Data Selection through Scenario-Balanced Coresets for Trajectory Prediction

Anonymous submission

Abstract

Modern vehicles are equipped with multiple information-collection devices, continuously generating a large volume of raw data. Accurately predicting the trajectories of neighboring vehicles is a vital component in understanding the complex driving environment. Yet, training trajectory prediction models is challenging in two ways. Processing the large-scale data is computation-intensive. Moreover, easy-medium driving scenarios often overwhelmingly dominate the dataset, leaving challenging driving scenarios such as dense traffic under-represented. In this paper, to mitigate data redundancy in the over-represented driving scenarios and to reduce the bias rooted in the data scarcity of complex ones, we propose a novel data-efficient training method based on coreset selection. This method strategically selects a small but representative subset of data while balancing the proportions of different scenario difficulties. To the best of our knowledge, we are the first to introduce a method capable of effectively condensing large-scale trajectory dataset, while achieving a state-of-the-art compression ratio. Notably, even when using only 50% of the Argoverse dataset, the model can be trained with little to no decline in performance. Moreover, the selected coreset maintains excellent generalization ability.

Introduction

Understanding the driving environment is crucial for the safe navigation of autonomous vehicles. Modern vehicles use sensors and cameras to generate vast amounts of data, enabling trajectory prediction models that capture both agent-agent and agent-infrastructure interactions (Zhou et al. 2022; Feng et al. 2023; Chai et al. 2019). The practice of training trajectory prediction models is challenging in two ways. First, training complex deep neural networks requires vast amounts of data, leading to extremely high computational costs. For example, training a SceneTransformer (Ngiam et al. 2021) model (15.3M parameters) on 0.2M trajectory samples consumes thousands of GPU hours. This dilemma raises the following pivotal question:

Q1: How can we identify and maintain the most critical data without significant accuracy degradation?

The second challenge is that standard machine learning tasks use the weighted average accuracy as the performance metric (Johnson and Khoshgoftaar 2019), giving more weight to scenarios with more samples. This biases the model towards well-represented scenarios, putting the under-represented scenarios at risk. We aim to address the following question:

Q2: How to mitigate the bias rooted in the unbalance of different driving scenarios?

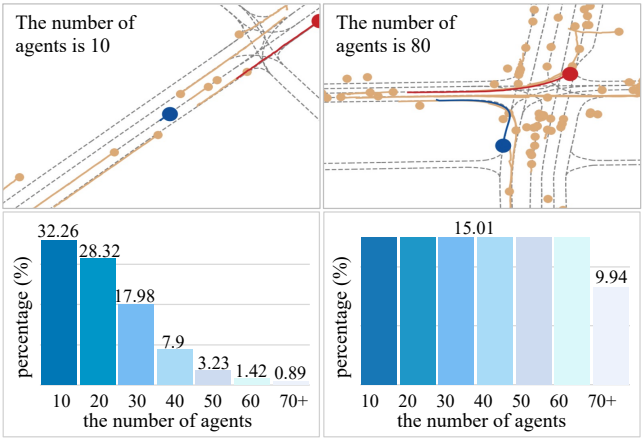

Figure 1: **Top**: Visualization of trajectory prediction scenarios with different traffic densities (i.e., number of agents) in the Argoverse1.1 (Chang et al. 2019). **Bottom left**: Scenario difficulty distribution in the training set. **Bottom right**: Scenario difficulty distribution in the 10% coreset selected by our balanced method. Numbers above bars indicate the proportion of each difficulty level.

In this paper, to address these two questions, we develop a data-efficient and balanced training method. Specifically, we focus on selecting a small coreset of trajectory data that significantly contributes to the model, while balancing the proportion of data across varying difficulty levels.

The concept of coreset was first introduced by approximating the k-means clustering centers and constructing a small weighted subset that maintains a clustering cost close to that of the original data (Har-Peled and Mazumdar 2004). They have since gained traction in NLP (Zhou et al. 2023a; Huang et al. 2022) and computer vision (Killamsetty et al. 2021) Recently, inspired by knowledge distillation (Hinton 2015), dataset distillation (Wang et al. 2018; Zhao, Mopuri, and Bilen 2020) compresses large datasets into smaller synthetic sets, achieving similar performance to full-data training. Coreset selection and data distillation both aim for data compression, but the latter is computationally expensive for large datasets (Zhou et al. 2023a). Coreset selection over-

comes this by choosing a small, representative subset from the original data, avoiding the cost of synthesizing new data.

