Proceedings Paper:
Lee, S.I., Shin, D. and Park, J. (2026) Unseen data detection using routing entropy in mixture-of-experts for autonomous vehicles. In: 2025 40th IEEE/ACM International Conference on Automated Software Engineering (ASE). 40th IEEE/ACM International Conference on Automated Software Engineering (ASE), 16-20 Nov 2025, Seoul, South Korea. Institute of Electrical and Electronics Engineers (IEEE). ISBN: 9798350357349. ISSN: 1938-4300. EISSN: 2643-1572.

https://doi.org/10.1109/ASE63991.2025.00332

Unseen Data Detection using Routing Entropy in Mixture-of-Experts for Autonomous Vehicles

Sang In Lee
Chungnam National University
Daejeon, South Korea
sangin.lee.life@o.cnu.ac.kr

Donghwan Shin
University of Sheffield
Sheffield, United Kingdom
d.shin@sheffield.ac.uk

Jihun Park
Chungnam National University
Daejeon, South Korea
jihun.park@cnu.ac.kr

***Abstract*—Unseen data that differ significantly from the training data can cause machine learning models to behave unpredictably, which is particularly problematic in safety-critical systems like autonomous vehicles. Detecting such data, commonly called out-of-distribution (OOD) data, is essential for ensuring the robustness of these models. Existing methods often rely on the model's final output, which are limited since the model can be overconfident on unseen data. In this paper, we propose *Routing Entropy*, a novel OOD detection method that leverages the internal routing behavior of Mixture-of-Experts (MoE) models, a design increasingly adopted in modern neural networks. We hypothesize that MoE models exhibit high confidence routing for in-distribution (ID) inputs, but greater uncertainty for OOD inputs. We quantify this uncertainty by calculating the entropy of the routing scores for a given input. Experimental results on a MoE-based semantic segmentation model used for perception in autonomous driving demonstrate that Routing Entropy is effective on its own and, more importantly, provides a complementary signal to existing output-based methods. Combining Routing Entropy with an existing method significantly improves OOD detection performance. These results suggest that leveraging internal routing behavior of MoE models is a promising direction for robust OOD detection.**

***Index Terms*—Out-of-distribution detection, uncertainty quantification, mixture-of-experts, routing entropy**

I. Introduction

Autonomous vehicles have become a key application area where machine learning (ML) models serve as critical software components [1]. As ML components are increasingly integrated into functionalities such as perception, planning, and control in autonomous driving systems (ADS), ensuring their quality has become a critical concern in software engineering [2]. Traditional quality assurance methods, built on assumptions of deterministic logic and specified behavior, are not well-suited to handle the non-deterministic nature of ML-based systems [3]. Modern ML models, especially deep learning-based models, are often "black boxes," making their internal logic impossible for engineers to fully understand. This lack of transparency makes it difficult to verify and ensure the reliability of ADS [4].

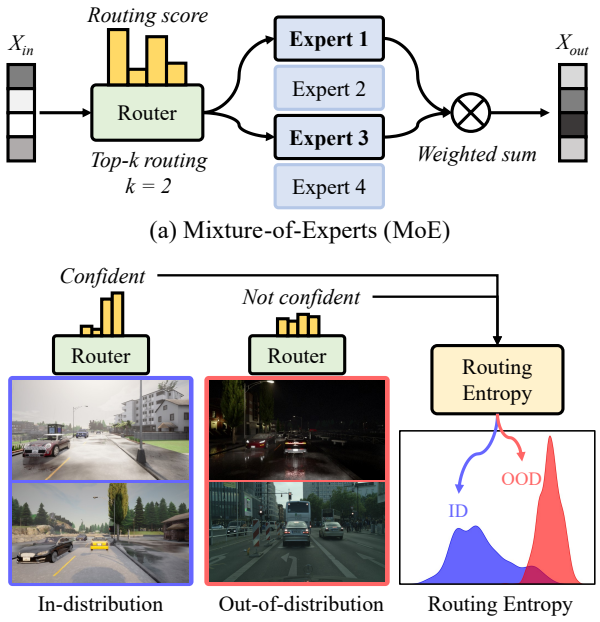

Fig. 1. Routing Entropy-based OOD detection in a Mixture-of-Experts (MoE) model. **(a) Illustration of a MoE layer,** the router selects the top-k expert networks based on routing scores, and their outputs are combined. **(b) Proposed method,** which uses Routing Entropy to detect unfamiliar inputs. In-distribution (ID) samples produce low entropy (confident routing), while out-of-distribution (OOD) samples produce high entropy (uncertain routing).

