Inter-Departmental Ticket Processing Time Prediction using Multi-Task Learning

Dongdong Zhang[1]†, Xinyu Yang[2]†, Yuanbo Tang[3], Naifan Zhang[3], Yiwen Liu[3], Feifan He[3], Shuhan Huang[4], Yang Li[3]*
[1]National Graduate College for Engineers, Tsinghua University
[2]College of Computer Science, Sichuan University
[3]Shenzhen Key Laboratory of Ubiquitous Data Enabling,
Tsinghua Shenzhen International Graduate School, Tsinghua University
[4]School of Computer Science, China University of Geosciences

***Abstract*—When customers submit problem tickets to the customer-service department, these requests are channeled into standardized processing workflows. Ensuring real-time visibility of progress updates and accurate estimation of anticipated resolution timelines is critical for maintaining positive customer experience. However, the complexity of manual processing time estimation arises from multiple interdependent factors, including ticket source, priority level, and incident classification. While part of these information are available, the difficulty of accessing others creates substantial discrepancies in handling durations across different ticket categories, thereby compounding the challenges of temporal prediction. Besides, full historical paths of ticket transitions between departments are rarely available during real-time prediction scenarios, where conventional single-task approaches may suffer from overfitting. To address these issues, we proposed a novel Multi Task Transformer-LSTM model (MT-TLM) that can simultaneously optimize three tasks: predicting subsequent service department routing, estimating department-specific processing durations, and estimating total resolution timelines. And our multi-task design has demonstrated better generalizability and robustness compared to some classic machine learning models, ensemble learning models, single-task, and multi-task deep learning models.**

***Index Terms*—ML(machine learning), Time series, MTL(Multi-task learning), ticket processing.**

I. Introduction

High-quality customer service is a key driver of company performance [23]. Timely resolution of concerns boosts satisfaction and retention. Effective service fosters loyalty and trust, encouraging repeat customers and enhancing reputation and profitability [18]. Similarly, in government and nonprofit settings, efficiently handling community requests—like pothole reports or legal advice—is vital for public trust. In corporate support, predicting response times for requests, such as emails [1], improves satisfaction and efficiency [20]. Optimizing response times is thus crucial for customer satisfaction and business success.

Our research scenario is the handling of complaint tickets on a public service platform. For instance, a resident annoyed by a neighbor's noisy renovation outside working hours submits a complaint via a community app after failing to resolve it directly. Customer service transfers the ticket to a government mediation center, then to the civil disputes department. As mediation proceeds, various departments manage tasks like contacting parties, investigating, and mediating until resolution. We aim to predict the total processing time upon receipt and give the resident an estimated resolution time, updating it based on case progress (e.g., inaccurate reports) and feedback (e.g., further mediation requests). Fig. 1 is a sample of the ticket transfer process. For a detailed description of the problem, see section III-A.

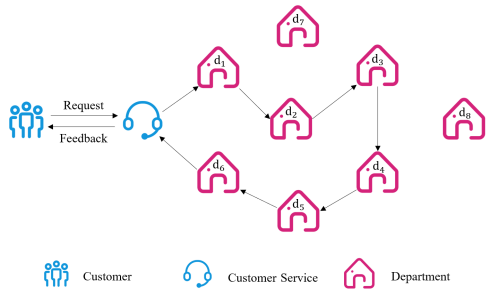

Fig. 1. The process of handling customer requests, passing through multiple departments (d_i).

So far, some studies have used classical machine learning, like Random Forest [2], Xgboost [5], and neural networks such as LSTM [11] and Transformer [24], to predict ticket processing times, proving their efficiency and feasibility. Yet, when tickets involve multiple departments, transfers between them add significant uncertainty to the total time.

Predictive Analytics (PA) uses current and historical data to forecast future outcomes [13]. For our task, Transformer models perform well in multi-output scenarios like speech separation [21], multi-scale attention effectively captures complex sequential features [16], and hybrid models, such as VAE-LSTM, which combine LSTM with other modules, have proven effective for time series tasks [14]. Inspired by Cao et al. [3] and Liu et al. [15], we combine Transformer and LSTM to model both long-range and short-range dependencies, enhancing complex feature representation. Unlike single-task models that overfit when predicting only time, our multi-task framework improves generalization by sharing features and leveraging task relationships [4].