Applying coreset selection methods to the field of trajectory prediction presents considerable challenges. Unlike the fixed dimensions of image data, the dimensions of trajectory prediction data are heterogeneous across driving scenarios and dynamic over time. Specifically, the number of vehicles in a scene directly affects the data's dimensionality; the more vehicles involved, the higher the data dimension. This heterogeneity can be substantial; in the Argoverse motion prediction dataset 1.1 (Chang et al. 2019), the number of vehicles in a scene can range from just a few to over 200. What's more, the mobility of vehicles results in varying data dimensions in time.

Contributions. We develop a novel data-efficient training method based on coreset selection. To the best of our knowledge, this is the first work on coreset construction for trajectory prediction. Our approach consists of multiple steps. We preprocess the dataset to group the data according to their difficulty levels (measured by the traffic density in a scene). Then, we calculate the submodular gain for each sample and select the most valuable data. To more effectively manage the coreset budget, we design two different coreset selection methods: *fixed selection* and *balanced selection*. Experimental results show that the fixed selection method achieves similar performance to the full dataset with just 50% of the data (minADE difference of 0.022, minFDE difference of 0.047). The balanced method excels in complex scenarios, with lower data distribution variance. We tested these coresets on HPNet, the current state-of-the-art (SOTA) model, and found that their performance across different difficulty scenarios closely matches that of the backbone model.

Related Work

Trajectory Prediction. Early trajectory prediction methods (Kaempchen et al. 2004; Ammoun and Nashashibi 2009; Batz, Watson, and Beyerer 2009) focused on modeling a single agent's path without considering interactions. Recent methods (Gao et al. 2020; Liang et al. 2020) shifted to vectorization-based techniques to better capture road geometry and context. Transformer models (Wang et al. 2024; Giuliari et al. 2021; Xu et al. 2020) emerged to model complex multi-agent interactions (Liu et al. 2021; Ngiam et al. 2021; Zhou et al. 2022, 2023b; Tang et al. 2024).

Training Data Selecting. Training complex neural networks on large datasets requires high computation resources. To address this, data-efficient approaches emerged, including optimization algorithms and dataset distillation. Optimization methods (Robbins and Monro 1951; Sutskever et al. 2013; Kingma 2014; Duchi, Hazan, and Singer 2011) improve training efficiency by accelerating convergence and using adaptive learning rates. Data distillation methods focused on directly reducing dataset size (Wang et al. 2018; Zhao, Mopuri, and Bilen 2020; Cazenavette et al. 2022). However, data distillation methods are computationally expensive, as they require synthesizing new data and repeatedly optimizing complex objectives (Zhao, Mopuri, and Bilen 2020). In contrast, coreset selection has proven more efficient by directly selecting a small representative subset from the original data, avoiding the high computational costs associated with generating synthetic datasets while retaining critical information for effective model training (Wang et al. 2022; Coleman et al. 2019; Paul, Ganguli, and Dziugaite 2021; Zhou et al. 2023a). Applying coreset techniques to our problem is challenging due to the high dimensionality of the data, and the dynamic nature of each driving scene.

Problem Formulation

The training dataset $\mathscr{D} = \{S^j\}_{j=1}^n$ consists of n driving scenarios/scenes. Each driving scenario (i.e. each sample) is described by a triple $S = (X, Y, \mathscr{M})$, where X and Y are the collections of observed and future trajectories of neighboring agents (an agent can be a vehicle or a pedestrian), and $\mathscr{M}$ is the map. Let m denote the number of agents in the scenario, then X and Y can be expressed as $X = \{x_1, ..., x_m\}$ and $Y = \{y_1, ..., y_m\}$, where $x_i \in \mathbb{R}^{2\times T_{obs}}$ and $y_i \in \mathbb{R}^{2\times T_{pre}}$ are the two-dimensional observed and future trajectory coordinates of agent i, with lengths T_{obs} and T_{pre}, respectively.