A primary threat to the ADS is their behavior on *unseen* data, also called *out-of-distribution* (OOD) data. Since ML models are fundamentally data-driven, their performance heavily depends on the training data [5]. When these models encounter OOD data—inputs that deviate from the training data distribution, they may produce unreliable or even dangerously incorrect outputs [6]. This problem is especially crucial in safety-critical systems such as ADS.

To address this challenge, it is essential to "*know what they don't know*" by quantifying uncertainty and detecting OOD inputs. By managing uncertainty, these techniques contribute to runtime monitoring, allowing systems to recognize when their predictions may be untrustworthy [7]. In safety-

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.RS-2022-00155857, 50%, Artificial Intelligence Convergence Innovation Human Resources Development (Chungnam National University)) and by the IITP (Institute of Information & Coummunications Technology Planning & Evaluation)-ITRC (Information Technology Research Center) grant funded by the Korea government (Ministry of Science and ICT) (IITP-2025-RS-2020-II201795, 50%). Donghwan Shin is supported by the UK Engineering and Physical Sciences Research Council (EPSRC) [EP/Y014219/1].

critical domains, this capability enables fail-safe mechanisms or fallback strategies that ensure safe operation under uncertainty [8, 9]. Furthermore, quantified uncertainty can inform test data generation by identifying regions of the input space where the model is less confident, guiding more effective and targeted software testing [10, 11].

Most existing OOD detection and uncertainty quantification methods for deep neural networks (DNNs) focus on analyzing the model's final output, such as softmax probabilities [12, 13] or output variability [14, 15]. These methods are based on the hypothesis that models will exhibit greater deviation and lower confidence in their predictions for OOD inputs compared to in-distribution (ID) inputs. While this is a natural approach for black-box deep learning models, it becomes problematic when models produce overconfident predictions on OOD inputs, which can be dangerous in safety-critical applications [13].

In this paper, we focus on internal uncertainty signals within a rising paradigm in modern neural network design, the *Mixture-of-Experts* (MoE) [16]. This design has recently gained considerable attention as it enables efficient scaling of model capacity (i.e., the number of parameters) while maintaining computational efficiency [17, 18, 19]. As shown in Fig. 1(a), this is achieved by an internal routing mechanism, which selectively activates a small subset of *experts* (i.e., sub-networks) for each input. During training, the model learns to predict which experts are best suited to process a given input, and each expert becomes specialized in handling the specific types of data assigned to it. We hypothesize that the routing behavior itself is a rich source of uncertainty information, as it tends to route familiar ID inputs with high confidence, while showing greater uncertainty for OOD inputs.

Based on this hypothesis, we propose *Routing Entropy*, a novel OOD detection method that quantifies the uncertainty in expert selection for MoE models. We detect OOD inputs by analyzing the entropy of routing decisions, which we quantify by calculating the Shannon entropy of the routing score distribution. This process is conceptually illustrated in Fig. 1(b). Our evaluation of a MoE-based semantic segmentation model demonstrates that Routing Entropy is an effective and competitive method for detecting OOD inputs. Furthermore, we demonstrate that Routing Entropy provides a complementary signal to existing output-based methods. Our results reveal that combining Routing Entropy with existing method significantly improves OOD detection performance across various scenarios. These results suggest that leveraging the internal routing behavior of MoE models is a promising direction for developing more robust and reliable OOD detection systems.

We summarize our contributions as follows:

(1) We propose Routing Entropy, a novel OOD detection method that quantifies the uncertainty in the routing decisions of Mixture-of-Experts (MoE) models. This provides a new perspective on internal model behavior for OOD detection.

(2) We validate the effectiveness of our proposed method across a range of real-world scenarios that autonomous vehicles may encounter, using a diverse OOD dataset built with a high-fidelity simulator

(3) We demonstrate that Routing Entropy provides complementary information to output-based uncertainty methods, and that their combination significantly enhances OOD detection performance.