In this paper, we design three loss functions. The entire

model will predict three different tasks: 1) the next department, 2) the processing time of the next department, and 3) the total processing time of the ticket. Additionally, we use a Transformer-LSTM-based encoder to encode the transfer sequence, employing a self-attention layer to capture long-range dependencies in the input features. Meanwhile, we analyzed and cleaned real community data, and the experimental results on this dataset demonstrate the advantages of our model.

The main contributions and innovations of the model are as follows:

1) We combine the self-attention mechanism of Transformer [24] with the LSTM [11] network to effectively capture long-range dependencies in department sequences while retaining the advantages of LSTM in time series modeling.
2) We employ three loss functions that mutually constrain each other, using a multi-task learning (MTL) [4] strategy with next department prediction and departmental processing time prediction as auxiliary tasks. This multi-task approach enhances the predictive performance of each task.
3) We validated the effectiveness of this method using community ticket data. Extensive experimental results show that our structure significantly outperforms single-task models and other common algorithms.

II. Related Work

A. Time Prediction for Customer Service

In 2020, Borg et al. [1] used Random Forest [2] to analyze 51,682 customer support emails, demonstrating its ability to predict response times and improve communication efficiency. In 2022, Haw et al. [10] evaluated neural networks, AdaBoost [9], and Random Forest [2] for predicting ticket resolution times, showing their effectiveness in enhancing service efficiency and customer satisfaction. While both studies highlight the utility of machine learning in optimizing customer support, neither addresses the prediction of ticket flow paths or multi-departmental influences, which are central to our task.

B. Time Prediction for Traffic Trajectory Modeling

Transportation trajectory modeling [25] is highly relevant to our task. Transportation trajectory modeling uses historical trajectory data to analyze and predict movement patterns of transportation modes like vehicles, pedestrians, and public transit. By processing data from GPS and other sources, extracting features such as speed and direction, and employing machine learning techniques like RNNs [8], LSTMs [11], and Graph Neural Networks, it aims to forecast future trajectories and travel times.

Wang et al. [26] proposed MTNet, which decomposes trajectories into map-matched road sequences with spatiotemporal features, using meta-learning to predict trajectory patterns accurately. Shao et al. [19] introduced TrajForesee, leveraging sparse urban traffic data and fine-grained GPS information to predict urban vehicle trajectories using spatiotemporal embedding and dynamic graph convolution. These models predict both the movement paths of traffic participants (e.g., vehicles and pedestrians) and overall travel time, similar to our task. However, department flow paths are concentrated on a few key departments, unlike traffic road maps, and customer service data differs significantly from traffic trajectory data.

C. LSTM-related and Transformer-related models

LSTM (Long Short-Term Memory network), an enhanced version of RNN(Recurrent Neural Network), overcomes gradient vanishing and exploding issues in traditional RNNs when handling long sequences. The classic LSTM has been proven effective in time series prediction tasks, such as RSSI-based indoor localization [6].

Transformer, a neural network architecture based entirely on attention mechanisms, excels in capturing long-range dependencies in data through self-attention, making it highly effective for tasks involving extended sequences. Transformer models have achieved success in multi-output scenarios such as speech separation [21].

III. Proposed Method

A. Problem Formulation

Consider a company with N departments, denoted as d_i where $i \in \{1, 2, \ldots, N\}$. When a customer service request (ticket) is received, it needs to be processed by a subset of these departments. Let K be the number of departments involved in processing a given ticket, with $K \leq N$. The sequence of departments through which the ticket passes is denoted by

$$S = (d_{i_1}, d_{i_2}, \ldots, d_{i_K}) \tag{1}$$

where $\{i_1, i_2, \ldots, i_K\} \subseteq \{1, 2, \ldots, N\}$.

The total processing time t is defined as the time elapsed from when customer service receives the ticket to when the final processing result is returned to the customer. In practical applications, the total processing time t is used as a unified metric instead of summing individual department times, as discrepancies between predicted and actual sequences can alter node counts, amplifying total processing time errors. Therefore, instead of summing the processing time of each department, we predict the total processing time t directly.

To improve the accuracy of ticket processing time prediction and reduce the impact of sequence discrepancies, we propose a multi-task learning model to predict the following tasks:

- The next department d_{next} in the sequence.
- The processing time t_d for the current department.
- The total processing time t for the ticket.

