electrical and Electronics Engineers Inc., 2019, p. 8942083.

[17] H. Fang, A. Shrestha, Z. Zhao, Y. Wang, and Q. Qiu, "A general framework to map neural networks onto neuromorphic processor," in *20th International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2019, pp. 20–25.

[18] Y. Lu, Y. Li, and S. Velipasalar, "Efficient human activity classification from egocentric videos incorporating actor-critic reinforcement learning," in *2019 IEEE International Conference on Image Processing (ICIP)*, Sep. 2019, pp. 564–568.

[19] Y. Lu and S. Velipasalar, "Autonomous human activity classification from wearable multi-modal sensors," *IEEE Sensors Journal*, vol. 19, no. 23, pp. 11 403–11 412, Dec 2019.

[20] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu, "Multi-agent tensor fusion for contextual trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 126–12 134.

[21] Z. Shi, M. Xu, Q. Pan, B. Yan, and H. Zhang, "Lstm-based flight trajectory prediction," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[22] Y. Li, H. Eslamiat, N. Wang, Z. Zhao, A. K. Sanyal, and Q. Qiu, "Autonomous waypoints planning and trajectory generation for multi-rotor uavs," *Proceedings of Design Automation for CPS and IoT*, 2019.

[23] H. Eslamiat, Y. Li, N. Wang, A. K. Sanyal, and Q. Qiu, "Autonomous waypoint planning, optimal trajectory generation and nonlinear tracking control for multi-rotor uavs."

[24] Z. Zhao, C. Luo, J. Zhao, Q. Qiu, M. C. Gursoy, C. Caicedo, and F. Basti, "A simulation framework for fast design space exploration of unmanned air system traffic management policies," in *2019 Integrated Communications, Navigation and Surveillance Conference (ICNS)*. IEEE, 2019, pp. 1–10.

[25] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[26] Z. Xie, Y.-H. Huang, G.-Q. Fang, H. Ren, S.-Y. Fang, Y. Chen, and J. Hu, "Routenet: Routability prediction for mixed-size designs using convolutional neural network," in *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2018, pp. 1–8.