II. Proposed Method

The Mixture-of-Experts (MoE) consists of a *router* and a set of N sub-networks, called experts. The core of MoE is conditional computation, which activates a sparse subset of experts, and this selection is handled by the router. Fig. 1(a) illustrates the process of a standard MoE layer with a top-k routing policy. For a given input X_{in}, the router computes a vector of routing scores over all N experts. Top-k selection is then applied, where only the k experts with the highest scores are chosen to process the input. The final output of the MoE layer X_{out} is then calculated as a weighted sum of the outputs from these selected experts, with the weights based on their routing scores.

While routing scores are mainly used for expert selection, we observe that their distribution also reflects the model's uncertainty in its decisions. For familiar ID inputs, the router tends to assign high scores to a few experts, showing confidence decisions. In contrast, for unfamiliar OOD inputs, the scores are more evenly spread across experts, indicating uncertainty. To quantify this behavior, we propose Routing Entropy, which measures the uncertainty of the routing distribution. A confident routing will produce low entropy, while a high entropy indicates an uncertain routing.

The calculation of the Routing Entropy is as follows. For a given input x, the router produces a vector of routing score logits $L(x) = [l_1, l_2, \ldots, l_N]$ for N available experts. We first convert these logits into a probability distribution $P(x) = [p_1, p_2, \ldots, p_N]$ using the Softmax function:

$$p_i(x) = \frac{e^{l_i}}{\sum_{j=1}^{N} e^{l_j}}.$$

From this probability distribution, we then calculate the Routing Entropy $H_{\text{routing}}(x)$ using the Shannon entropy formula:

$$H_{\text{routing}}(x) = -\sum_{i=1}^{N} p_i(x) \log p_i(x).$$

This scalar value represents our OOD detection score. A higher score indicates a higher likelihood of the input being OOD.

Fig. 1(b) shows the concept of our method. For typical ID inputs, such as clear-day driving scenes, the router confidently assigns high scores to a few specific experts. Conversely, for OOD inputs, such as nighttime scenes or domain shifts (e.g., from synthetic to real-world image), the router produces a flatter, more uniform score distribution. By calculating the Routing Entropy, our method translates these patterns into a scalar value. As shown on the Fig. 1(b) right, ID inputs produce low entropy values (blue), while OOD inputs produce high values (red). The separability between these two distributions allows for OOD detection with a simple threshold.

III. Evaluation

We evaluate the proposed Routing Entropy method with the following research questions:

RQ1 (Effectiveness): How effective is the Routing Entropy compared to baseline OOD detection methods?

RQ2 (Complementarity): Does utilizing both Routing Entropy and an output-based method achieve better performance than when either method is used alone?

RQ3 (Optimal Combination): Which combination of Routing Entropy and an output-based baseline shows the best performance?

A. Experimental Setup

Model. We trained a semantic segmentation model for autonomous driving scenarios. We use Mask2Former [21], a state-of-the-art Transformer-based architecture known for its strong performance on various segmentation tasks. To incorporate our method, we modified its architecture by replacing the standard Feed-Forward Network (FFN) in the final layer of the Transformer decoder with a Mixture-of-Experts FFN layer. We configured this MoE layer with $N = 8$ experts and a top-k routing policy where $k = 2$. Since Mask2Former operates with a multiple input-output structure due to its internal design, we take the mean of their OOD scores to obtain a single score per image for further evaluation.

Training. As the focus of our work is OOD detection rather than segmentation performance, we utilized the CARLA simulator [22] to generate both ID and OOD data. Our training data consists of 20,000 images captured under various but "normal" daytime driving conditions. The model was trained on this dataset until it converged without data augmentation.

OOD Data. To evaluate the OOD detection capabilities of our method, we created five types of OOD datasets. These datasets were designed to simulate a range of challenges that an autonomous vehicle might encounter, which are sparsely represented in standard training datasets. The OOD types cover three main categories of distribution shift: perceptual changes (e.g., lighting, weather), data corruption from potential sensor failures, and the domain gap between synthetic and real-world data. An ID test set was also used in evaluation, which is generated under the same conditions as the training data. The five OOD types are described as follows:

- ***Low Illumination***: Nighttime scenes with limited lighting, evaluating day-to-night shifts.
- ***Heavy Fog***: Reduced visibility due to dense fog, representing adverse weather conditions.
- ***Blur***: Gaussian blur applied to images, mimicking focus loss or motion blur.
- ***Noise***: Random black pixels injected into images to simulate sensor or transmission errors.
- ***Syn-to-Real***: Real-world images from Cityscapes [23] to evaluate synthetic to real-world domain shifts.