B. Overall Structure of the Framework

The entire model framework is shown in Fig. 2. Our model uses a multi-task learning approach to predict ticket department sequences and processing times. It takes department sequence data, along with temporal and categorical features, as input. The model simultaneously predicts the next department, its processing time, and the total processing time. By combining self-attention, dual-layer LSTM, and cyclic encoding, it

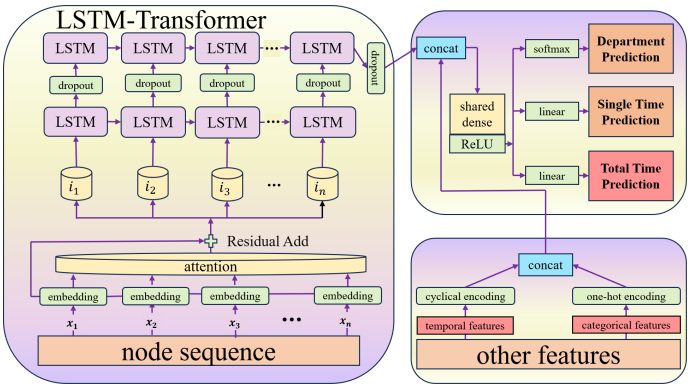

Fig. 2. Schematic of multi-task Transformer-LSTM model architecture.

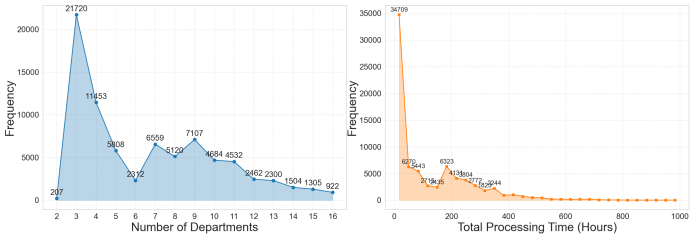

Fig. 3. Distribution of ticket transfer counts and total processing times.

captures temporal and sequential patterns, improving accuracy within a unified framework.

The model begins with an embedding layer converting department indices into high-dimensional vectors, followed by self-attention to improve feature representation. A residual connection refines features while retaining input data. A dual-layer LSTM captures sequence dependencies, and cyclic encoding handles temporal periodicity. Categorical features use one-hot encoding. Combined features pass through a shared ReLU-activated hidden layer, with task-specific outputs: softmax for node prediction and regression for time prediction, enhancing performance by leveraging shared patterns.

C. LSTM-Transformer Encoder

Our LSTM-Transformer structure is shown in Fig. 2, where the area labeled "LSTM-Transformer" represents the encoder.

The embedding layer converts node indices into fixed-dimensional vectors to capture semantic information. A multi-head self-attention layer then processes these vectors to detect long-range dependencies, improving feature representation. A residual connection and layer normalization stabilize training by combining the attention output with the original embedding. Next, a dual-layer LSTM extracts sequence patterns: the first layer captures basic features across all time steps, followed by Dropout to prevent overfitting, and the second layer focuses on global temporal patterns from the last time step, also with Dropout for regularization.

This structure processes input sequentially through embedding, self-attention, and LSTM, integrating node semantics, global dependencies, and temporal dynamics. The resulting features support classification or prediction tasks, enhancing the model's ability to handle complex time series by extracting multi-dimensional node features effectively.

D. Multi-Task Loss Function Design

In MT-TLM, the multi-task loss function integrates three task-specific losses: department prediction, processing time prediction, and total time prediction. These are defined as follows:

- **Department Prediction Loss**:
  $L_{\text{department\_output}} = -\sum_{i=1}^{M}\sum_{j=1}^{C} c_{ij}\log(\hat{c}_{ij})$, a cross-entropy loss over M samples and C classes, where c_{ij} is the true label and $\hat{c}_{ij}$ is the predicted probability for sample i and class j.
- **Processing Time Loss**:
  $L_{\text{time\_output}} = \frac{1}{M}\sum_{i=1}^{M}\left(t_{d,i} - \hat{t}_{d,i}\right)^2$, the mean squared error (MSE) between true $(t_{d,i})$ and predicted $(\hat{t}_{d,i})$ processing times for the current department across M.
- **Total Time Loss**:
  $L_{\text{total\_time\_output}} = \frac{1}{M}\sum_{i=1}^{M}\left(t_i - \hat{t}_i\right)^2$, the MSE between true (t_i) and predicted $(\hat{t}_i)$ total processing times over M.