We quantify the difficulty of a scenario S^j by its traffic density (i.e., the number of agents involved), denoted as m^j. Let $m^* = \max_{j\in[n]} m^j$. Let $\mathscr{L} = \{l_1, l_2, \cdots, l_{\max}\}$ denote the user-specific difficulty fidelity. For example, one can set $l_1 = \{1, \cdots, 10\}$, $l_2 = \{11, \cdots, 20\}$, $\cdots$, $l_{\max} = \{m^* - (m^* \mod 10) + 1, \cdots, m^*\}$. For ease of exposition, in this example, we assume $(m^* \mod 10) \neq 0$. In this example, class l_1 is the group of difficulties $1 \sim 10$, and the same applies to other classes. With a bit of abuse of notation, let

$$L_k := \{S^j \subseteq \mathscr{D} : m^j \in l_k\} \text{ for } k = 1, \cdots, \max,$$

be a partition of $\mathscr{D}$ for the given data difficulty fidelity $\mathscr{L}$. Our goal is to efficiently and meticulously pinpoint a subset of data $\mathscr{C} \subseteq \mathscr{D}$ such that (1) a model trained on $\mathscr{C}$ performs comparably to one trained on the entire dataset $\mathscr{D}$, and (2) the proportion of data in $\mathscr{C}$ across different difficulty levels $\mathscr{L}$ is as close to uniform as possible.

Method

Managing Heterogeneity in Data Dimensions. Unlike fixed-dimension image data, trajectory prediction data dimensions vary with the number of agents in the scene. In addition, a given agent may be present or absent at different time frames. This is because trajectory prediction data is temporally continuous, with agents moving at varying speeds and limited sensor ranges. We address these challenges by preprocessing the trajectory data in two steps.

Filtering: For each scene, we determine its difficulty m by ranking all involved agents in descending order based on their frequency of appearance in the scene duration, and removing agents that appear infrequently.

Categorization: We partition the dataset into subsets based on user-specified difficulty fidelity $\mathscr{L}$, and obtain $L_1, L_2, \cdots, L_{\max}$.

Coreset Selection. We use HiVT-64 as the backbone for coreset construction, applying a submodular function P to evaluate the contribution of each sample. We propose two coreset constructions, which we refer to as *fixed selection*

and *balanced selection*, respectively. Since they share a common structure, we describe them together in Algorithm 1. In Algorithm 1, $\Delta(S^j \mid \mathscr{C}) := P(\mathscr{C} \cup \{S^j\}) - P(\mathscr{C})$ is the submodular gain of adding S^j to the current coreset $\mathscr{C}$. Notably, the identified coreset is not limited to the HiVT-64 model but can be effectively used to train other trajectory prediction models, as demonstrated in our experiments.

Fixed Selection. This method uses a fixed sample ratio of data to include in the coreset. Let α represent the fixed ratio (e.g., $\alpha = 0.1$ for 10%). The number of samples selected from each difficulty level $l_k \in \mathscr{L}$ is determined by $n_k = \alpha \times |L_k|$, where $|L_k|$ is the total number of samples in L_k. To select the most important samples to iteratively build $\mathscr{C}$, we utilize the following submodular function

$$P(S^j) = \sum_{S^i \in \mathscr{C}} |f(S^i) - f(S^j)|^2 - \sum_{S^i \in \mathscr{D} \setminus \mathscr{C}} |f(S^i) - f(S^j)|^2, \quad (1)$$

where $f(\cdot)$ represents the objective function of HiVT-64, composed of Laplace Negative Log-Likelihood Loss and Soft Target Cross-Entropy Loss, $\mathscr{C}$ is the set of already selected samples, S^j is the sample currently being considered for inclusion in the coreset, and S^i represents the samples already in the coreset. The first term, $\sum_{S^i \in \mathscr{C}} |f(S^i) - f(S^j)|^2$, measures the similarity between the candidate sample S^j and the samples S^i in $\mathscr{C}$. This term assesses how well S^j aligns with the samples already selected in $\mathscr{C}$. The second term, $\sum_{S^i \in \mathscr{D} \setminus \mathscr{C}} |f(S^i) - f(S^j)|^2$, measures the similarity between S^j and the samples in the remainder of the dataset $\mathscr{D} \setminus \mathscr{C}$. This term evaluates how distinct S^j is from the samples that have not yet been selected. By maximizing $P(S^j)$, we select samples that are highly representative of the current subset $\mathscr{C}$ while being sufficiently distinct from $\mathscr{D} \setminus \mathscr{C}$.