Baselines. To ensure a fair and practically meaningful comparison, we limit our evaluation to training-free methods. Methods that require access to OOD data for additional training [24, 25] or hyperparameter tuning [26, 27] are not considered. These approaches typically assume availability of a portion of OOD samples during development, which is a significant limitation in scenarios where such examples are unknown. In contrast, our method does not rely on any additional training or tuning with OOD data. Therefore, we consider the following baselines that are training-free:

- **Maximum Softmax Probability (MSP)** [12]: A standard baseline for OOD detection. This uses the maximum value of the model's final softmax probability vector as a confidence score. The intuition is that models produce lower confidence scores for OOD inputs.
- **Prediction Entropy** [20]: Also known as Softmax Entropy, this evaluates the entire distribution of the final prediction. It is calculated by Shannon Entropy formula to the softmax probabilities of the prediction logits. A higher entropy means greater uncertainty in the model's prediction, which suggests a potential OOD input.
- **Gini Coefficient** [13]: This method, recently proposed for OOD detection, calculates the Gini coefficient—a measure of inequality from economics—to the model's final softmax probabilities. It quantifies the level of disparity in the prediction confidence distribution.

Evaluation Metrics. We measure the false positive rate of OOD samples when true positive rate of ID samples is at 95% (FPR95) and the area under the Receiver Operating Characteristic curve (AUROC) as threshold independent metrics. These metrics quantify how well a method can distinguish between ID and OOD data and are widely adopted in prior works on OOD detection [12, 24, 27].

B. RQ1 Results: Effectiveness

Table I shows our evaluation results across the five OOD types. To answer RQ1, we evaluate Routing Entropy (RE) as a standalone method against the baselines. The results in the top four rows of Table I show that RE is highly effective, though its performance varies depending on the OOD type. RE shows a substantial improvement over all baselines in three OOD types. Notably, in the *Syn-to-Real* scenario, RE achieves an FPR95 of 2.29%, while baselines score above 25%. It shows similarly dominant performance on *Low Illumination* (7.40% FPR95) and *Noise* (17.50% FPR95). However, RE shows clear limitation in the *Heavy Fog* scenario. Its performance is significantly lower than baselines, particularly Prediction Entropy (79.20% vs. 95.56% in AUROC).

The reason why the model's routing behavior is highly sensitive to some OOD types but less so others is a topic for future investigation. We argue that this observation opens a promising direction: to deeply analyze the characteristics of MoE routing behavior and understand which types of distribution shifts it can and cannot effectively capture.

C. RQ2 Results: Complementarity

To answer RQ2, we investigate if a combined approach improves OOD detection by utilizing both our internal routing

TABLE I
EVALUATION RESULTS ON VARIOUS OOD TYPES

Method \ Type	Low Illum.		Heavy Fog		Blur		Noise		Syn-to-Real	
	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑	FPR95 ↓	AUROC ↑
Standalone Method										
MSP [12]	38.18	88.42	24.43	92.71	52.25	81.25	63.96	77.92	25.78	91.30
Pred. Entopy [20]	38.91	86.02	**12.24**	**95.56**	50.99	80.72	69.22	72.43	26.25	89.69
Gini Coef. [13]	38.91	84.47	21.77	93.91	50.10	79.80	63.44	72.81	29.90	87.12
Rout. Entropy (ours)	7.40	98.38	31.67	79.20	39.79	80.49	17.50	93.18	2.29	99.60
Combination with Routing Entropy										
MSP + Rout. Entropy	5.42	99.11	24.84	83.70	32.97	83.78	14.90	94.54	0.52	99.84
Pred. + Rout. Entropy	**2.97**	**99.50**	15.21	93.63	**27.34**	**89.05**	**12.76**	**95.95**	**0.00**	**99.96**
Gini + Rout. Entropy	3.54	99.45	17.19	92.27	27.45	88.38	13.33	95.68	0.00	99.95
Combination of Baselines										
MSP + Pred. Entropy	39.37	86.88	16.61	95.19	51.51	80.99	69.06	73.95	26.67	90.41
MSP + Gini Coef.	38.91	85.66	22.34	93.82	50.68	80.24	63.28	74.23	29.43	88.38
Pred. + Gini Coef.	38.80	85.33	18.28	94.81	50.73	80.30	66.04	72.58	27.55	88.52

Note: **Bold** indicates the best performance for each OOD type. Underline marks the best performance within standalone method.