The combined loss is:

$$L_{\text{total}} = w_1 L_{\text{department\_output}} + w_2 L_{\text{time\_output}} + w_3 L_{\text{total\_time\_output}} \quad (2)$$

In this formulation, the weights w_1, w_2, and w_3 correspond to the department output, processing time output, and total time output losses, respectively. Given the primary importance of total time prediction, we assign $w_3 = 0.5$ to prioritize it. The department and processing time predictions, which are equally significant and interdependent, are each assigned weights of $w_1 = 0.25$ and $w_2 = 0.25$. Cross-validation experiments validate that this weighting scheme ($w_1 = 0.25$, $w_2 = 0.25$, $w_3 = 0.5$) achieves an optimal balance of performance across all tasks.

IV. Experiment Results and Analysis

A. Data Analysis

The dataset includes about 92,000 customer demand records from Shenzhen neighborhoods (July 2022–October 2024), tracking requests from submission to resolution. After removing unprocessed or misrouted entries, 88,000 valid records remain. Most tickets need 3–5 transfers, with cases over 16 transfers excluded. Data is split 8:1:1 into training, validation, and test sets (random seed 5140). It features 9 categorical variables (e.g., community, event source) encoded with one-hot or target methods, and 3 cyclic temporal features (hour, month, weekday). Processing times typically range from 0–400 hours, with outliers above 500 hours affecting averages and challenging the model.

Fig. 3 shows ticket transfer counts and processing times. Most tickets need 3–5 transfers, with those over 16 deemed invalid and removed. Processing times are typically 0–400 hours, but outliers above 500 or 1,000 hours, though rare, raise the average and challenge the model.

Model	Multi-task		Single-task	
	Node Prediction	Time Prediction	Node Prediction	Time Prediction
	Acc	RMSE / MAE	Acc	RMSE / MAE
SVR	/	/	/	383.22 / 318.49
Kalman Filter	/	/	/	206.17 / 127.62
Gaussian Process	/	/	/	194.24 / 96.86
Random Forest	/	/	/	187.57 / 104.82
CatBoost	/	/	/	141.08 / 73.90
LightGBM	/	/	/	134.07 / 70.44
XGBoost	/	/	/	133.66 / 70.96
Transformer	85.28%	142.88 / 72.37	83.47%	150.07 / 75.22
Seq2Seq-LSTM	86.66%	135.22 / 78.04	86.53%	142.57 / 80.42
LSTM	86.47%	131.39 / 73.43	86.13%	138.71 / 78.95
MT-TLM (WD)	84.66%	138.03 / 77.25	83.30%	132.98 /72.19
MT-TLM(Ours)	**87.04%**	**125.82 / 66.16**	86.35%	133.98 / 69.47

TABLE I
COMPARISON OF MODEL PERFORMANCE IN MULTI-TASK AND SINGLE-TASK SETTINGS, SHOWING ACCURACY (ACC) AND ERROR METRICS (RMSE/MAE) FOR NODE AND TIME PREDICTION TASKS.

B. Preprocessing and Experiment Setup

We evaluated department prediction accuracy and total processing time RMSE separately on a machine with 32GB RAM, an i5-13600KF CPU, and a 4070s GPU. To simulate real-world prediction without full node sequences, we augmented data by creating multiple versions of each record (1 to n nodes for an n-node sequence), expanding the training and test sets to about 470,000 and 59,000 records, respectively. Community distribution of the original dataset is in Tab. II. Model performance (Accuracy, RMSE, MAE) for multi-task and single-task settings is in Tab. I. Parameters included: embedding dimension 512, hidden units 512, learning rate 0.0001, dropout 0.2, and 48 attention heads.

C. Result

We tested three classic machine learning models—SVR (Support Vector Regression), KF (Kalman Filter), GP (Gaussian Process), four ensemble models—Random Forest [2], CatBoost [17], LightGBM [12] and XGBoost [5]—and five deep learning models, including Transformer [24], Seq2Seq-LSTM [22], and LSTM [11]. Seq2Seq models, like those used in Transformer-based speech recognition [7], are common in time series tasks. We also evaluated two multi-task models: MT-TLM and MT-TLM (WD). All multi-task models had an extra ReLU-activated fully connected layer, except MT-TLM (WD), which omitted the shared Dense layer. Total processing time was assessed on a larger original test set, while accuracy used an augmented set. In real scenarios, customer requests submitted via an app or mini-program are manually assigned to various initial departments by dispatchers based on input information and submission channels, with subsequent departments unknown. To simulate this, we emptied the node sequences in all test sets, allowing the model to predict the total processing time without prior node information.