Balanced Selection This method is designed to mitigate biases stemming from the distribution imbalance in scenes with heterogeneous difficulties. It focuses on selecting more challenging but less frequent samples to ensure a relatively balanced representation of difficulty levels in the final coreset. The selection budget is dynamically allocated between different difficulty levels to reduce bias towards over-represented, easier scenarios. We use the `DynamicBudget` function as shown in Algorithm 1 to determine n_k. That is, n_k is determined starting from $k = \max$ so that greater complexity are prioritized. The balanced selection method dynamically adjusts the budget based on the remaining coreset size and the number of difficulty levels.

Within each difficulty level l_k, similar to fixed selection, the balanced selection method also uses the given submodular optimization to select the most representative samples, as shown in Eq. (1). By focusing on more challenging but less frequent samples, the balanced selection method achieves a relatively balanced proportion of different difficulty levels in the selected coreset. The results are shown in Table 1.

Algorithm 1: Coreset Selection

Input: Entire dataset $\mathscr{D}$, ratio α, submodular function $P(\cdot)$, selection method (`Fixed` or `Balanced`)
Output: Selected coreset $\mathscr{C}$
Initialize: $\mathscr{C} \leftarrow \emptyset$;
for *difficulty level* $l_k \in \mathscr{L}$ ($k = \max, \cdots, 1$) **do**
  $\mathscr{C}_k \leftarrow \emptyset$;
  $n_{\text{rest}} \leftarrow |\mathscr{D}|$;
  if *selection method is* `Fixed` **then**
    $n_k \leftarrow \alpha \times |L_k|$;
  else
    $n_k \leftarrow \text{DynamicBudget}(n_{\text{rest}}, \alpha, k)$;
  for $j \leftarrow 1$ **to** n_k **do**
    $S^j \leftarrow \arg\max_{S^j \in l_k \setminus \mathscr{C}_k} \Delta(S^j \mid \mathscr{C}_k)$;
    $\mathscr{C}_k \leftarrow \mathscr{C}_k \cup \{S^j\}$;
  $\mathscr{C} \leftarrow \mathscr{C} \cup \mathscr{C}_k$;
  $\mathscr{D} \leftarrow \mathscr{D} \setminus \mathscr{C}_k$;
return $\mathscr{C}$;
Function `DynamicBudget` (n_{rest}, α, k) **:**
  budget $\leftarrow \alpha \times n_{\text{rest}}$;
  $l_{\text{rest}} \leftarrow k$;
  if $|L_k| \leq$ *budget* **then**
    $n_k \leftarrow |L_k|$;
  else
    $n_k \leftarrow \frac{\text{budget}}{l_{\text{rest}}}$;
  return n_k;

Experiments and Evaluations

Dataset, Metrics and Implementation Details. We use the Argoverse Motion Forecasting Dataset 1.1 (Chang et al. 2019) and evaluate performance with minADE, minFDE, and MR. MinADE measures average trajectory error, minFDE captures endpoint deviation, and MR is the percentage of endpoints with over 2 meters error. Using HiVT-64 (Zhou et al. 2022), we identified the best coreset by pretraining for 10 epochs and iteratively adding top samples based on submodular scores. We trained HiVT-64, HiVT-128 (Zhou et al. 2022), and HPNet (Tang et al. 2024) for 64 epochs on an RTX 2080 Ti using AdamW with a batch size of 32, learning rate 3×10^{-4}, weight decay 1×10^{-4}, and 0.1 dropout, applying cosine annealing for learning rate decay. HiVT-64 and HiVT-128 only differ in hidden units. No additional techniques like ensembles were used.

Experiment Results We use randomly sampled subsets as benchmarks to compare against our selected coresets (results in Figure 2). HiVT-64 trained on the full dataset achieves a minADE of 0.692 and minFDE of 1.047. Using 50% of the data, the fixed and balanced coresets achieve minADEs of 0.714 and 0.711, closely matching full dataset performance. HiVT-128 shows similar trends with these reduced coresets.