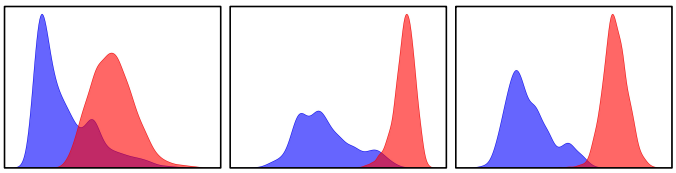

Fig. 2. OOD score distributions on the *Syn-to-Real* scenario. Distributions of scores for Prediction Entropy (left), Routing Entropy (middle), and their combination (right). Blue and red represent ID and OOD data, respectively. Note that the distribution shapes for the two individual methods are visibly different. The combined method clearly shows improved separability.

signal and an external prediction signal. We create these combined scores using a simple additive approach. We compute a combined score by adding the Routing Entropy and the baseline score. For methods like MSP, and Gini coefficient, which produce higher scores for confident predictions, we negate the values before adding them.

The results in Table I demonstrate that this combination strategy is highly effective. We first observe from the bottom three rows of Table I that combining baselines with each other (e.g., MSP + Pred. Entropy) yields no meaningful performance gain, indicating they capture redundant information.

In contrast, combining RE with any baseline leads to a significant improvement. The improvement is noteworthy in the *Blur* scenario, while RE alone (80.49% AUROC) performs similarly with the baselines, combining it with Prediction Entropy (Pred. + Rout. Entropy) boosts the AUROC to 89.05%. This suggests that even when the individual signals are not overwhelmingly strong, they capture complementary aspects of the OOD input. Similar gains are observed in *Low Illumination*, *Noise*, and *Syn-to-Real*. This complementary nature is visually illustrated in Fig. 2, which shows the score distributions for the *Syn-to-Real* scenario. We observe that Prediction Entropy and our Routing Entropy produce visibly different distribution shapes for both ID (blue) and OOD (red) data. When these two scores are additively combined, the resulting distributions for ID and OOD data become more separated (Fig. 2 right).

However, in the *Heavy Fog* scenario where RE's standalone performance is particularly poor, combining it actually degrades the performance (FPR95 12.24% → 15.21% in Pred. Entropy). This result indicates that if the routing signal itself is too noisy for a certain OOD type, a simple additive combination can harm an already effective output-based method.

D. RQ3 Results: Optimal Combination

Finally, to answer RQ3, we identify which combination of Routing Entropy and a baseline method yields the best performance. The results in the middle section of Table I consistently show that combining Routing Entropy with Prediction Entropy (Pred. + Rout. Entropy) is the most effective approach across all evaluated OOD types, with its scores marked in **bold**.

An interesting result from the *Noise* scenario shows the unique synergy between these two entropy-based methods. In this case, the Gini coefficient is better standalone baseline than Prediction Entropy. However, the Pred. + Rout. Entropy combination is still superior to the Gini + Rout. Entropy combination. This suggests that the effectiveness of a combination depends not just on the performance of the individual methods, but on the compatibility of the signals they represent.

We believe the synergy arises from using entropy to measure uncertainty at two consecutive stages: routing and prediction. Future work could explore more advanced approaches and combination methods to better leverage the complementary information from these for OOD detection.

IV. CONCLUSION

In this paper, we propose *Routing Entropy*, a novel method that leverages the internal routing mechanism of Mixture-of-Experts (MoE) models for out-of-distribution (OOD) detection. We found that it appears to capture different aspects of model uncertainty compared to existing methods that analyze the model's final output. This complementarity was confirmed when combining routing entropy with an output-based method, which consistently led to significantly better OOD detection performance. We hope this work contributes to opening a

new direction of research in OOD detection by leveraging the internal behaviors of modern neural network architectures.