Vertically, ensemble learning models and classic machine learning models struggle with our data, and Transformer underperforms compared to LSTM-based models, which achieve 86% node prediction accuracy for time series.

Horizontally, deep learning models excel in multi-task settings, except MT-TLM (WD), where multi-task metrics lag behind single-task ones. Multi-task learning reuses knowledge and cuts labeling costs, while multi-objective optimization boosts performance over simple weighting or single-task training.

Community Name	Test Set		Training Set	
	Data Count	Percentage (%)	Data Count	Percentage (%)
Shuijing	2409	34.90	19244	34.87
Sanlian	1162	16.83	9430	17.09
Lihu	1096	15.88	8969	16.25
Cuihu	893	12.94	7048	12.77
Gankeng	770	11.15	5844	10.59
Guanghua	362	5.24	2882	5.22
Others(62 Communities)	211	3.06	1774	3.21
Sum	6903	100.00	48191	100.00

TABLE II
DATA COUNT AND PERCENTAGE FOR EACH COMMUNITY IN TEST AND TRAINING SETS

D. Discussion on Regional Results

This discussion assesses the model's adaptability and performance with uneven community data distribution. By comparing its performance in communities with more and less data, we identify limitations and guide future improvements. This enhances the model's generalization, accuracy, stability, and reliability with imbalanced data in real-world use. The original data spans 68 communities but is unevenly distributed (Tab. II). Six communities exceed 5% of the data, while 62 have less than 2%. We exclude small-sample communities and focus on generalizability and scalability in the top 6 datasets.

Community	Quantity	Proportion	RMSE	MAE
Shuijing	2409	30.88%	123.94	58.71
Sanlian	1162	14.90%	97.17	55.11
Lihu	1096	14.05%	74.20	43.27
Cuihu	893	11.45%	141.16	64.08
Gankeng	770	9.87%	160.62	87.01
Guanghua	362	4.64%	141.78	84.47

TABLE III
METRICS FOR DIFFERENT COMMUNITIES.

Model Adaptability Across Communities We grouped MT-TLM results from Tab. I by community (Tab. III). Shuijing, the largest, aligns with overall results. The model performs better in data-rich communities (Sanlian, Lihua) but average in data-poor ones (Cuihu, Gankeng, Guanghua). More data boosts accuracy; less data limits it, showing varied data traits across communities in one city.

Future work could improve low-data community predictions with more sampling or augmentation.

Model Adaptability with Partial Node Sequence Fig. 4 shows the link between node length and data quantity. We tested metrics for node sequences of 9 to 16, from just ticket features to full department sequences. As known nodes increase, RMSE drops and the accuracy increases, showing the key role of context in the prediction of the sequence. More initial nodes help the model capture patterns, boosting accuracy across sequence lengths.

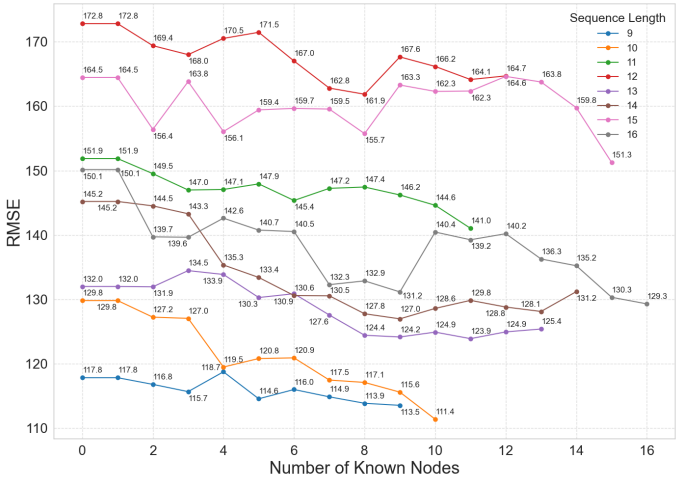

Fig. 4. The chart shows the variation in RMSE values with increasing prefix length for different sequence lengths (from 9 to 16). Each line represents data for a different sequence length, with colors distinguishing the curves for each sequence length.