Generalization Performance Analysis. To assess the generalization capability of the selected coresets, we further tested them on HPNet(Tang et al. 2024), the current SOTA model. The models trained on half of the coreset data exhibited only marginal increases in minADE, with differences of 0.051 and 0.042, and minFDE differences of 0.132 and 0.089 for the fixed and balanced methods, respectively, compared to models trained on the full dataset. This indicates that our coreset selection methods can effectively reduce the training data while maintaining competitive performance.

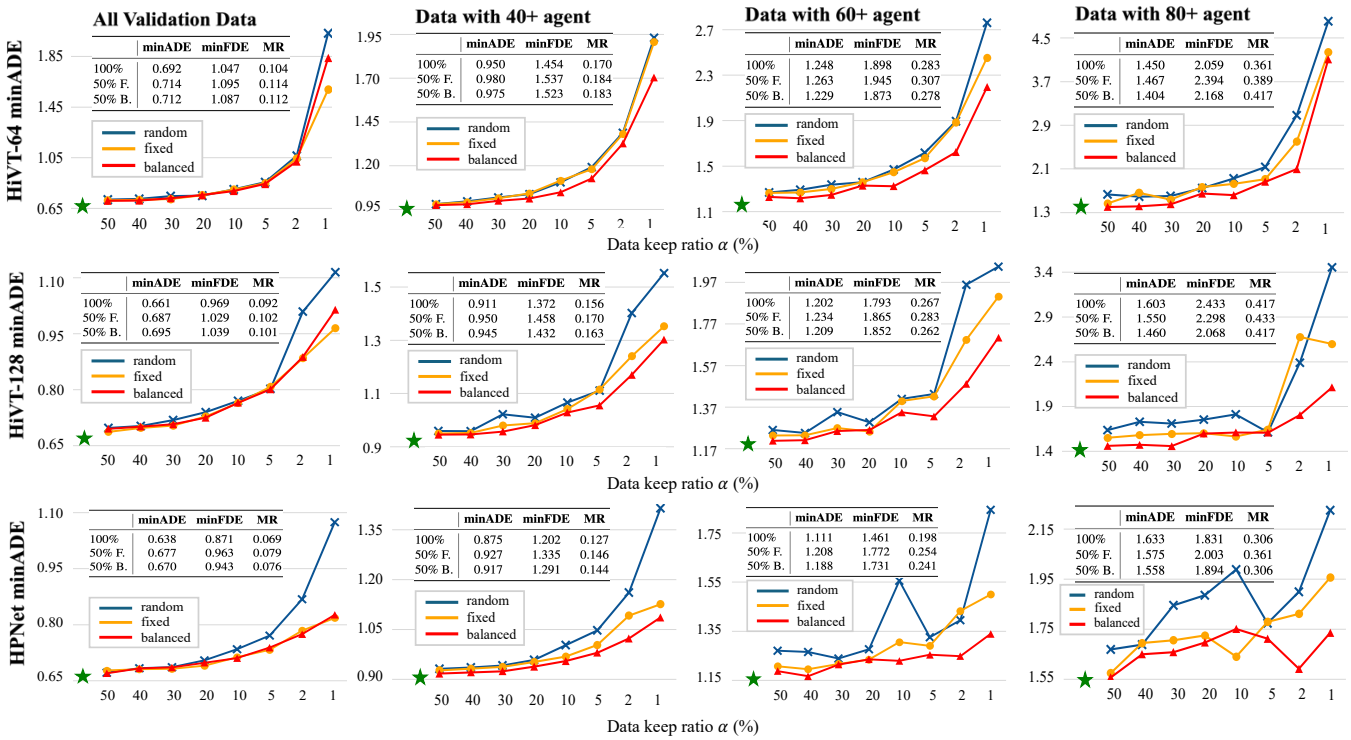

Figure 2: The coresets were tested on three models: **(top)** HiVT-64 (Zhou et al. 2022), **(middle)** HiVT-128 (Zhou et al. 2022), and **(bottom)** HPNet (Tang et al. 2024). The results are organized from left to right according to different validation set difficulties: **(left)** full set, **(left2)** ≥ 40 agents, **(right2)** ≥ 60 agents, and **(right)** ≥ 80 agents. The **x-axis** shows data keep ratios, and the **y-axis** shows minADE. Each figure includes a **star** for the minADE on the full Argoverse 1.1 dataset (Chang et al. 2019). Additionally, the embedded **table** presents the minADE, minFDE, and MR results for the models trained on the complete Argoverse dataset, alongside the results for coresets with a 50% data keep ratio selected using the fixed (F.) and balanced (B.) methods.