REFERENCES

[1] M. R. Bachute and J. M. Subhedar, "Autonomous driving architectures: insights of machine learning and deep learning algorithms," *Machine Learning with Applications*, vol. 6, p. 100164, 2021.

[2] S. Martínez-Fernández, J. Bogner, X. Franch, M. Oriol, J. Siebert, A. Trendowicz, A. M. Vollmer, and S. Wagner, "Software engineering for ai-based systems: a survey," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 31, no. 2, pp. 1–59, 2022.

[3] G. Giray, "A software engineering perspective on engineering machine learning systems: State of the art and challenges," *Journal of Systems and Software*, vol. 180, p. 111031, 2021.

[4] T.-D. Nguyen, H. Tian, B. Le, P. Thongtanunam, and S. McIntosh, "A systematic survey on debugging techniques for machine learning systems," *arXiv preprint arXiv:2503.03158*, 2025.

[5] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[6] S. Khatiri, F. M. Amin, S. Panichella, and P. Tonella, "When uncertainty leads to unsafety: Empirical insights into the role of uncertainty in unmanned aerial vehicle safety," *arXiv preprint arXiv:2501.08908*, 2025.

[7] Q. Rahman, P. Corke, and F. Dayoub, "Run-time monitoring of machine learning for robotic perception: A survey of emerging trends," *IEEE Access*, vol. PP, pp. 1–1, 01 2021.

[8] M. Weiss and P. Tonella, "Fail-safe execution of deep learning based systems through uncertainty monitoring," in *2021 14th IEEE conference on software testing, verification and validation (ICST)*. IEEE, 2021, pp. 24–35.

[9] R. Grewal, P. Tonella, and A. Stocco, "Predicting safety misbehaviours in autonomous driving systems using uncertainty quantification," in *2024 IEEE Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 2024, pp. 70–81.

[10] N. Walkinshaw and G. Fraser, "Uncertainty-driven black-box test data generation," in *2017 IEEE International Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 2017, pp. 253–263.

[11] M. Zhang, S. Ali, and T. Yue, "Uncertainty-wise test case generation and minimization for cyber-physical systems," *Journal of Systems and Software*, vol. 153, pp. 1–21, 2019.

[12] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *International Conference on Learning Representations*, 2017.

[13] H. Abdelkader, J.-G. Schneider, M. Abdelrazek, P. Rani, and R. Vasa, "Towards robust ml-enabled software systems: Detecting out-of-distribution data using gini coefficients," in *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, 2024, pp. 2289–2293.

[14] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*. PMLR, 2016, pp. 1050–1059.

[15] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in neural information processing systems*, vol. 30, 2017.

[16] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," in *International Conference on Learning Representations*, 2017.

[17] N. Du, Y. Huang, A. M. Dai, S. Tong, D. Lepikhin, Y. Xu, M. Krikun, Y. Zhou, A. W. Yu, O. Firat *et al.*, "Glam: Efficient scaling of language models with mixture-of-experts," in *International conference on machine learning*. PMLR, 2022, pp. 5547–5569.

[18] D. Dai, C. Deng, C. Zhao, R. Xu, H. Gao, D. Chen, J. Li, W. Zeng, X. Yu, Y. Wu *et al.*, "Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 1280–1297.

[19] J. Ludziejewski, M. Pióro, J. Krajewski, M. Stefaniak, M. Krutul, J. Małaśnicki, M. Cygan, P. Sankowski, K. Adamczewski, P. Miłoś, and S. Jaszczur, "Joint moe scaling laws: Mixture of experts can be memory efficient," in *Forty-second International Conference on Machine Learning*, 2025.

[20] J. Mukhoti, A. Kirsch, J. van Amersfoort, P. H. Torr, and Y. Gal, "Deep deterministic uncertainty: A new simple baseline," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 24 384–24 394.

[21] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 1290–1299.

[22] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[23] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[24] D. Hendrycks, M. Mazeika, and T. Dietterich, "Deep anomaly detection with outlier exposure," *arXiv preprint arXiv:1812.04606*, 2018.

[25] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty," *Advances in neural information processing systems*, vol. 32, 2019.

[26] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *International Conference on Learning Representations*, 2018.

[27] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," *Advances in neural information processing systems*, vol. 31, 2018.