Lower RMSE with more nodes highlights improved performance with more data, vital for real-world cases with partial info. In our request-handling case, early-stage insights enhance later predictions, aiding decisions and resource use.

V. Conclusion

MT-TLM merges LSTM, Transformer, and self-attention in a multi-task framework with three inter-constrained losses, improving real-time ticket flow prediction. Our study shows it excels in predicting appeal processing times, aiding businesses in boosting user experience, retention, and internal optimization.

Future research should tackle data imbalance with augmentation or synthetic data to boost model generalization across communities. Exploring tailored multi-task architectures, like attention- or graph-based models, could improve handling of complex department dependencies.

We hope the MT-TLM model excels in both experiments and real-world applications, inspiring further research and industry innovation in multi-task learning for customer service optimization.

References

[1] Anton Borg, Jim Ahlstrand, and Martin Boldt. Predicting e-mail response time in corporate customer support. In *22nd International Conference on Enterprise Information Systems, ICEIS 2020 Prague, Virtual, Online, 5 May 2020 through 7 May 2020*, pages 305–314. SciTePress, 2020.

[2] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[3] Kangjie Cao, Ting Zhang, and Jueqiao Huang. Advanced hybrid lstm-transformer architecture for real-time multi-task prediction in engineering systems. *Scientific Reports*, 14(1):4890, 2024.

[4] Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

[5] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[6] Su Fong Chien, David Chieng, Samuel YC Chen, Charilaos C Zarakovitis, Heng Siong Lim, and YH Xu. Applying hybrid quantum lstm for indoor localization based on rssi. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 131–135. IEEE, 2024.

[7] Linhao Dong, Shuang Xu, and Bo Xu. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5884–5888. IEEE, 2018.

[8] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[9] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[10] Su Cheng Haw, Kyle Ong, Lit Jie Chew, Kok Why Ng, Palanichamy Naveen, and Elham Abdulwahab Anaam. Improving the prediction resolution time for customer support ticket system. *Journal of System and Management Sciences*, 12(6):1–16, 2022.

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[12] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[13] Vaibhav Kumar and ML Garg. Predictive analytics: a review of trends and techniques. *International Journal of Computer Applications*, 182(1):31–37, 2018.

[14] Shuyu Lin, Ronald Clark, Robert Birke, Sandro Schönborn, Niki Trigoni, and Stephen Roberts. Anomaly detection for time series using vae-lstm hybrid model. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4322–4326. Ieee, 2020.

[15] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1871–1880, 2019.

[16] Daliang Ouyang, Su He, Guozhong Zhang, Mingzhu Luo, Huaiyong Guo, Jian Zhan, and Zhijie Huang. Efficient multi-scale attention module with cross-spatial learning. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[17] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

[18] David J Reibstein. What attracts customers to online stores, and what keeps them coming back? *Journal of the academy of Marketing Science*, 30:465–473, 2002.

[19] Kangjia Shao, Yang Wang, Zhengyang Zhou, Xike Xie, and Guang Wang. Trajforesee: How limited detailed trajectories enhance large-scale sparse information to predict vehicle trajectories? In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 2189–2194. IEEE, 2021.

[20] Amit Sheoran, Sonia Fahmy, Matthew Osinski, Chunyi Peng, Bruno Ribeiro, and Jia Wang. Experience: Towards automated customer issue resolution in cellular networks. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pages 1–13, 2020.

[21] Cem Subakan, Mirco Ravanelli, Samuele Cornell, Mirko Bronzi, and Jianyuan Zhong. Attention is all you need in speech separation. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 21–25. IEEE, 2021.

[22] I Sutskever. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*, 2014.

[23] Ofir Turel, Yufei Yuan, and Catherine E Connelly. In justice we trust: predicting user acceptance of e-customer services. *Journal of Management Information Systems*, 24(4):123–151, 2008.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[25] Sheng Wang, Zhifeng Bao, J Shane Culpepper, and Gao Cong. A survey on trajectory data management, analytics, and learning. *ACM Computing Surveys (CSUR)*, 54(2):1–36, 2021.

[26] Yong Wang, Guoliang Li, Kaiyu Li, and Haitao Yuan. A deep generative model for trajectory modeling and utilization. *Proceedings of the VLDB Endowment*, 16(4):973–985, 2022.