Analysis of Diverse Difficult Scenarios. We evaluated our methods in complex real-world scenarios, focusing on environments with many agents, like dense urban settings where autonomous systems may encounter interference from multiple vehicles. We partitioned the validation set based on the number of agents present in each scenario, creating new validation subsets with more than 40 agents, more than 60 agents, and more than 80 agents, respectively. Models were trained on the full dataset and the selected coresets, and tested on these subsets. The results are presented in Figure 2.

Across all validation subsets, the coresets selected by our balanced method consistently demonstrated excellent performance, particularly excelling in more complex scenarios. For in the validation set with over 60 agents, the balanced method achieved a minADE of 1.216 at a 40% keep ratio, compared to 1.292 for the random method. Additionally, the fixed method exhibited strong performance at higher data keep ratios, outperforming the random selection approach. When operating under low data keep ratios, the balanced method showed a more significant advantage over the other methods. Notably, the performance of the random method deteriorated considerably at low data keep ratios, underscoring the superiority of our fixed and balanced selection strategies in maintaining model accuracy with reduced data.

Insight into Data Distribution. The performance of trajectory prediction models is significantly influenced by the distribution of data within the training dataset. To quantify the data distribution in the Argoverse 1(Chang et al. 2019) training dataset, we calculated the proportion of samples for each scene category and the mean variance of the dataset, as shown in Table 1. Variance σ^2 is a measure of data spread, reflecting how evenly the data is distributed within a dataset. We categorized the dataset into 10 difficulty levels based on the number of agents per scene, from 10 to 90, grouping samples with over 100 agents into one level.

	α (%)	10	20	30	40	50	60+	σ^2
	All	32.26	28.32	17.98	7.90	3.23	2.31	140.12
random	50	33.98	31.44	20.05	8.53	3.57	2.85	162.60
balanced	50	23.45	23.45	23.45	17.41	7.12	5.11	101.53
	40	21.18	21.18	21.18	21.18	8.90	6.40	89.12
	30	19.90	19.90	19.90	19.90	11.87	8.53	75.71
	20	17.44	17.44	17.44	17.44	17.44	12.78	59.36
	10	15.01	15.01	15.01	15.01	15.01	24.95	39.62

Table 1: Distribution of scenario difficulty levels across different training sets and selection methods, along with corresponding mean variance (σ^2). The **top row** shows the full Argoverse 1(Chang et al. 2019) training set, the **second row** a random 50% subset, and the **following rows** coresets selected by our balanced method at various keep ratios. The **last column** shows mean variance, indicating deviation from the standard 10% proportion.

References

Ammoun, S.; and Nashashibi, F. 2009. Real time trajectory prediction for collision risk estimation between vehicles. In *2009 IEEE 5Th international conference on intelligent computer communication and processing*, 417–422. IEEE.

Batz, T.; Watson, K.; and Beyerer, J. 2009. Recognition of dangerous situations within a cooperative group of vehicles. In *2009 IEEE Intelligent Vehicles Symposium*, 907–912. IEEE.

Cazenavette, G.; Wang, T.; Torralba, A.; Efros, A. A.; and Zhu, J.-Y. 2022. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4750–4759.

Chai, Y.; Sapp, B.; Bansal, M.; and Anguelov, D. 2019. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. *arXiv preprint arXiv:1910.05449*.

Chang, M.-F.; Lambert, J.; Sangkloy, P.; Singh, J.; Bak, S.; Hartnett, A.; Wang, D.; Carr, P.; Lucey, S.; Ramanan, D.; et al. 2019. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8748–8757.

Coleman, C.; Yeh, C.; Mussmann, S.; Mirzasoleiman, B.; Bailis, P.; Liang, P.; Leskovec, J.; and Zaharia, M. 2019. Selection via proxy: Efficient data selection for deep learning. *arXiv preprint arXiv:1906.11829*.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).

Feng, C.; Zhou, H.; Lin, H.; Zhang, Z.; Xu, Z.; Zhang, C.; Zhou, B.; and Shen, S. 2023. Macformer: Map-agent coupled transformer for real-time and robust trajectory prediction. *IEEE Robotics and Automation Letters*.

Gao, J.; Sun, C.; Zhao, H.; Shen, Y.; Anguelov, D.; Li, C.; and Schmid, C. 2020. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11525–11533.

Giuliari, F.; Hasan, I.; Cristani, M.; and Galasso, F. 2021. Transformer networks for trajectory forecasting. In *2020 25th international conference on pattern recognition (ICPR)*, 10335–10342. IEEE.

Har-Peled, S.; and Mazumdar, S. 2004. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, 291–300.

Hinton, G. 2015. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*.

Huang, X.; Khetan, A.; Bidart, R.; and Karnin, Z. 2022. Pyramid-BERT: Reducing complexity via successive core-set based token selection. *arXiv preprint arXiv:2203.14380*.

Johnson, J. M.; and Khoshgoftaar, T. M. 2019. Survey on deep learning with class imbalance. *Journal of big data*, 6(1): 1–54.

Kaempchen, N.; Weiss, K.; Schaefer, M.; and Dietmayer, K. C. 2004. IMM object tracking for high dynamic driving maneuvers. In *IEEE Intelligent Vehicles Symposium, 2004*, 825–830. IEEE.

Killamsetty, K.; Zhao, X.; Chen, F.; and Iyer, R. 2021. Retrieve: Coreset selection for efficient and robust semi-supervised learning. *Advances in neural information processing systems*, 34: 14488–14501.

Kingma, D. 2014. Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Liang, M.; Yang, B.; Hu, R.; Chen, Y.; Liao, R.; Feng, S.; and Urtasun, R. 2020. Learning lane graph representations for motion forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, 541–556. Springer.

Liu, Y.; Zhang, J.; Fang, L.; Jiang, Q.; and Zhou, B. 2021. Multimodal motion prediction with stacked transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 7577–7586.

Ngiam, J.; Caine, B.; Vasudevan, V.; Zhang, Z.; Chiang, H.-T. L.; Ling, J.; Roelofs, R.; Bewley, A.; Liu, C.; Venugopal, A.; et al. 2021. Scene transformer: A unified architecture for predicting multiple agent trajectories. *arXiv preprint arXiv:2106.08417*.

Paul, M.; Ganguli, S.; and Dziugaite, G. K. 2021. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34: 20596–20607.

Robbins, H.; and Monro, S. 1951. A stochastic approximation method. *The annals of mathematical statistics*, 400–407.

Sutskever, I.; Martens, J.; Dahl, G.; and Hinton, G. 2013. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, 1139–1147. PMLR.

Tang, X.; Kan, M.; Shan, S.; Ji, Z.; Bai, J.; and Chen, X. 2024. Hpnet: Dynamic trajectory forecasting with historical prediction attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 15261–15270.

Wang, J.; Su, L.; Han, S.; Song, D.; and Miao, F. 2024. Towards safe autonomy in hybrid traffic: Detecting unpredictable abnormal behaviors of human drivers via information sharing. *ACM Transactions on Cyber-Physical Systems*, 8(2): 1–25.

Wang, K.; Zhao, B.; Peng, X.; Zhu, Z.; Yang, S.; Wang, S.; Huang, G.; Bilen, H.; Wang, X.; and You, Y. 2022. Cafe: Learning to condense dataset by aligning features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12196–12205.

Wang, T.; Zhu, J.-Y.; Torralba, A.; and Efros, A. A. 2018. Dataset distillation. *arXiv preprint arXiv:1811.10959*.

Xu, M.; Dai, W.; Liu, C.; Gao, X.; Lin, W.; Qi, G.-J.; and Xiong, H. 2020. Spatial-temporal transformer networks for traffic flow forecasting. *arXiv preprint arXiv:2001.02908*.

Zhao, B.; Mopuri, K. R.; and Bilen, H. 2020. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*.

Zhou, D.; Wang, K.; Gu, J.; Peng, X.; Lian, D.; Zhang, Y.; You, Y.; and Feng, J. 2023a. Dataset quantization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 17205–17216.

Zhou, Z.; Wang, J.; Li, Y.-H.; and Huang, Y.-K. 2023b. Query-centric trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17863–17873.

Zhou, Z.; Ye, L.; Wang, J.; Wu, K.; and Lu, K. 2022. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8823–8833.